# Атрибуты мутекса

# pthread_mutexattr_t

```
#include <pthread.h>

int pthread_mutexattr_init(
  pthread_mutexattr_t * attr);

int pthread_mutexattr_destroy(
  pthread_mutexattr_t *attr);
```

# Использование

```
#include <pthread.h>

int  pthread_mutex_init(
   pthread_mutex_t   *restrict   mutex,
   const pthread_mutexattr_t *restrict attr);
```

Или

```
pthread_mutex_t mutex= PTHREAD_MUTEX_INITIALIZER;

int pthread_mutex_destroy(pthread_mutex_t *mutex);
```

# Операции над pthread_mutexattr_t

- pthread_mutexattr_get/settype
- pthread_mutexattr_get/setpshared
- pthread_mutexattr_get/setprotocol
- pthread_mutexattr_get/setprioceiling
- pthread_mutexattr_get/setrobust_np

# pthread_mutexattr_get/settype

```
#include <pthread.h>

int pthread_mutexattr_gettype(
  pthread_mutexattr_t  *restrict attr,
  int *restrict type);

int pthread_mutexattr_settype(
  pthread_mutexattr_t *_attr,
  int  type);
```

# Типы

- PTHREAD_MUTEX_NORMAL
- PTHREAD_MUTEX_ERRORCHECK
- PTHREAD_MUTEX_RECURSIVE
- PTHREAD_MUTEX_DEFAULT

# pthread_mutexattr_get/setpshared

```
#include <pthread.h>

int  pthread_mutexattr_getpshared(
   const  pthread_mutexattr_t *restrict attr,
   int *restrict pshared);

int pthread_mutexattr_setpshared(
   pthread_mutexattr_t  * attr,
   int pshared);
```

PTHREAD_PROCESS_SHARED
PTHREAD_PROCESS_PRIVATE

# pthread_mutexattr_get/setrobust_np

```
#include <pthread.h>

int pthread_mutexattr_getrobust_np(
  const pthread_mutexattr_t *attr,
  int *robustness);

int  pthread_mutexattr_setrobust_np(
  pthread_mutexattr_t  *attr,
  int robustness);
```

# Robustness

- PTHREAD_MUTEX_ROBUST_NP
- PTHREAD_MUTEX_STALLED_NP

# Что означает ROBUST

- Если процесс, удерживающий мутекс, умер, ресурс, защищенный этим мутексом, остается в несогласованном состоянии
- Попытки его захватить возвращают EOWNERDEAD
- Если ресурс удалось восстановить, следует вызвать функцию pthread_mutex_consistent_np

# pthread_mutex_consistent_np

```
int pthread_mutex_consistent_np(
  pthread_mutex_t *mutex);
```

# pthread_mutexattr_get/setprotocol

```
#include <pthread.h>

int pthread_mutexattr_getprotocol(
  const  pthread_mutexattr_t *restrict attr,
  int *restrict protocol);

int pthread_mutexattr_setprotocol(
  pthread_mutexattr_t *attr,
     int protocol);
```

# Протоколы

- PTHREAD_PRIO_NONE,
- PTHREAD_PRIO_INHERIT,
- PTHREAD_PRIO_PROTECT

# Инверсия приоритета

- INHERIT – наследование приоритета
- Нить, удерживающая мутекс, исполняется с приоритетом, наивысшим среди всех нитей, ждущих этого мутекса (включая себя)

# Инверсия приоритета

- PROTECT – потолок приоритета (priority ceiling)
- Нить, удерживающая мутекс, исполняется с наивысшим приоритетом из всех нитей, которые *могут* удерживать этот мутекс.

# pthread_mutexattr_get/setprioceiling

```
#include <pthread.h>

int  pthread_mutexattr_getprioceiling(
   const  pthread_mutexattr_t  *restrict attr,
   int *restrict prioceiling);

int     pthread_mutexattr_setprioceiling(
   pthread_mutexattr_t *attr,
   int prioceiling, int *oldceiling);
```