## PROLOG

This manual page is part of the POSIX Programmer's Manual. The Cygwin implementation of this interface may differ (consult the corresponding Cygwin manual page for details of Cygwin behavior), or the interface may not be implemented on Cygwin.

## NAME

regex.h — regular expression matching types

## SYNOPSIS

#include <regex.h>

## DESCRIPTION

The *<regex.h>* header shall define the structures and symbolic constants used by the *regcomp*(), *regexec*(), *regerror*(), and *regfree*() functions.

The *<regex.h>* header shall define the **regex_t** structure type, which shall include at least the following member:

**size_t** **re_nsub** Number of parenthesized subexpressions.

The *<regex.h>* header shall define the **size_t** type as described in *<sys/types.h>*.

The *<regex.h>* header shall define the **regoff_t** type as a signed integer type that can hold the largest value that can be stored in either a **ptrdiff_t** type or a **ssize_t** type.

The *<regex.h>* header shall define the **regmatch_t** structure type, which shall include at least the following members:

**regoff_t** **rm_so** Byte offset from start of string to start of substring.

regoff_t rm_eo Byte offset from start of string of the first character after the end of substring.

The *<regex.h>* header shall define the following symbolic constants for the *cflags* parameter to the *regcomp*() function:

REG_EXTENDED
: Use Extended Regular Expressions.

REG_ICASE
: Ignore case in match.

REG_NOSUB
: Report only success or fail in *regexec*().

REG_NEWLINE
: Change the handling of <newline>.

The *<regex.h>* header shall define the following symbolic constants for the *eflags* parameter to the *regexec*() function:

REG_NOTBOL
: The <circumflex> character (**'^'**), when taken as a special character, does not match the beginning of *string*.

REG_NOTEOL
: The <dollar-sign> (**'$'**), when taken as a special character, does not match the end of *string*.

The *<regex.h>* header shall define the following symbolic constants as error return values:

REG_NOMATCH
: *regexec*() failed to match.

REG_BADPAT
: Invalid regular expression.

REG_ECOLLATE
: Invalid collating element referenced.

REG_ECTYPE
: Invalid character class type referenced.

REG_EESCAPE
: Trailing <backslash> character in pattern.

REG_ESUBREG
: Number in *\digit* invalid or in error.

REG_EBRACK
: **"[]"** imbalance.

REG_EPAREN
: **"\(\)"** or **"()"** imbalance.

REG_EBRACE
: **"\{\}"** imbalance.

REG_BADBR
: Content of **"\{\}"** invalid: not a number, number too large, more than two numbers, first larger than second.

REG_ERANGE
: Invalid endpoint in range expression.

REG_ESPACE
: Out of memory.

REG_BADRPT
: **'?'**, **'*'**, or **'+'** not preceded by valid regular expression.

The following shall be declared as functions and may also be defined as macros. Function prototypes shall be provided.

```
int    regcomp(regex_t *restrict, const char *restrict, int);
size_t regerror(int, const regex_t *restrict, char *restrict, size_t);
int    regexec(const regex_t *restrict, const char *restrict, size_t,
         regmatch_t [restrict], int);
void   regfree(regex_t *);
```

The implementation may define additional macros or constants using names beginning with REG_.

*The following sections are informative.*

## APPLICATION USAGE

None.

## RATIONALE

None.

## FUTURE DIRECTIONS

None.

## SEE ALSO

**<sys_types.h>**

The System Interfaces volume of POSIX.1-2008, *regcomp*( )

## COPYRIGHT

https://www.kernel.org/doc/man-pages/reporting_bugs.html .