

Операционные системы

Семафоры и мьютексы в ОС Unix

Олег Французов
2017

System V IPC

- Очереди сообщений
- Области разделяемой памяти
- Массивы семафоров

`msgget`, `shmat`, `semget`

System V IPC

- Существуют независимо от породившего процесса
- Имеют права доступа
- Являются объектами ядра

Семафоры System V IPC

- Объект содержит массив семафоров
- Возможность формировать сложные запросы, которые будут атомарны
- Блокировать, пока семафор $\neq 0$
- Мьютексов нет

Семафоры и мьютексы POSIX

- Переменная в процессе
- Только up и down для семафоров
- Стандарт предусматривает использование семафоров из различных процессов

pthread

```
void *thread_main(void *arg) {  
    /* . . . */  
}
```

Главная функция треда

Аргумент

Возвращаемое значение

Идентификатор треда имеет тип pthread_t

pthread

```
int pthread_create(pthread_t *thr,  
                  pthread_attr_t *attr,  
                  void *(*start_routine)(void *),  
                  void *arg);
```

attr можно NULL

Идентификатор треда запишется в thr

pthread

```
/* возвращение значения */
```

```
return retval;
```

```
void pthread_exit(void *retval);
```

```
/* ожидание треда */
```

```
void pthread_join(pthread_t thr,  
                  void **result);
```


threads

```
int pthread_detach(pthread_t thr);
```

```
pthread_t pthread_self();
```

```
/* detach self */
```

```
pthread_detach(pthread_self());
```

```
int pthread_cancel(pthread_t thr);
```

```
/* result == PTHREAD_CANCELED */
```

Мьютексы pthreads

```
pthread_mutex_t my_mutex =  
    PTHREAD_MUTEX_INITIALIZER;  
  
int pthread_mutex_lock(pthread_mutex_t *mtx);  
int pthread_mutex_trylock(pthread_mutex_t *mtx);  
int pthread_mutex_unlock(pthread_mutex_t *mtx);  
  
int pthread_mutex_destroy(pthread_mutex_t *mtx);
```

Семафоры POSIX

```
int sem_init(sem_t *sem, int pshared,  
             unsigned int value);  
/* also sem_open */  
  
int sem_post(sem_t *sem); /* up */  
int sem_wait(sem_t *sem); /* down */  
  
int sem_trywait(sem_t *sem);  
int sem_getvalue(sem_t *sem, int *sval);  
  
int sem_destroy(sem_t *sem);
```

Производители и потребители

- с. 179-181

Q & A