

**NAME**

**pthread\_attr\_init**, **pthread\_attr\_destroy**, **pthread\_attr\_setstack**, **pthread\_attr\_getstack**,  
**pthread\_attr\_setstacksize**, **pthread\_attr\_getstacksize**, **pthread\_attr\_setguardsize**,  
**pthread\_attr\_getguardsize**, **pthread\_attr\_setstackaddr**, **pthread\_attr\_getstackaddr**,  
**pthread\_attr\_setdetachstate**, **pthread\_attr\_getdetachstate**, **pthread\_attr\_setinheritsched**,  
**pthread\_attr\_getinheritsched**, **pthread\_attr\_setschedparam**, **pthread\_attr\_getschedparam**,  
**pthread\_attr\_setschedpolicy**, **pthread\_attr\_getschedpolicy**, **pthread\_attr\_setscope**, **pthread\_attr\_getscope**  
- thread attribute operations

**LIBRARY**

POSIX Threads Library (libpthread, -lpthread)

**SYNOPSIS**

```
#include <pthread.h>
```

*int*

```
pthread_attr_init(pthread_attr_t *attr);
```

*int*

```
pthread_attr_destroy(pthread_attr_t *attr);
```

*int*

```
pthread_attr_setstack(pthread_attr_t *attr, void *stackaddr, size_t stacksize);
```

*int*

```
pthread_attr_getstack(const pthread_attr_t * restrict attr, void ** restrict stackaddr,  
size_t * restrict stacksize);
```

*int*

```
pthread_attr_setstacksize(pthread_attr_t *attr, size_t stacksize);
```

*int*

```
pthread_attr_getstacksize(const pthread_attr_t * restrict attr, size_t * restrict stacksize);
```

*int*

```
pthread_attr_setguardsize(pthread_attr_t *attr, size_t guardsize);
```

*int*

```
pthread_attr_getguardsize(const pthread_attr_t * restrict attr, size_t * restrict guardsize);
```

*int*

**pthread\_attr\_setstackaddr**(*pthread\_attr\_t \*attr, void \*stackaddr*);

*int*

**pthread\_attr\_getstackaddr**(*const pthread\_attr\_t \*attr, void \*\*stackaddr*);

*int*

**pthread\_attr\_setdetachstate**(*pthread\_attr\_t \*attr, int detachstate*);

*int*

**pthread\_attr\_getdetachstate**(*const pthread\_attr\_t \*attr, int \*detachstate*);

*int*

**pthread\_attr\_setinheritsched**(*pthread\_attr\_t \*attr, int inheritsched*);

*int*

**pthread\_attr\_getinheritsched**(*const pthread\_attr\_t \*restrict attr, int \*restrict inheritsched*);

*int*

**pthread\_attr\_setschedparam**(*pthread\_attr\_t \*attr, const struct sched\_param \*param*);

*int*

**pthread\_attr\_getschedparam**(*const pthread\_attr\_t \*attr, struct sched\_param \*param*);

*int*

**pthread\_attr\_setschedpolicy**(*pthread\_attr\_t \*attr, int policy*);

*int*

**pthread\_attr\_getschedpolicy**(*const pthread\_attr\_t \*restrict attr, int \*restrict policy*);

*int*

**pthread\_attr\_setscope**(*pthread\_attr\_t \*attr, int contentionscope*);

*int*

**pthread\_attr\_getscope**(*const pthread\_attr\_t \*restrict attr, int \*restrict contentionscope*);

## DESCRIPTION

Thread attributes are used to specify parameters to **pthread\_create()**. One attribute object can be used in multiple calls to **pthread\_create()**, with or without modifications between calls.

The **pthread\_attr\_init()** function initializes *attr* with all the default thread attributes.

The **pthread\_attr\_destroy()** function destroys *attr*.

The **pthread\_attr\_set\*()** functions set the attribute that corresponds to each function name.

The **pthread\_attr\_get\*()** functions copy the value of the attribute that corresponds to each function name to the location pointed to by the second function parameter.

## RETURN VALUES

If successful, these functions return 0. Otherwise, an error number is returned to indicate the error.

## ERRORS

The **pthread\_attr\_init()** function will fail if:

[ENOMEM]            Out of memory.

The **pthread\_attr\_destroy()** function will fail if:

[EINVAL]            Invalid value for *attr*.

The **pthread\_attr\_setstacksize()** and **pthread\_attr\_setstack()** functions will fail if:

[EINVAL]            *stacksize* is less than PTHREAD\_STACK\_MIN.

The **pthread\_attr\_setdetachstate()** function will fail if:

[EINVAL]            Invalid value for *detachstate*.

The **pthread\_attr\_setinheritsched()** function will fail if:

[EINVAL]            Invalid value for *attr*.

The **pthread\_attr\_setschedparam()** function will fail if:

[EINVAL]            Invalid value for *attr*.

[ENOTSUP]           Invalid value for *param*.

The **pthread\_attr\_setschedpolicy()** function will fail if:

[EINVAL] Invalid value for *attr*.

[ENOTSUP] Invalid or unsupported value for *policy*.

The **pthread\_attr\_setscope()** function will fail if:

[EINVAL] Invalid value for *attr*.

[ENOTSUP] Invalid or unsupported value for *contentionscope*.

#### SEE ALSO

`pthread_attr_affinity_np(3)`, `pthread_attr_get_np(3)`, `pthread_create(3)`

#### STANDARDS

**pthread\_attr\_init()**, **pthread\_attr\_destroy()**, **pthread\_attr\_setstacksize()**, **pthread\_attr\_getstacksize()**, **pthread\_attr\_setstackaddr()**, **pthread\_attr\_getstackaddr()**, **pthread\_attr\_setdetachstate()**, and **pthread\_attr\_getdetachstate()** functions conform to ISO/IEC 9945-1:1996 ("POSIX.1")

The **pthread\_attr\_setinheritsched()**, **pthread\_attr\_getinheritsched()**, **pthread\_attr\_setschedparam()**, **pthread\_attr\_getschedparam()**, **pthread\_attr\_setschedpolicy()**, **pthread\_attr\_getschedpolicy()**, **pthread\_attr\_setscope()**, and **pthread\_attr\_getscope()** functions conform to Version 2 of the Single UNIX Specification ("SUSv2")