-- --

IEEE P1003.2 Draft 11.2 – September 1991

B.6 C Binding for Match Filename or Pathname

Function: *fnmatch* ()

B.6.1 Synopsis

```
#include <fnmatch.h>
int fnmatch(const char *pattern, const char *string, int flags);
```

B.6.2 Description

The *fnmatch* () function shall match patterns as described in 3.13.1 and 3.13.2. It checks the string specified by the *string* argument to see if it matches the pattern specified by the *pattern* argument.

The *flags* argument modifies the interpretation of *pattern* and *string*. It is the bitwise inclusive OR of zero or more of the flags shown in Table B-11, which are defined in the header `<fnmatch.h>`. If the FNM_PATHNAME flag is set in *flags*, then a slash character in *string* shall be explicitly matched by a slash in *pattern*; it shall not be matched by either the asterisk or question-mark special characters, nor by a bracket expression. If the FNM_PATHNAME flag is not set, the slash character shall be treated as an ordinary character.

Table B-11 – *fnmatch* () *flags* Argument

flags	Description	
FNM_NOESCAPE	Disable backslash escaping	1
FNM_PATHNAME	Slash in *string* only matches slash in *pattern*	
FNM_PERIOD	Leading period in *string* must be exactly matched by period in *pattern*	

If FNM_NOESCAPE is not set in *flags*, a backslash character (\) in *pattern* fol- 1
lowed by any other character shall match that second character in *string*. In par-
ticular, `'\\'` shall match a backslash in *string*. If FNM_NOESCAPE is set, a 1
backslash character shall be treated as an ordinary character.

If FNM_PERIOD is set in *flags*, then a leading period in *string* shall match a period
in *pattern* as described by rule (2) in 3.13.2, where the location of ‘‘leading’’ is indi- 1
cated by the value of FNM_PATHNAME: 1

- If FNM_PATHNAME is set, a period is ‘‘leading’’ if it is the first character in *string* or if it immediately follows a slash.
- If FNM_PATHNAME is not set, a period is ‘‘leading’’ only if it is the first character of *string*.

If FNM_PERIOD is not set, then no special restrictions shall be placed on matching a period.

B.6.3 Returns

If *string* matches the pattern specified by *pattern*, then *fnmatch*() shall return zero. If there is no match, *fnmatch*() shall return FNM_NOMATCH, which shall be defined in the header <fnmatch.h>. If an error occurs, *fnmatch*() shall return another nonzero value.

B.6.4 Errors

This standard does not specify any error conditions that are required to be detected by the *fnmatch*() function. Some errors may be detected under unspecified conditions.

B.6.5 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

The *fnmatch*() function has two major uses. It could be used by an application or utility that needs to read a directory and apply a pattern against each entry. The `find` utility is an example of this. It can also be used by the `pax` utility to process its *pattern* operands, or by applications that need to match strings in a similar manner.

History of Decisions Made

This function replaces the REG_FILENAME flag of *regcomp*() in early drafts. It provides virtually the same functionality as the *regcomp*() and *regexec*() functions using the REG_FILENAME and REG_FSLASH flags [the REG_FSLASH flag was proposed for *regcomp*(), and would have had the opposite effect from FMN_PATHNAME], but with a simpler interface and less overhead.

The name *fnmatch*() is intended to imply *filename* match, rather than *pathname* match. The default action of this function is to match filenames, rather than pathnames, since it gives no special significance to the slash character. With the FNM_PATHNAME flag, *fnmatch*() does match pathnames, but without tilde expansion, parameter expansion, or special treatment for period at the beginning of a filename.

B.7 C Binding for Command Option Parsing

Function: *getopt* ()

B.7.1 Synopsis

```
#include <unistd.h>
int getopt(int argc, char * const argv[], const char *optstring);        1
extern char *optarg;
extern int optind, opterr, optopt;
```

B.7.2 Description

The *getopt* () function is a command-line parser that can be used by applications that follow Utility Syntax Guidelines 3, 4, 5, 6, 7, 9, and 10 in 2.10.2. The remaining guidelines are not addressed by *getopt* () and are the responsibility of the application.

The parameters *argc* and *argv* are the argument count and argument array as passed to *main* (). The argument *optstring* is a string of recognized option characters; if a character is followed by a colon, the option takes an argument. All option characters allowed by Utility Syntax Guideline 3 are allowed in *optstring* . The implementation may accept other characters as an extension.

The variable *optind* is the index of the next element of the *argv* [] vector to be processed. It is initialized to 1 by the system, and *getopt* () updates it when it finishes with each element of *argv* []. When an element of *argv* [] contains multiple option characters, it is unspecified how *getopt* () determines which options have already been processed.

The *getopt* () function shall return the next option character from *argv* that matches a character in *optstring* , if there is one that matches. If the option takes 1
an argument, *getopt* () shall set the variable *optarg* to point to the option-argument as follows:

(1) If the option was the last character in the string pointed to by an element of *argv* , then *optarg* contains the next element of *argv* , and *optind* shall be incremented by 2. If the resulting value of *optind* is not less than *argc* , this indicates a missing option argument, and *getopt* () shall return an error indication.

(2) Otherwise, *optarg* points to the string following the option character in that element of *argv* , and *optind* shall be incremented by 1.

If, when *getopt* () is called, *argv* [*optind*] is **NULL** , **argv* [*optind*] is not the character −, or *argv* [*optind*] points to the string `"-"`, *getopt* () shall return −1 without changing *optind* . If *argv* [*optind*] points to the string `"--"`, *getopt* () shall return −1 after incrementing *optind* .