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>

XSH regcomp

CHANGE HISTORY

First released in Issue 4.

Originally derived from the ISO POSIX-2 standard.

Issue 6

The REG_ENOSYS constant is marked obsolescent.

The **restrict** keyword is added to the prototypes for *regcomp()*, *regerror()*, and *regexec()*.

A statement is added that the **size_t** type is defined as described in *<sys/types.h>*.

Issue 7

SD5-XBD-ERN-60 is applied.

The obsolescent REG_ENOSYS constant is removed.

This reference page is clarified with respect to macros and symbolic constants.

End of informative text.

---

return to top of page

[Main Index | XBD | XSH | XCU | XRAT]

<<< Previous Home Next >>>