1. 做一个daemon进程，让其每2秒钟，在/tmp/log文件中写一句 wohenshuai

#include <func.h>

void Daemon()

{

    if(fork())//创建子进程，父进程退出

    {

        exit(0);

    }

    setsid();//成立新会话

    chdir("/");//改变路径到根

    umask(0);//修改文件掩码

    int i;

    for(i=0;i<3;i++)

    {

        close(i);

    }

    return;

}

int main()

{

    Daemon();

    int fd,ret;

    char buf[128]="wohenshuai";

    fd=open("/tmp/log",O\_RDWR|O\_CREAT,0666);

    ERROR\_CHECK(fd,-1,"open")

    while(1)

    {

        sleep(2);

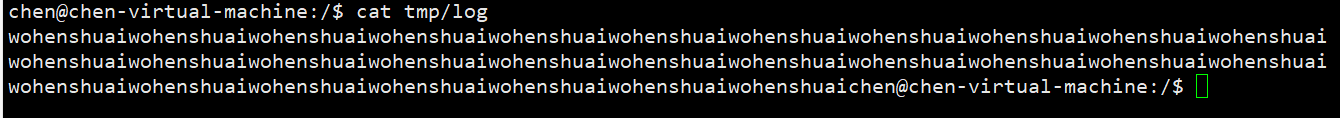
        ret=write(fd,buf,strlen(buf));

        ERROR\_CHECK(ret,-1,"write")

    }

    return 0;

}



1. 创建一个无名管道，fork子进程，然后子进程向管道写端写Hello，父进程接收数据并打印

#include <func.h>

int main()

{

    int fds[2];//数组

    pipe(fds);//初始化一条无名管道，fds[0]读端，fds[1]写端

    if(!fork())

    {

        close(fds[0]);

        //孩子写

        write(fds[1],"Hello",5);

        return 0;

    }else{

        close(fds[1]);

        //父亲读

        sleep(3);

        char buf[128]={0};

        read(fds[0],buf,sizeof(buf));//管道没数据会阻塞

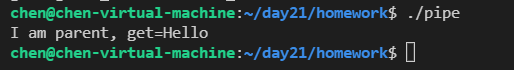
        printf("I am parent, get=%s\n",buf);

        wait(NULL);

        return 0;

    }

}



1. 新建共享内存，连接，然后往共享内存写入How are you,然后另外一个进程连接共享内存，读取数据并打印显示,删除共享内存

#include <func.h>

int main()

{

    int shmid;

    shmid=shmget(1000,4096,IPC\_CREAT|0600);

    ERROR\_CHECK(shmid,-1,"shmget")

    printf("shmid=%d\n",shmid);

    char \*p=shmat(shmid,NULL,0);//共享内存连接到本进程内

    ERROR\_CHECK(p,(char\*)-1,"shmat")

    if(!fork())

    {

        strcpy(p,"How are you");

        sleep(1);

        return 0;

    }else{

        wait(NULL);

        printf("%s\n",p);

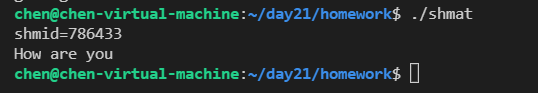
        int ret=shmctl(shmid,IPC\_RMID,NULL);

        ERROR\_CHECK(ret,-1,"shmctl")

        return 0;

    }

}



1. 两个进程对一个共享内存各加1000万，查看最后结果

#include <func.h>

#define NUM 10000000

int main()

{

    int shmid;

    shmid=shmget(1000,4096,IPC\_CREAT|0600);

    ERROR\_CHECK(shmid,-1,"shmget")

    printf("shmid=%d\n",shmid);

    int \*p=shmat(shmid,NULL,0);//共享内存连接到本进程内

    ERROR\_CHECK(p,(char\*)-1,"shmat")

    p[0]=0;

    int i;

    //对内存操作产生了并发，没有加锁，造成结果不等于20000000

    if(!fork())

    {

        for(i=0;i<NUM;i++)

        {

            p[0]++;

        }

        return 0;

    }else{

        for(i=0;i<NUM;i++)

        {

            p[0]++;

        }

        wait(NULL);

        printf("p[0]=%d\n",p[0]);

        return 0;

    }

}

