NAME

pfm − Personal File Manager for Linux/Unix

SYNOPSIS

pfm [*directory*] [−s, −−swap *directory*] [−c, −−colorset *string*] [−l, −−layout *number*] [−−login] [−o, −−sort *mode*]

pfm { −−help | −−usage | −−version }

DESCRIPTION

pfm is a terminal-based file manager, based on PFM.COM for MS-DOS.

All pfm commands are accessible through one or two keystrokes, and a few are accessible with the mouse. Most command keys are case-insensitive. pfm can operate in single-file mode or multiple-file mode. In single-file mode, the command corresponding to the keystroke will be performed on the current (highlighted) file only. In multiple-file mode, the command will apply to a selection of files.

Note that throughout this manual page, *file* can mean any type of file, not just plain regular files. These will be referred to as *regular files*.

This manual pertains to pfm version 2.12.1.

OPTIONS

Most of pfm's configuration is read from a config file. The default location for this file is *$HOME/.pfm/.pfmrc*, but an alternative location may be specified using the environment variable PFMRC. If there is no config file present at startup, one will be created. The file contains many comments on the available options, and is therefore supposed to be self-explanatory. pfm will issue a warning if the version number of an existing config file is less than the version of pfm you are running, and will offer to update it to its own version. Alternatively, you could let pfm create a new default config file and compare the changes with your own settings, so that you can configure the new options right away. See also the **C**onfig command under ''MORE COMMANDS'' below, and ''DIAGNOSIS''.

There are two commandline options that specify starting directories. The CDPATH environment variable is taken into account when pfm tries to find these directories.

directory
: The directory that pfm should initially use as its main directory. If unspecified, the current directory is used.

−−help
: Print extended usage information, then exit.

−c, −−colorset *string*
: Start pfm using the specified colorset (as defined in the *.pfmrc*).

−l, −−layout *number*
: Start pfm using the specified column layout (as defined in the *.pfmrc*).

−−login
: Start pfm as a login shell. pfm will change its name to −pfm and start a new session using *setsid()*.

−o, −−sort *mode*
: Start pfm using the specified sort mode (as shown by the **F6** command).

−s, −−swap *directory*
: The directory that pfm should initially use as swap directory. (See also below, under the **F7** command).

  There would be no point in setting the swap directory and subsequently returning to the main directory if 'persistentswap' is turned off in your config file. Therefore, pfm will swap back to the main directory *only* if 'persistentswap' is turned on.

−−usage
: Print concise usage information, then exit.

−−version
Print current version, then exit.

NAVIGATION

Navigation through directories is essentially done using the arrow keys and the *vi* (1) cursor keys (**hjkl**). The following additional navigation keys are available:

Movement inside a directory:

up arrow, *down arrow*	move the cursor by one line
k, **j**	move the cursor by one line
-, **+**	move the cursor by ten lines
CTRL-E, **CTRL-Y**	scroll the screen by one line
CTRL-U, **CTRL-D**	move the cursor by half a page
CTRL-B, **CTRL-F**	move the cursor by a full page
PgUp, **PgDn**	move the cursor by a full page
HOME, **END**	move the cursor to the top or bottom line
SPACE	mark the current file, then move the cursor one line down

Movement between directories:

l, *right arrow*	*chdir()* to a subdirectory
h, *left arrow*	*chdir()* to the parent directory
ENTER	*chdir()* to a subdirectory
ESC, **BS**	*chdir()* to the parent directory

If the option 'chdirautocmd' has been specified in the *.pfmrc* file, pfm will execute that command after every *chdir()*. This can be used *e.g.* to set the title of an xterm window.

Note: the **l** and **ENTER** keys function differently when the cursor is on a non-directory file (see below under **L**ink and ''LAUNCHING FILES'' respectively).

COMMANDS

Attribute

Changes the mode of the file if you are the owner. The mode may be specified either symbolically or numerically, see *chmod* (1) for more details.

Note 1: the mode on a symbolic link cannot be set. See *chmod* (1) for more details.

Note 2: the name **Attribute** for this command is a reminiscence of the DOS version.

Copy

Copy current file. You will be prompted for the destination filename. Directories will be copied recursively with all underlying files.

In multiple-file mode, it is not allowed to specify a single non-directory filename as a destination. Instead, the destination name must be a directory or a name containing a **=1** or **=2** escape (see below under c**O**mmand).

If clobber mode is off (see below under the **!** command), existing files will not be overwritten unless the action is confirmed by the user.

Whether **C**opy follows symlinks or copies them is OS-specific; you can change the default behavior by setting the 'copyoptions' option in your *.pfmrc*. See *cp* (1).

Delete

Delete a file or directory. You must confirm this command with **Y** to actually delete the file. If the current file is a directory which contains files, and you want to delete it recursively, you must respond with **A**ffirmative to the additional prompt. Lost files (files on the screen but not actually present on disk) can be deleted from the screen listing without confirmation. Whiteouts cannot be deleted; use un**W**hiteout for this purpose.

Edit

Edit a file with your external editor. You can specify an editor with the environment variable VISUAL or EDITOR or with the 'editor' option in the *.pfmrc* file. Otherwise *vi* (1) is used.

If a capital **E** is pressed, the foreground editor is used ('fg_editor').

Find

If the current sort mode is by filename, you are prompted for a (partial) filename. While you type, the cursor is positioned on the best match. Type **ENTER** to end typing.

If the current sort mode is not by filename, then you are prompted for a filename. The cursor is then positioned on that file.

tarGet

Allows you to change the target that a symbolic link points to. You must have permission to remove the current symbolic link.

The new name may use the **=0** escape to modify the existing target (see below under c**O**mmand).

Include

Allows you to mark a group of files which meet a certain criterion:

After / **B**efore
: files newer/older than a specified date and time

Every file
: all files, including dotfiles (but not the . and .. entries)

Files only
: regular files of which the filenames match a specified regular expression (not a glob pattern!)

Directories only
: all directories (but not the . and .. entries)

Greater / **S**maller
: files that are bigger or smaller than the provided size in bytes.

Newmarks
: files which were created during a previous multiple command and are now denoted with a *newmark* (**~**)

Oldmarks
: files which were marked (*****) before a previous multiple command and are now denoted with an *oldmark* (**.**)

User
: files owned by the current user

.
: all dotfiles (but not the . and .. entries).

Oldmarks and newmarks may be used to perform more than one command on the same group of files.

Note that the . and .. entries will never automatically become marked.

There is also the option:

Invert
: Inverts the selection, except for the . and .. entries, which become unmarked.

Link

Prompts to create either:

an **A**bsolute symlink
: This will create a symlink containing an absolute path to the target, irrespective of whether you enter a relative or an absolute symlink name.

  Example: when the cursor is on the file */home/rene/incoming/.plan*, and you request an absolute

symlink to be made with either the name *../.plan* or */home/rene/.plan*, the actual symlink will become:

```
/home/rene/.plan -> /home/rene/incoming/.plan
```

a **H**ard link
: This will create an additional hard link to the current file with the specified name, which must be on the same filesystem (see *ln* (1)).

a **R**elative symlink
: This will create a symlink containing a relative path to the target, irrespective of whether you enter a relative or an absolute symlink name.

  Example: when the cursor is on the file */home/rene/incoming/.plan*, and you request a relative symlink to be made with either the name *../.plan* or */home/rene/.plan*, the actual symlink will become:

```
/home/rene/.plan -> incoming/.plan
```

If a directory is specified, `pfm` will follow the behavior of *ln* (1), which is to create the new link inside that directory.

In multiple-file mode, it is not allowed to specify a single non-directory filename as a new name. Instead, the new name must be a directory or a name containing a **=1** or **=2** escape (see below under c**O**mmand).

If clobber mode is off (see below under the **!** command), existing files will not be overwritten.

Note that if the current file is a directory, the **l** key, being one of the *vi* (1) cursor keys, will *chdir()* you into the directory. The capital **L** command will *always* try to make a link.

More
: Presents you with a choice of operations not related to the current file. Use this *e.g.* to configure `pfm`, edit a new file, make a new directory, show a different directory, or write the history files to disk. See below under ‘‘MORE COMMANDS’’. Pressing **ESC** or **ENTER** will take you back to the main menu.

Name
: Shows the complete long filename. For a symbolic link, this command will also show the complete target of the symbolic link. This is useful in case the terminal is not wide enough to display the entire name, or if the name contains non-printable characters. Non-ASCII characters and control characters will be displayed as their octal, decimal (html entity-like) or hexadecimal equivalents like the examples in the following table.

  The ’defaultradix’ config file option specifies the radix that will initially be used. The ’defaulttranslatespace’ config file option controls whether spaces will initially be converted as well.

  When the name is shown in its converted form, pressing **N** will change the radix, and pressing **SPACE** will toggle the translation of spaces. Any other key will exit the **N** command.

  Examples:

character	representation in radix		
	octal	hexadecimal	decimal
CTRL-A	\001	\0x01	&#1;
space	\040	\0x20	&#32;
c cedilla (**ç**)	\347	\0xe7	&#231;
backslash (\)	\\	\\	\\

cOmmand
: Allows execution of a shell command. After the command completes, `pfm` will resume. If the command is `cd`, `pfm` itself will change to that directory.

  On the commandline, you may use several special abbreviations, which `pfm` will replace with the

current filename, directoryname etc. (see below). These abbreviations start with an escape character. This escape character is defined with the option 'escapechar' in your *.pfmrc* file. By default it is =. (Previous versions of `pfm` used \, but this was deemed too confusing because backslashes are parsed by the shell as well. This manual page (and the default config file) will assume you are using = as 'escapechar').

The following abbreviations are available:

=1 the current filename without extension (see below)

=2 the current filename, complete

=3 the full current directory path

=4 the mountpoint of the current filesystem

=5 the full swap directory path (see **F7** command)

=6 the basename of the current directory

=7 the extension of the current filename (see below)

=8 a space-separated list of all marked filenames. If the **=8** escape takes the form *prefix***=8***suffix*, then the *prefix* and *suffix* strings are applied to each of the names.

=9 the full previous directory path (see **F2** command)

=0 the symlink target if the current file is a symlink; otherwise an empty string

== a single literal =

=e the editor specified with the 'editor' option in the config file

=E the 'foreground' editor, specified with the 'fg_editor' option in the config file. This is expected to be defined as an editor that does not fork into the background. `pfm` uses this editor in a few cases so that it can wait for its results.

=p the pager specified with the 'pager' option in the config file

=v the image viewer specified with the 'viewer' option in the config file

The *extension* of the filename is defined as follows:

If the filename does not contain a period at all (*e.g. hosts*) or only has an initial period (*e.g. .profile*), then the file has no extension (**=7** is empty) and its whole name is regarded as **=1**.

If the filename does contain a non-initial period (*e.g. mail.log* or *.profile.old*), the extension **=7** is defined as the final part of the filename, starting at the last period in the name. The filename **=1** is the part before the last period.

In all cases, the concatenation of **=1** and **=7** is equal to **=2**.

Examples:

=2	**=1**	**=7**
track01.wav	track01	.wav
garden.jpg	garden	.jpg
end.	end	.
hosts	hosts	*empty*
.profile	.profile	*empty*
.profile.old	.profile	.old

See also below under ''ESCAPE MODIFIERS'' and ''QUOTING RULES''.

Print

Will prompt for a print command (default `lpr -P$PRINTER =2`, or `lpr =2` if PRINTER is unset) and will run it. No formatting is done. You may specify a print command with the 'printcmd' option in the *.pfmrc* file.

Quit

Exit `pfm`. The option 'confirmquit' in the *.pfmrc* file specifies whether `pfm` should ask for confirmation. Note that by pressing a capital **Q** (quick quit), you will *never* be asked for confirmation.

Rename

Change the name of the file and/or move it into another directory. You will be prompted for the new filename. Depending on your Unix implementation, a pathname on another filesystem may or may not be allowed.

In multiple-file mode, it is not allowed to specify a single non-directory filename as a new name. Instead, the new name must be a directory or a name containing a **=1** or **=2** escape (see above under c**O**mmand).

If clobber mode is off (see below under the **!** command), existing files will not be overwritten unless the action is confirmed by the user.

Show

Displays the contents of the current file or directory on screen. You can choose which pager to use for file viewing with the environment variable PAGER, or with the 'pager' option in the *.pfmrc* file.

Time

Changes mtime (modification date/time) of the file. The time may be entered either with or without clarifying interpunction (*e.g.* 2008−12−04 08:42.12). Enter **.** to set the mtime to the current date and time. If the current file does not exist in the directory (lost file or whiteout), it is *touch* (1)ed first.

User

Changes ownership of a file. Note that many Unix variants do not allow normal (non−`root`) users to change ownership. Symbolic links will be followed.

Version

Updates the current file with status information of the applicable versioning system. `pfm` will examine the current directory to figure out which versioning system is used. Supported versioning systems are: Subversion, CVS, Bazaar and Git. See also **M**ore − re**V**ision.

unWhiteout

(Only on platforms that support whiteout files). Provides the option to remove the whiteout entry in the top layer of a stacked/overlay filesystem, thereby restoring access to the corresponding file in the lower layer.

eXclude

Allows you to erase marks on a group of files which meet a certain criterion. See **I**nclude for details.

Your command

Like c**O**mmand (see above), except that it uses preconfigured commands from the *.pfmrc* file. These commands are identified by a single letter (case-sensitive) or digit.

Your command definitions may use **=0** up to **=9** and **=e**, **=E**, **=p** and **=v** escapes just as in c**O**mmand, *e.g.*:

```
your[A]:svn add =8
your[c]:tar cvf − =2 | gzip > =2.tar.gz
your[t]:tar tvf =2 | =p
your[1]:vimdiff =2*
```

Navigation through the available commands works in the same way as for directories (see above under ''NAVIGATION''), except that **ENTER** selects the highlighted command, and **j** and **k** select their own command (as do the other letters and digits).

siZe

For directories, reports the grand total (in bytes) of the directory and its contents.

For other file types, reports the total number of bytes in allocated data blocks. For regular files, this is often more than the reported file size. For special files and *fast symbolic links*, the number is zero, as

no data blocks are allocated for these file types.

If the screen layout (selected with **F9**) does not contain the 'grand total' column, then the 'filesize' column will temporarily be used instead. A 'grand total' column in the layout will never be filled in when entering the directory.

Note: since *du* (1) commands are not portable, `pfm` guesses how it can calculate the size according to the Unix variant that it runs on. If `pfm` makes an incorrect guess, please notify the author of any corrections that should be made.

MORE COMMANDS

These commands are accessible through the main screen **M**ore command.

Acl Edit the Access Control List for this file. Note: This feature has not yet been implemented for all Un*x variants.

WARNING: This feature has not been well tested on all Unices. Use it at your own risk.

Any help getting the commands right would be appreciated by the author.

Bookmark

Lists all the available bookmarks and asks the user in which slot the current directory should be bookmarked. If the input is valid, a new bookmark is created in the bookmark list.

Navigation through the bookmark list works in the same way as for directories (see above under ''NAVIGATION''), except that **ENTER** selects the highlighted slot, and **j** and **k** select their own slot (as do the other letters and digits).

Config pfm

This command will open the *.pfmrc* config file with the configured editor. The file will be re-read by `pfm` after you exit your editor. Options that are only modifiable through the config file (like 'columnlayouts') will be reinitialized immediately, options that affect settings modifiable by key commands (like 'defaultsortmode') will not.

Edit any file

You will be prompted for a filename, then your editor will be spawned.

If a capital **E** is pressed, the foreground editor is used ('fg_editor').

make Fifo

Prompts for a name, then creates a FIFO file (named pipe) with that name. See also *fifo* (4) and *mkfifo* (1).

Go to bookmark

Lists all the available bookmarks and asks the user which bookmark should be loaded to the current directory (and file). If the input is valid, the bookmark is loaded and a *chdir()* is done to the directory.

Navigation through the bookmark list works as noted above, under **Bookmark**.

sHell

Spawns your default login shell. When you exit from it, `pfm` will resume.

foLlow

If the current file is a symlink, it is followed and the cursor positioned at the target file.

Make new directory

Specify a new directory name and `pfm` will create it for you. Furthermore, if you don't have any files marked, your current directory will be set to the newly created directory.

Open window

Opens a new (file manager) window on the current directory, as configured with 'windowtype' and 'windowcmd' in the *.pfmrc*. Some examples:

If 'windowtype' is ''pfm'' and 'windowcmd' is something like ''xterm −e'', then a new terminal window running `pfm` will be opened.

If 'windowtype' is ''standalone'' and 'windowcmd' is something like ''nautilus'', then a new `nautilus` window will be opened.

Physical path
Shows the physical pathname of the current directory until a key is pressed.

Read history/bookmarks
Offers the user the choice to read the input history and/or bookmarks from disk.

The input history is the history of Unix commands, pathnames, regular expressions, modification times, and file modes that the user enters. `pfm` uses the readline library to manage these.

The input history is also read in when `pfm` starts, from individual files in *$HOME/.pfm/*.

The bookmarks are also read in when `pfm` starts, from *$HOME/.pfm/bookmarks*.

See also **M**ore−**W**rite history/bookmarks.

Show directory
You will be asked for the directory you want to view. Note that this command is different from **F7** because this will not change your current swap directory status.

alTernate screen
If the terminal has an alternate screen (like **xterm**), and `pfm` has been configured to use it (through the 'altscreenmode' option in the *.pfmrc*), then this command shows the alternate screen until a key is pressed. This is useful for reading error messages of shell commands and so on.

Version
Updates the current directory with status information of the applicable versioning system. `pfm` will examine the current directory to figure out which versioning system is used. Supported versioning systems are: Subversion, CVS, Bazaar and Git.

If you set the 'autorcs' option in your *.pfmrc*, this will automatically be done every time `pfm` shows directory contents.

Write history/bookmarks
Offers the user the choice to write the input history and/or bookmarks to disk.

The input history is also written when `pfm` exits and the 'autowritehistory' option is set in *.pfmrc*.

The bookmarks are also written when `pfm` exits and the 'autowritebookmarks' option is set in *.pfmrc*.

See also **M**ore−**R**ead history/bookmarks.

F2 Redescend
If the previous directory was a descendant of the current directory, return one level down. *e.g.* if the previous directory was */home/rene/projects/pfm* and the current directory is */home*, this command will subsequently descend into */home/rene*, */home/rene/projects* and */home/rene/projects/pfm*.

F4 Color
Like **F4**, but cycles backward.

F5 Smart refresh
Refreshes the directory like **F5** (remove lost files from the listing, read the current directory again *etc.*) but keeps the marks.

F6 Multilevel sort
Allows the user to enter a string of sort mode characters which will be applied sequentially. Example: a string of **sN** will sort by size ascending followed by name descending; the string **tn** will sort directories before files, each sorted alphabetically.

F9 Layout
Like **F9**, but cycles backward.

@ Starts a perl shell in the context of `pfm`. Primarily used for debugging.

MISCELLANEOUS and FUNCTION KEYS

ENTER

If the current file is a directory, `pfm` will *chdir()* to that directory. Otherwise, `pfm` will attempt to *launch* the file. See below under ‘‘LAUNCHING FILES’’.

DEL

Identical to the **D**elete command (see above).

! Toggle clobber mode. This controls whether a file should be overwritten when its name is reused in **C**opy, **L**ink or **R**ename.

" Toggle pathname handling. In **physical** mode, the current directory path will always be transformed to its canonical form (the simplest form, with symbolic names resolved). In **logical** mode, all symbolic link components in the current directory path will be preserved.

% Toggle show/hide whiteout files.

. Toggle show/hide dot files.

/ Identical to **F**ind (see above).

; Toggle show/hide svn ignored files.

< Pan the menu and footer, in order to view all available commands.

= Cycle through displaying identity information: username, hostname, and/or ttyname.

> Pan the menu and footer, in order to view all available commands.

? Display help. Identical to **F1**.

@ Allows the user to enter a perl command to be executed in the context of `pfm`. Primarily used for debugging, but may for example be used for setting environment variables, *e.g.*:

```
$ENV{LC_ALL} = 'C'
```

F1 Help

Display help, version number and license information.

F2 Previous

chdir() back to the previous directory. In shell commands, the previous directory path may be referred to as **=9**.

F3 Redraw

Fit the file list into the current window and refresh the display.

F4 Color

Change the current colorset. Multiple colorsets may be defined, see the *.pfmrc* file itself for details.

F5 Refresh

Current directory will be reread. Use this when the contents of the directory have changed. This command will erase all marks.

F6 Sort

Allows you to re-sort the directory listing. You will be presented a number of sort modes.

F7 Swap

Alternates the display between two directories. When switching for the first time, you are prompted for a directory path to show. When you switch back by pressing **F7** again, the contents of the alternate directory are displayed unchanged. Menu text changes color when in swap screen. In shell commands, the directory path from the alternate screen may be referred to as **=5**. If the ’persistentswap’ option has been set in the config file, then leaving the swap mode will store the main directory path as swap path again.

F8 In/Exclude

Toggles the mark (include flag) on an individual file.

F9 Layout
Toggle the column layout. Layouts are defined in your *.pfmrc*, in the 'defaultlayout' and 'columnlayouts' options. See the config file itself for information on changing the column layout.

Note that a 'grand total' column in the layout will only be filled when the si**Z**e command is issued, not when reading the directory contents.

F10 Multiple
Switch between single-file and multiple-file mode.

F11 Restat
Refresh (using *lstat* (2)) the displayed file data for the current file (or files, in multiple mode).

F12 Mouse
Toggle mouse use. See below under ''MOUSE COMMANDS''.

ESCAPE MODIFIERS

The above mentioned escapes **=0** to **=9** and **=e**, **=E**, **=p** and **=v** can make use of the following modifiers:

={escape#prefix}
={escape##prefix}
prefix is a word which may use ***** characters as a wildcard. If the *prefix* matches the beginning of the value of the escape, then the result of the expansion is the expanded value of *escape* with the shortest (**#**) or longest (**##**) matching string deleted.

={escape%suffix}
={escape%%suffix}
suffix is a word which may use ***** characters as a wildcard. If the *suffix* matches a trailing portion of the value of the escape, then the result of the expansion is the expanded value of *escape* with the shortest (**%**) or longest (**%%**) matching string deleted.

={escape/string/replacement}
={escape//string/replacement}
Find occurrences of *string* in the expanded value of *escape* and replace them with *replacement*. For **/**, replace only the first occurrence; for **//**, replace all occurrences.

={escape^letters}
={escape^^letters}
={escape,letters}
={escape,,letters}
This expansion modifies the case of alphabetic characters in *escape*. The **^** operator converts letters in *letters* to uppercase; the **,** operator converts letters in *letters* to lowercase. The **^^** and **,,** expansions convert each matched character in the expanded value; the **^** and **,** expansions match and convert only the first character in the expanded value. If *letters* is omitted, it is treated like a **?**, which matches every letter.

If a modification is done on **=8**, it is done on each of the expanded values.

Examples:

escape	**result**
=2	ActivateDebtor.php
={2}	ActivateDebtor.php
={2#Activate}	Debtor.php
={2#t}	ActivateDebtor.php
={2#*t}	ivateDebtor.php
={2##*t}	or.php
={2%hp}	ActivateDebtor.p
={2%t}	ActivateDebtor.php
={2%t*}	ActivateDeb

={2%%t*}	Ac
={2,d}	Activatedebtor.php
={2^ph}	ActivateDebtor.Php
={2^^ph}	ActivateDebtor.PHP
={2,,t}	ActivateDebtor.php
={2^^t}	AcTivaTeDebTor.php
={2,,}	activatedebtor.php
={2^^}	ACTIVATEDEBTOR.PHP
=8	UserId.php UserName.php
={8,}	userId.php userName.php
={8#User}	Id.php Name.php
=8.old	UserId.php.old UserName.php.old
={8#User}.bak	Id.php.bak Name.php.bak
Local={8/.php/}	LocalUserId LocalUserName

LAUNCHING FILES

The **ENTER** key, when used on a non-directory file, will make `pfm` attempt to launch the file.

`pfm` can be configured to use any combination of four methods for determining the appropriate command. These methods are:

extension

The filename extension will be translated to a file type (preferably a MIME type) using the 'extension[*.*extension*]' options in the config file.

Example:

```
extension[*.gif]: image/gif
extension[*.pdf]: application/pdf
```

Launch commands for every file type may be defined using the 'launch[*filetype*]' options.

Example:

```
launch[image/gif]      : =v =2 &
launch[application/pdf]: acroread =2 &
```

magic

The *file* (1) command will be run on the current file. Its output will be translated to a file type using the 'magic[*regular expression*]' options in the config file.

Example:

```
magic[GIF image data]: image/gif
magic[PDF document]  : application/pdf
```

The file type will then be used to look up a launch command as described above.

xbit

The executable bits in the file permissions will be checked (after symbolic links have been followed). If the current file is executable, `pfm` will attempt to start the file as an executable command.

name

Some filenames have their own unique way of launching them. These can be configured using the 'launchname' config option:

```
launchname[Makefile]   : make
launchname[Imakefile]  : xmkmf
launchname[Makefile.PL]: perl =2
```

To select which method or methods (*extension*, *magic*, *xbit*, and/or *name*) should be used for determining the file type, you should specify these using the 'launchby' option (separated by commas if there is more than one).

Example:

```
launchby:name,xbit,extension
```

`pfm` will try these methods in succession until one succeeds, or all fail.

If the file type cannot be determined, the current file will be displayed using your pager.

The **ENTER** key will always behave as if `pfm` runs in single-file mode. It will *not* launch multiple files. Use **Y**our or c**O**mmand to launch multiple files.

QUOTING RULES

`pfm` adds an extra layer of parsing to filenames and shell commands. It is therefore important to take notice of the rules that `pfm` uses.

In versions prior to 1.93.1, the default escape character was \. Since this causes confusing results, this is no longer the default, and you are discouraged from using it.

The following six types of input can be distinguished:

a regular expression (only the **I**nclude and e**X**clude commands)
The input is parsed as a regular expression.

a time (e.g. the **T**ime or **I**nclude − **B**efore commands)
Characters not in the set `[0-9.]` are removed from the input.

a literal pattern (only the **F**ind command)
The input is taken literally.

not a filename or shell command (e.g. in **A**ttribute or **U**ser)
The input is taken literally.

a filename (e.g. in **C**opy or tar**G**et).
First of all, tilde expansion is performed.

Next, any `=[0-9eEpv]` character sequence is expanded to the corresponding value.

At the same time, any `=[^0-9eEpv]` character sequence is just replaced with the character itself.

Finally, if the filename is to be processed by `pfm`, it is taken literally; if it is to be handed over to a shell, all metacharacters are replaced *escaped*.

a shell command (e.g. in c**O**mmand or **P**rint)
First of all, tilde expansion is performed.

Next, any `=[0-9eEpv]` character sequence is expanded to the corresponding value, *with shell metacharacters escaped*.

At the same time, any `=[^0-9eEpv]` character sequence is just replaced with the character itself.

In short:

- `pfm` always escapes shell metacharacters in expanded `=2` *etc.* constructs.
- In filenames entered, shell metacharacters are taken literally.
- In shell commands entered, metacharacters that you want to be taken literally must be escaped one extra time.

Examples:

char(s) wanted in filename	char(s) to type in filename	char(s) to type in shell command
any non-metachar	*that char*	*that char*
\	\	\\ **or** '\'
"	"	\" **or** '"'
=	==	==

space	*space*	*space* **or** '*space*'
filename	=2	=2
\2	\2	\\2 **or** '\2'
=2	==2	==2

MOUSE COMMANDS

When `pfm` is run in an xterm or other terminal (or emulator) that supports the use of a mouse, turning on mouse mode (either initially with the 'defaultmousemode' option in the *.pfmrc* file, or while running using the **F12** key) will give mouse access to the following commands:

btn	location clicked					
	pathline	menu/ footer	heading	file-line	file-name	dirname
1	*chdir()*	*pfm-cmd*	sort	F8	**S**how	**S**how
2	c**O**mmand	*pfm-cmd*	sort rev	**S**how	ENTER	**M**ore - **O**pen win
3	c**O**mmand	*pfm-cmd*	sort rev	**S**how	ENTER	**M**ore - **O**pen win
up	*five lines up*					
down	*five lines down*					

If the config option 'mouse_moves_cursor' has been set to 'yes', a mouse click on a non-directory will move the cursor to that file. Otherwise, the cursor will *only* move when a directory is clicked or the mouse wheel is used.

The mouse wheel moves the cursor five lines per notch by default, or one line if shift is pressed. The actual number of lines can be configured in your *.pfmrc* using 'mousewheeljumpsize'.

Clicking button 1 on the current directory path will *chdir()* up to the clicked ancestor directory. If the current directory was clicked, or the device name, it will act like a **M**ore − **S**how command.

Clicking button 2 on a directory name will open a new window like **M**ore − **O**pen window.

Clicking on the column headings will sort the directory contents by that heading. Clicking again will sort the directory in reverse order.

Clicking on ''Sort'' in the footer will cycle through a number of preconfigured sort modes as defined in the config option 'sortcycle'.

Clicking on the menu or footer will execute the command that was clicked.

Clicking on one of the identity lines in the info column will toggle the display like the **=** command does.

Mouse use will be turned off during the execution of commands.

WORKING DIRECTORY INHERITANCE

Upon exit, `pfm` will save its current working directory in the file *$HOME/.pfm/cwd*, and its swap directory, if any, in *$HOME/.pfm/swd*. This enables the user to have the calling process (shell) ''inherit'' `pfm`'s current working directory, and to reinstate the swap directory upon the next invocation. To achieve this, you may call `pfm` using a function or alias like the following:

Example for *ksh* (1), *bash* (1) and *zsh* (1):

```
pfm() {
    if [ -s ~/.pfm/swd ]; then
        swd=-s"`cat ~/.pfm/swd`"
    fi
    # providing $swd is optional
    env pfm $swd "$@"
    if [ -s ~/.pfm/cwd ]; then
        cd "`cat ~/.pfm/cwd`"
        rm -f ~/.pfm/cwd
    fi
}
```

Example for *csh* (1) and *tcsh* (1):

```
alias pfm ':                                \
if (-s ~/.pfm/swd) then                     \
    set swd=-s"`cat ~/.pfm/swd`"            \
endif                                       \
: providing $swd is optional                \
env pfm $swd \!*                            \
if (-s ~/.pfm/cwd) then                     \
    cd "`cat ~/.pfm/cwd`"                   \
    rm -f ~/.pfm/cwd                        \
endif'
```

ENVIRONMENT

ANSI_COLORS_DISABLED

Detected as an indication that ANSI coloring escape sequences should not be used.

CDPATH

A colon-separated list of directories specifying the search path when changing directories. There is always an implicit **.** entry at the start of this search path.

DISPLAY

The X display on which the 'windowcmd' will be opened.

EDITOR

The editor to be used for the **E**dit command. Overridden by VISUAL.

LC_ALL
LC_COLLATE
LC_CTYPE
LC_MESSAGES
LC_NUMERIC
LC_TIME
LANG

Determine locale settings, most notably for collation sequence, messages and date/time format. See *locale* (7).

PAGER

Identifies the pager with which to view text files. Defaults to *less* (1) for Linux systems or *more* (1) for Unix systems.

PERL_RL

Indicate whether and how the readline prompts should be highlighted. See *Term::ReadLine* (3pm). If unset, a good guess is made based on your config file 'framecolors[]' setting.

PFMRC

Specify a location of an alternate *.pfmrc* file. If unset, the default location *$HOME/.pfm/.pfmrc* is used. The cwd- and history-files cannot be displaced in this manner, and will always be located in the

directory *$HOME/.pfm/*.

PRINTER
May be used to specify a printer to print to using the **P**rint command.

SHELL
Your default login shell, spawned by **M**ore − s**H**ell.

VISUAL
The editor to be used for the **E**dit command. Overrides EDITOR.

FILES

The directory *$HOME/.pfm/* and files therein. A number of input histories and the current working directory on exit are saved to this directory.

The default location for the config file is *$HOME/.pfm/.pfmrc*.

EXIT STATUS

0 Success (could also be a user requested exit, *e.g.* after **−−help** or **−−version**).

1 Invalid commandline option.

2 No valid layout found in the *.pfmrc* file.

DIAGNOSIS

If `pfm` reports that your config file might be outdated, you might be missing some of the newer configuration options (or default values for these). The best way to solve this is by having `pfm` generate a new config file, *e.g.* by running the following command and comparing the new config file with your original one:

```
env PFMRC=~/.pfm/.pfmrc-new pfm
```

Alternatively, you may reply with **y** when `pfm` offers to update the config file for you.

If a function key like **F1** or **F10** is intercepted by your windowing system or OS, and it is not feasible to change that mapping, then you can define an alternative key mapping (like Shift−**F1**, Shift−**F10**) in your *.pfmrc*, *e.g.*:

```
keydef[*]:k1=\eO1;2P:k10=\e[21;2~:
```

Or for a specific value of **TERM**:

```
keydef[xterm]:k1=\eO1;2P:k10=\e[21;2~:
```

BUGS and WARNINGS

The smallest terminal size supported is 80x24. The display will be messed up if you resize your terminal window to a smaller size, unless you specify 'force_minimum_size' in the config file and the terminal supports resizing; in that case `pfm` will resize the terminal to at least 80 columns and/or 24 rows.

The author once almost pressed **ENTER** when logged in as root and with the cursor on the file */sbin/reboot*. You have been warned.

VERSION

This manual pertains to `pfm` version 2.12.1.

AUTHOR and COPYRIGHT

This program was based on PFM.COM version 2.32, originally written for MS-DOS by Paul R. Culley and Henk de Heer. The name 'pfm' was adopted with kind permission of the original authors.

Special thanks to Ewoud Kappers for useful suggestions and testing.

Special thanks to Maurice Makaay for useful suggestions and assistance with debugging.

SEE ALSO

The documentation on PFM.COM. The manual pages for *chmod* (1), *cp* (1), *file* (1), *less* (1), *ln* (1), *locale* (7), *lpr* (1), *touch* (1), *vi* (1).

For developers:

Term::Screen (3pm), *Term::ScreenColor* (3pm), *Term::ReadLine* (3pm).

App::PFM::Abstract (3pm), *App::PFM::Application* (3pm), *App::PFM::Browser* (3pm), *App::PFM::CommandHandler* (3pm), *App::PFM::Config* (3pm), *App::PFM::Directory* (3pm), *App::PFM::Event* (3pm), *App::PFM::File* (3pm), *App::PFM::History* (3pm), *App::PFM::JobHandler* (3pm), *App::PFM::Job::Abstract* (3pm), *App::PFM::OS* (3pm), *App::PFM::Screen* (3pm), *App::PFM::State* (3pm) and *App::PFM::Util* (3pm).

The pfm project page: <http://sourceforge.net/projects/p−f−m/>

The pfm homepage: <http://p−f−m.sourceforge.net/>