

**NAME**

PCRE - Perl-compatible regular expressions

**SYNOPSIS**

```
#include <pcre.h>
```

```
pcre *pcre_compile(const char *pattern, int options,
    const char **errptr, int *erroffset,
    const unsigned char *tableptr);
```

```
pcre16 *pcre16_compile(PCRE_SPTR16 pattern, int options,
    const char **errptr, int *erroffset,
    const unsigned char *tableptr);
```

```
pcre32 *pcre32_compile(PCRE_SPTR32 pattern, int options,
    const char **errptr, int *erroffset,
    const unsigned char *tableptr);
```

**DESCRIPTION**

This function compiles a regular expression into an internal form. It is the same as **pcre[16|32]\_compile2()**, except for the absence of the *errorcodeptr* argument. Its arguments are:

*pattern*     A zero-terminated string containing the  
              regular expression to be compiled  
*options*     Zero or more option bits  
*errptr*      Where to put an error message  
*erroffset*   Offset in pattern where error was found  
*tableptr*    Pointer to character tables, or NULL to  
              use the built-in default

The option bits are:

PCRE\_ANCHORED     Force pattern anchoring  
PCRE\_AUTO\_CALLOUT   Compile automatic callouts  
PCRE\_BSR\_ANYCRLF   \R matches only CR, LF, or CRLF  
PCRE\_BSR\_UNICODE   \R matches all Unicode line endings  
PCRE\_CASELESS      Do caseless matching  
PCRE\_DOLLAR\_ENDONLY \$ not to match newline at end  
PCRE\_DOTALL        . matches anything including NL  
PCRE\_DUPNAMES      Allow duplicate names for subpatterns  
PCRE\_EXTENDED      Ignore white space and # comments  
PCRE\_EXTRA         PCRE extra features  
                  (not much use currently)  
PCRE\_FIRSTLINE     Force matching to be before newline  
PCRE\_JAVASCRIPT\_COMPAT JavaScript compatibility  
PCRE\_MULTILINE     ^ and \$ match newlines within data  
PCRE\_NEVER\_UTF      Lock out UTF, e.g. via (\*UTF)  
PCRE\_NEWLINE\_ANY    Recognize any Unicode newline sequence  
PCRE\_NEWLINE\_ANYCRLF Recognize CR, LF, and CRLF as newline  
                  sequences  
PCRE\_NEWLINE\_CR     Set CR as the newline sequence  
PCRE\_NEWLINE\_CRLF   Set CRLF as the newline sequence  
PCRE\_NEWLINE\_LF     Set LF as the newline sequence  
PCRE\_NO\_AUTO\_CAPTURE Disable numbered capturing paren-

theses (named ones available)

PCRE\_NO\_AUTO\_POSSESS Disable auto-possessification

PCRE\_NO\_START\_OPTIMIZE Disable match-time start optimizations

PCRE\_NO\_UTF16\_CHECK Do not check the pattern for UTF-16 validity (only relevant if PCRE\_UTF16 is set)

PCRE\_NO\_UTF32\_CHECK Do not check the pattern for UTF-32 validity (only relevant if PCRE\_UTF32 is set)

PCRE\_NO\_UTF8\_CHECK Do not check the pattern for UTF-8 validity (only relevant if PCRE\_UTF8 is set)

PCRE\_UCP Use Unicode properties for \d, \w, etc.

PCRE\_UNGREEDY Invert greediness of quantifiers

PCRE\_UTF16 Run in **pcr e16\_compile()** UTF-16 mode

PCRE\_UTF32 Run in **pcr e32\_compile()** UTF-32 mode

PCRE\_UTF8 Run in **pcr e\_compile()** UTF-8 mode

PCRE must be built with UTF support in order to use PCRE\_UTF8/16/32 and PCRE\_NO\_UTF8/16/32\_CHECK, and with UCP support if PCRE\_UCP is used.

The yield of the function is a pointer to a private data structure that contains the compiled pattern, or NULL if an error was detected. Note that compiling regular expressions with one version of PCRE for use with a different version is not guaranteed to work and may cause crashes.

There is a complete description of the PCRE native API in the **pcreapi** page and a description of the POSIX API in the **pcreposix** page.