NAME

PCRE - Perl-compatible regular expressions

SYNOPSIS

#include <pcre.h>

void pcre_assign_jit_stack(pcre_extra ****extra*,**
pcre_jit_callback *callback***, void ****data***);**

void pcre16_assign_jit_stack(pcre16_extra ****extra*,**
pcre16_jit_callback *callback***, void ****data***);**

void pcre32_assign_jit_stack(pcre32_extra ****extra*,**
pcre32_jit_callback *callback***, void ****data***);**

DESCRIPTION

This function provides control over the memory used as a stack at run-time by a call to **pcre[16|32]_exec()** with a pattern that has been successfully compiled with JIT optimization. The arguments are:

extra	the data pointer returned by**pcr e[16|32]_study()**
callback	a callback function
data	a JIT stack or a value to be passed to the callback function

If *callback* is NULL and *data* is NULL, an internal 32K block on the machine stack is used.

If *callback* is NULL and *data* is not NULL, *data* must be a valid JIT stack, the result of calling **pcre[16|32]_jit_stack_alloc()**.

If *callback* not NULL, it is called with *data* as an argument at the start of matching, in order to set up a JIT stack. If the result is NULL, the internal 32K stack is used; otherwise the return value must be a valid JIT stack, the result of calling **pcre[16|32]_jit_stack_alloc()**.

You may safely assign the same JIT stack to multiple patterns, as long as they are all matched in the same thread. In a multithread application, each thread must use its own JIT stack. For more details, see the **pcre-jit** page.

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.