Rocky Enterprise Linux 9.2 Manual Pages on command 'pthread_getattr_default_np.3'

$ man pthread_getattr_default_np.3

PTHREAD_GETATTR_DEFAULT_NP(Linux Programmer's ManPTHREAD_GETATTR_DEFAULT_NP(3)

NAME

pthread_getattr_default_np, pthread_setattr_default_np, - get or set

default thread-creation attributes

SYNOPSIS

#define _GNU_SOURCE /* See feature_test_macros(7) */

#include <pthread.h>

int pthread_getattr_default_np(pthread_attr_t *attr);

int pthread_setattr_default_np(pthread_attr_t *attr);

Compile and link with -pthread.

DESCRIPTION

The pthread_setattr_default_np() function sets the default attributes

used for creation of a new thread?that is, the attributes that are used

when pthread_create(3) is called with a second argument that is NULL.

The default attributes are set using the attributes supplied in *attr,

a previously initialized thread attributes object. Note the following

details about the supplied attributes object:

* The attribute settings in the object must be valid.

* The stack address attribute must not be set in the object.

* Setting the stack size attribute to zero means leave the default stack size unchanged.

The pthread_getattr_default_np() function initializes the thread at? tributes object referred to by attr so that it contains the default at? tributes used for thread creation.

ERRORS

EINVAL (pthread_setattr_default_np()) One of the attribute settings in attr is invalid, or the stack address attribute is set in attr.

ENOMEM (pthread_setattr_default_np()) Insufficient memory.

VERSIONS

These functions are available in glibc since version 2.18.

ATTRIBUTES

For an explanation of the terms used in this section, see at? tributes(7).

Interface	Attribute	Value
pthread_getattr_default_np(), pthread_setattr_default_np()	Thread safety	MT-Safe

CONFORMING TO

These functions are nonstandard GNU extensions; hence the suffix "_np" (nonportable) in their names.

EXAMPLES

The program below uses pthread_getattr_default_np() to fetch the de? fault thread-creation attributes and then displays various settings from the returned thread attributes object. When running the program, we see the following output:

```
$ ./a.out
Stack size:          8388608
Guard size:          4096
Scheduling policy:   SCHED_OTHER
```

```
        Scheduling priority: 0
        Detach state:        JOINABLE
        Inherit scheduler:   INHERIT
```

Program source

```
    #define _GNU_SOURCE
    #include <pthread.h>
    #include <stdio.h>
    #include <stdlib.h>
    #include <errno.h>
    #define errExitEN(en, msg) \
                            do { errno = en; perror(msg); \
                                exit(EXIT_FAILURE); } while (0)
    static void
    display_pthread_attr(pthread_attr_t *attr)
    {
        int s;
        size_t stacksize;
        size_t guardsize;
        int policy;
        struct sched_param schedparam;
        int detachstate;
        int inheritsched;
        s = pthread_attr_getstacksize(attr, &stacksize);
        if (s != 0)
            errExitEN(s, "pthread_attr_getstacksize");
        printf("Stack size:          %zd\n", stacksize);
        s = pthread_attr_getguardsize(attr, &guardsize);
        if (s != 0)
            errExitEN(s, "pthread_attr_getguardsize");
        printf("Guard size:          %zd\n", guardsize);
        s = pthread_attr_getschedpolicy(attr, &policy);
        if (s != 0)
            errExitEN(s, "pthread_attr_getschedpolicy");
```

```
    printf("Scheduling policy:   %s\n",
           (policy == SCHED_FIFO) ? "SCHED_FIFO" :
           (policy == SCHED_RR) ? "SCHED_RR" :
           (policy == SCHED_OTHER) ? "SCHED_OTHER" : "[unknown]");
    s = pthread_attr_getschedparam(attr, &schedparam);
    if (s != 0)
        errExitEN(s, "pthread_attr_getschedparam");
    printf("Scheduling priority: %d\n", schedparam.sched_priority);
    s = pthread_attr_getdetachstate(attr, &detachstate);
    if (s != 0)
        errExitEN(s, "pthread_attr_getdetachstate");
    printf("Detach state:        %s\n",
           (detachstate == PTHREAD_CREATE_DETACHED) ? "DETACHED" :
           (detachstate == PTHREAD_CREATE_JOINABLE) ? "JOINABLE" :
           "???");
    s = pthread_attr_getinheritsched(attr, &inheritsched);
    if (s != 0)
        errExitEN(s, "pthread_attr_getinheritsched");
    printf("Inherit scheduler:   %s\n",
           (inheritsched == PTHREAD_INHERIT_SCHED) ? "INHERIT" :
           (inheritsched == PTHREAD_EXPLICIT_SCHED) ? "EXPLICIT" :
           "???");
}
int
main(int argc, char *argv[])
{
    int s;
    pthread_attr_t attr;
    s = pthread_getattr_default_np(&attr);
    if (s != 0)
        errExitEN(s, "pthread_getattr_default_np");
    display_pthread_attr(&attr);
    exit(EXIT_SUCCESS);
```

```
}
```

SEE ALSO

pthread_attr_getaffinity_np(3), pthread_attr_getdetachstate(3), pthread_attr_getguardsize(3), pthread_attr_getinheritsched(3), pthread_attr_getschedparam(3), pthread_attr_getschedpolicy(3), pthread_attr_getscope(3), pthread_attr_getstack(3), pthread_attr_getstackaddr(3), pthread_attr_getstacksize(3), pthread_attr_init(3), pthread_create(3), pthreads(7)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.