NAME

pthread_attr_init, pthread_attr_destroy, pthread_attr_setdetachstate, pthread_attr_getdetachstate, pthread_attr_setschedparam, pthread_attr_getschedparam, pthread_attr_setschedpolicy, pthread_attr_getschedpolicy, pthread_attr_setinheritsched, pthread_attr_getinheritsched, pthread_attr_setscope, pthread_attr_getscope − thread creation attributes

SYNOPSIS

#include <pthread.h>

int pthread_attr_init(pthread_attr_t ***attr*);**

int pthread_attr_destroy(pthread_attr_t ***attr*);**

int pthread_attr_setdetachstate(pthread_attr_t ***attr*, int** *detachstate***);**

int pthread_attr_getdetachstate(const pthread_attr_t ***attr*, int ***detachstate***);**

int pthread_attr_setschedpolicy(pthread_attr_t ***attr*, int** *policy***);**

int pthread_attr_getschedpolicy(const pthread_attr_t ***attr*, int ***policy***);**

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_setinheritsched(pthread_attr_t ***attr*, int** *inherit***);**

int pthread_attr_getinheritsched(const pthread_attr_t ***attr*, int ***inherit***);**

int pthread_attr_setscope(pthread_attr_t ***attr*, int** *scope***);**

int pthread_attr_getscope(const pthread_attr_t ***attr*, int ***scope***);**

DESCRIPTION

Setting attributes for threads is achieved by filling a thread attribute object *attr* of type **pthread_attr_t**, then passing it as second argument to **pthread_create**(3). Passing **NULL** is equivalent to passing a thread attribute object with all attributes set to their default values.

pthread_attr_init initializes the thread attribute object *attr* and fills it with default values for the attributes. (The default values are listed below for each attribute.)

Each attribute *attrname* (see below for a list of all attributes) can be individually set using the function **pthread_attr_set***attrname* and retrieved using the function **pthread_attr_get***attrname.*

pthread_attr_destroy destroys a thread attribute object, which must not be reused until it is reinitialized. **pthread_attr_destroy** does nothing in the LinuxThreads implementation.

Attribute objects are consulted only when creating a new thread. The same attribute object can be used for creating several threads. Modifying an attribute object after a call to **pthread_create** does not change the attributes of the thread previously created.

The following thread attributes are supported:

detachstate

Control whether the thread is created in the joinable state (value **PTHREAD_CREATE_JOINABLE**) or in the detached state (**PTHREAD_CREATE_DETACHED**).

Default value: **PTHREAD_CREATE_JOINABLE**.

In the joinable state, another thread can synchronize on the thread termination and recover its termination code using **pthread_join**(3), but some of the thread resources are kept allocated after the thread terminates, and reclaimed only when another thread performs **pthread_join**(3) on that thread.

In the detached state, the thread resources are immediately freed when it terminates, but **pthread_join**(3) cannot be used to synchronize on the thread termination.

A thread created in the joinable state can later be put in the detached thread using **pthread_detach**(3).

schedpolicy

Select the scheduling policy for the thread: one of **SCHED_OTHER** (regular, non-realtime scheduling), **SCHED_RR** (realtime, round-robin) or **SCHED_FIFO** (realtime, first-in first-out). See **sched_setpolicy**(2) for more information on scheduling policies.

Default value: **SCHED_OTHER**.

The realtime scheduling policies **SCHED_RR** and **SCHED_FIFO** are available only to processes with superuser privileges.

The scheduling policy of a thread can be changed after creation with **pthread_setschedparam**(3).

schedparam

Contain the scheduling parameters (essentially, the scheduling priority) for the thread. See **sched_setparam**(2) for more information on scheduling parameters.

Default value: priority is 0.

This attribute is not significant if the scheduling policy is **SCHED_OTHER**; it only matters for the realtime policies **SCHED_RR** and **SCHED_FIFO**.

The scheduling priority of a thread can be changed after creation with **pthread_setschedparam**(3).

inheritsched

Indicate whether the scheduling policy and scheduling parameters for the newly created thread are determined by the values of the *schedpolicy* and *schedparam* attributes (value **PTHREAD_EXPLICIT_SCHED**) or are inherited from the parent thread (value **PTHREAD_INHERIT_SCHED**).

Default value: **PTHREAD_EXPLICIT_SCHED**.

scope

Define the scheduling contention scope for the created thread. The only value supported in the LinuxThreads implementation is **PTHREAD_SCOPE_SYSTEM**, meaning that the threads contend for CPU time with all processes running on the machine. In particular, thread priorities are interpreted relative to the priorities of all other processes on the machine. The other value specified by the standard, **PTHREAD_SCOPE_PROCESS**, means that scheduling contention occurs only between the threads of the running process: thread priorities are interpreted relative to the priorities of the other threads of the process, regardless of the priorities of other processes. **PTHREAD_SCOPE_PROCESS** is not supported in LinuxThreads.

Default value: **PTHREAD_SCOPE_SYSTEM**.

RETURN VALUE

All functions return 0 on success and a non-zero error code on error. On success, the **pthread_attr_get***attrname* functions also store the current value of the attribute *attrname* in the location pointed to by their second argument.

ERRORS

The **pthread_attr_setdetachstate** function returns the following error codes on error:

EINVAL
: the specified *detachstate* is not one of **PTHREAD_CREATE_JOINABLE** or **PTHREAD_CREATE_DETACHED**.

The **pthread_attr_setschedparam** function returns the following error codes on error:

EINVAL
: the priority specified in *param* is outside the range of allowed priorities for the scheduling policy currently in *attr* (1 to 99 for **SCHED_FIFO** and **SCHED_RR**; 0 for **SCHED_OTHER**).

The **pthread_attr_setschedpolicy** function returns the following error codes on error:

EINVAL
: the specified *policy* is not one of **SCHED_OTHER**, **SCHED_FIFO**, or **SCHED_RR**.

ENOTSUP
: *policy* is **SCHED_FIFO** or **SCHED_RR**, and the effective user of the calling process is not super-user.

The **pthread_attr_setinheritsched** function returns the following error codes on error:

EINVAL
: the specified *inherit* is not one of **PTHREAD_INHERIT_SCHED** or **PTHREAD_EXPLICIT_SCHED**.

The **pthread_attr_setscope** function returns the following error codes on error:

EINVAL
: the specified *scope* is not one of **PTHREAD_SCOPE_SYSTEM** or **PTHREAD_SCOPE_PROCESS**.

ENOTSUP
: the specified *scope* is **PTHREAD_SCOPE_PROCESS** (not supported).

AUTHOR

Xavier Leroy <Xavier.Leroy@inria.fr>

SEE ALSO

pthread_create(3), **pthread_join**(3), **pthread_detach**(3), **pthread_setschedparam**(3).