4.7 进程间通讯 - 消息队列 (二)_物联网 / 嵌入式工程师 - 慕课网

> 慕课网慕课教程 4.7 进程间通讯 – 消息队列（二）涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

函数头文件

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

#include <sys/ipc.h>

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

函数原型

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

1. 函数参数:
    1. msqid：消息队列 ID
    2. msgp：消息结构体指针
    3. msgsz：消息内容的长度
    4. msgflg：消息队列标志，默认可以填 0
        1. IPC_NOWAIT：可以设置非阻塞

函数返回值

- 成功：返回 0
- 失败： –1, 并设置 errno

消息结构定义形式如下:

```
struct msgbuf {

 long mtype; /* message type, must be > 0 */

 char mtext[1]; /* message data */

 };
```

- mtype：消息类型
- mtext：消息内容数组

函数头文件

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

#include <sys/ipc.h>

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

函数原型

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

1. 函数参数
    1. msqid：消息队列 id
    2. msgp：消息结构指针
    3. msgtyp：消息类型
    4. msgflg：消息队列标志，默认可以填 0
        1. IPC_NOWAIT：可以设置非阻塞
2.

- 函数返回值:
    - 成功：返回实际读取消息内容的字节数
    - 失败：-1, 并设置 errno
- 示例: 创建两个没有血缘关系的进程, 使用 消息队列进行通讯
    - msg_write.c
    -
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>


#define PATHNAME "."
#define PRO_ID 10

#define MSG_TYPE 100
#define MSG_SZ 64

struct msgbuf{
    long mtype;
    char mtext[MSG_SZ];
};

int main(void)
{
    key_t key;
    int msgid,ret;
    ssize_t rbytes;
    struct msgbuf rcv_msg;

    key = ftok(PATHNAME,PRO_ID);
    if(key == -1){
        perror("[ERROR] fotk(): ");
        exit(EXIT_FAILURE);
    }

    msgid = msgget(key,IPC_CREAT | 0666);
    if(msgid == -1){
        perror("msgget(): ");
        exit(EXIT_FAILURE);
    }

    printf("msg id : %d\n",msgid);


    rbytes = msgrcv(msgid,(void *)&rcv_msg,MSG_SZ,MSG_TYPE,0);

    if(rbytes == -1){
        perror("[ERROR] msgrcv(): ");
        exit(EXIT_FAILURE);
    }

    printf("mtype : %ld\n",rcv_msg.mtype);
    printf("mtext : %s\n",rcv_msg.mtext);

    ret = msgctl(msgid,IPC_RMID,NULL);
    if(ret == -1){
        perror("msgctl(): ");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```
    - msg_write.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

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


#define PATHNAME "."
#define PRO_ID 10

#define MSG_TYPE 100
#define MSG_SZ 64

struct msgbuf{
    long mtype;
    char mtext[MSG_SZ];
};

int main(void)
{
    key_t key;
    int msgid,ret;
    struct msgbuf msg;

    key = ftok(PATHNAME,PRO_ID);
    if(key == -1){
        perror("fotk(): ");
        exit(EXIT_FAILURE);
    }

    msgid = msgget(key,IPC_CREAT | 0666);
    if(msgid == -1)
    {
        perror("msgget(): ");
        exit(EXIT_FAILURE);
    }

    printf("msg id : %d\n",msgid);

    msg.mtype = MSG_TYPE;
    strcpy(msg.mtext,"Hello msg queue");

    ret = msgsnd(msgid,(const void *)&msg,strlen(msg.mtext) + 1,0);

    if(ret == -1)
    {
        perror("msgsnd(): ");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```

练习

- 创建两个子进程 A 与 B，父进程分别给两个子进程发型消息，消息类型为 100 与 200
- 父进程从键盘循环接收数据, 发送给子进程，输入 quit 则退出

全文完