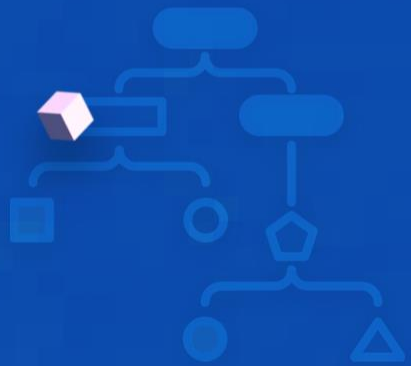
강원혁신플랫폼

리눅스프로그래밍

메시지 큐



Linux 시스템에서 프로세스가 큐 기반 메커니즘을 통해 메시지를 교환할 수 있도록 하는 프로세스 간 통신의 한 형태는 무엇인가요?

메시지큐

학습목표

학습 내용

1 메시지 큐의 주요 함수

학습 목표

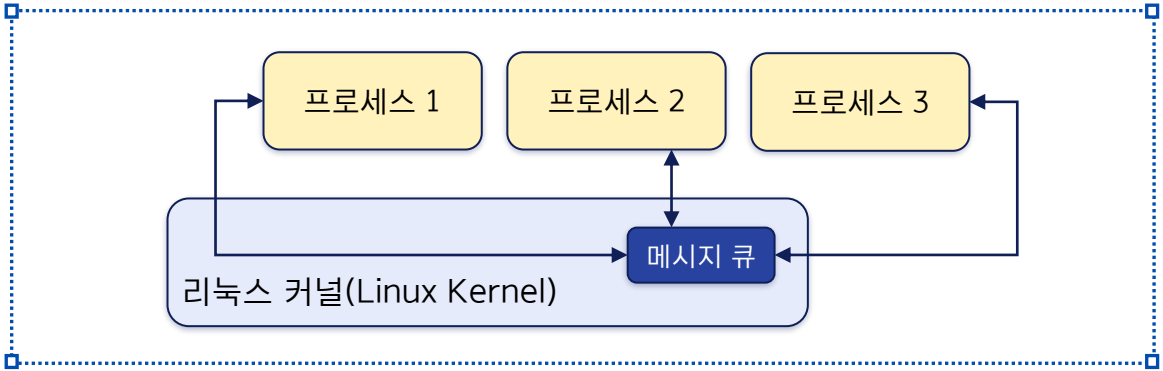
메시지 큐의 주요 함수를 파악할 수 있다.

메시지 큐의 주요 함수

메시지 큐의 다중화

메시지 큐의 특징

- Linux에서 메시지 큐는 프로세스가 큐 기반 메커니즘을 통해 메시지를 교환할 수 있도록 하는 프로세스 간 통신(IPC)의 한 형태
- 메시지 대기열은 프로세스가 서로 통신할 수 있는 유연하고 안정적인 방법을 제공



출처 서영진, 사물인터넷을 위한 리눅스 프로그래밍 with 라즈베리 파이, 제이펍, 2020

메시지 큐의 msqid_ds

- 메시지 큐에서는 msqid_ds 구조체를 사용

```
#include <sys/msg.h>

struct msqid_ds {
        struct ipc_perm msg_perm;   /* Ownership and permissions */
        time_t          msg_stime;  /* Time of last msgsnd(2) */
        time_t          msg_rtime;  /* Time of last msgrcv(2) */
        time_t          msg_ctime;  /* Time of creation or last
                                       modification by msgctl() */
        unsigned long   msg_cbytes; /* # of bytes in queue */
```

메시지 큐의 msqid_ds

- 메시지 큐에서는 msqid_ds 구조체를 사용

```
msgqnum_t       msg_qnum;   /* # number of messages in queue */
        msglen_t        msg_qbytes; /* Maximum # of bytes in queue */
        pid_t           msg_lspid;  /* PID of last msgsnd(2) */
        pid_t           msg_lrpid;  /* PID of last msgrcv(2) */
};
```

메시지 큐의 주요 함수

msgget(): 새 메시지 대기열을 생성하거나 기존 메시지 대기열의 식별자를 검색

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

int msgget(key_t key, int msgflg);
```

메시지 큐의 주요 함수

msgsnd(): 메시지 큐에 메시지를 보냄

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

int msgsnd(int msqid, const void *msgp, size_t msgsz, int msgflg);
```

메시지 큐의 주요 함수

msgrcv(): 메시지 큐에서 메시지를 받음

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

ssize_t msgrcv(int msqid, void *msgp, size_t msgsz, long msgtyp, int
msgflg);
```

메시지 큐의 주요 함수

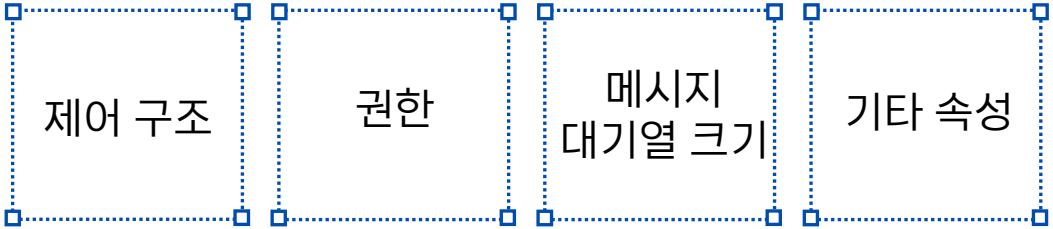
msgctl(): 삭제 또는 수정과 같은 메시지 대기열에 대한 제어 작업을 수행

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

int msgctl(int msqid, int cmd, struct msqid_ds *buf);
```

메시지 큐의 주요 함수

- struct msqid_ds



메시지 큐와 관련된 정보 보유

메시지 큐의 주요 함수

IPC_CREAT 및 IPC_EXCL

- 메시지 큐의 생성 및 존재 확인을 지정하기 위해 msgget()과 함께 사용할 수 있는 플래그 값

- IPC_CREAT : key값으로 생성된 message queue가 존재하지 않으면 message queue를 신규로 생성
- IPC_EXCL : 이미 생성된 message queue가 있으면 오류 발생

따라하기 msg_server.c


```
#include <stdio.h>
#include <unistd.h>
#include <sys/msg.h>

#define MSQKEY 51234

struct msgbuf {
    long mtype; /* 메시지의 타입 : 0 이상의 정숫값 */
    char mtext[BUFSIZ]; /* 메시지의 내용 : 1바이트 이상의 문자열 */
};
```

따라하기 msg_server.c


```
int main(int argc, char **argv)
{
    key_t key;
    int n, msqid;
    struct msgbuf mb;
    key = MSQKEY;

    /* 메시지 큐의 채널을 생성한다. */
    if((msqid = msgget(key, IPC_CREAT | IPC_EXCL | 0666)) < 0) {
        perror("msgget()");
        return -1;
    }
```

따라하기 msg_server.c


```
/* 메시지 큐에서 데이터를 가져온다. */
    while((n = msgrcv(msqid, &mb, sizeof(mb), 0, 0)) > 0) {
        switch (mb.mtype) {
            /* 메시지 타입(mtype)이 1이면 화면에 가져온 데이터를 출력한다. */
            case 1:
                write(1, mb.mtext, n);
                break;
            /* 메시지 타입(mtype)이 2이면 메시지 큐의 채널을 삭제한다. */
            case 2:
                if(msgctl(msqid, IPC_RMID, (struct msqid_ds *) 0) < 0) {
                    perror("msgctl()");
                    return -1;
```

```
}
            break;
        }
    }

    return 0;
}
```

따라하기 msg_client.c


```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/msg.h>

#define MSQKEY 51234

struct msgbuf {
    long mtype;
    char mtext[BUFSIZ];
};
```

따라하기 msg_client.c


```
int main(int argc, char **argv)
{
    key_t key;
    int rc, msqid;
    char* msg_text = "hello world\n";

    struct msgbuf *mb;
    mb = (struct msgbuf*)malloc(sizeof(struct msgbuf) + strlen(msg_text));
```

따라하기 msg_client.c


```
key = MSQKEY;
    if((msqid = msgget(key, 0666)) < 0) {              /* 메시지 큐의 채널을 가져온다. */
        perror("msgget()");
        return -1;
    }
    /* mtype을 1로 설정하고 hello world라는 문자열을 보낸다. */
    mb->mtype = 1;
    strcpy(mb->mtext, msg_text);
    rc = msgsnd(msqid, mb, strlen(msg_text)+1, 0);      /* 메시지 큐로 데이터를 보낸다. */
    if(rc == -1) {
        perror("msgsnd()");
        return -1;
```

```
}
    /* mtype을 2로 설정하고 보낸다. */
    mb->mtype = 2;
    memset(mb->mtext, 0, sizeof(mb->mtext));
    if(msgsnd(msqid, mb, sizeof(mb->mtext), 0) < 0) {
        perror("msgsnd()");
        return -1;
    }
    return 0;
}
```

메시지 큐 실행결과

실행결과

- msg_client는 msqid를 구해 문자열을 msgsnd()로 송신,
- msg_server는 msqid를, 문자열을 msgget()로 수신 화면에 출력, mtype이 2면 큐 채널 삭제

```
$ gcc -o msg_server msg_server.c
$ gcc -o msg_client msg_client.c
$
$ ./msg_server &
[1] 3539
$ ./msg_client
hello world
[1]+  Done                    ./msg_server
$
```

메시지 큐 관련 명령


```
$ ./msg_server &
[1] 3542

명령어로 IPC상태 확인
$ ipcs
------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
0x0000c822 1          freetime   666        0            0

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
```

메시지 큐 관련 명령


```
------ Semaphore Arrays --------
key        semid      owner      perms      nsems

ipcrm – remove certain IPC resources
$ ipcrm -q 1
[1]+  Done                    ./msg_server
$
```

01 • 메시지 큐의 주요 함수

<sys/msg.h>
msgget( )
msgctl( )
msgsnd( ), msgrcv( )