# 命名管道的概述

命名管道（FIFO）具有名称，**以 FIFO 的文件形式存在于文件系统中**，进程通过访问该文件进行通信

# 命名管道的创建

#include <sys/types.h>

#include <sys/stat.h>

int mkfifo( const char \*pathname, mode\_t mode);

功能：

命名管道的创建。

参数：

pathname: 普通的路径名，也就是创建后 FIFO 的名字。

mode: 文件的权限，与打开普通文件的 open() 函数中的 mode 参数相同

返回值：

成功：0  
 失败：如果文件已经存在，则会出错且返回 -1。

示例代码如下：

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

int main(int argc, char \*argv[])

{

    int ret;

    ret = mkfifo("my\_fifo", 0666); // 创建命名管道

    if(ret != 0){   // 出错

        perror("mkfifo");

    }

    return 0;

}

运行结果如下：



# 命名管道的默认操作

命名管道操作与普通文件一样：open()、write()、read()、close()。

int ret = mkfifo("my\_fifo", 0666);

// 以只读或只写打开，进程会阻塞到另一个进程以只写或只读打开

int fd = open("my\_fifo", O\_RDONLY); // 只写打开

int fd = open("my\_fifo", O\_WRONLY); // 只读打开

int fd = open("my\_fifo", O\_RDWR);   // 可读写打开

假如 FIFO 里没有数据，调用 read() 函数从 FIFO 里读数据时 read() 也会阻塞  
调用 write() 函数向 FIFO 里写数据，当缓冲区已满时 write() 也会阻塞

# 

# 命名管道非阻塞标志操作

命名管道可以以非阻塞标志（O\_NONBLOCK）方式打开：

fd = open("my\_fifo", O\_WRONLY|O\_NONBLOCK);

fd = open("my\_fifo", O\_RDONLY|O\_NONBLOCK);

非阻塞标志（O\_NONBLOCK）打开的命名管道有以下特点：  
2、先以只写方式打开，必须有另一个进程以读方式打开FIFO，否则 open() 将出错返回 -1。  
3、read()、write() 读写命名管道中读数据时不阻塞。

**示例**

#include <iostream>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <fstream>

int main(int argc, char \*argv[])

{

    const char\* fifoName = "myfifo";

    // 创建管道

    if(mkfifo(fifoName, 0666) < 0){

        std::cout << "管道创建错误" << std::endl;

        return 0;

}

    int childPid = fork();

    if(childPid == 0){

        // 子进程向管道写入数据

        std::ofstream file;

        file.open(fifoName);

        file << "这是子进程发送的消息";

        file.close();

    }

    else{

        // 父进程从管道读取数据

        std::ifstream file;

        file.open(fifoName);

        std::string text;

        file >> text;

        std::cout << text;

        file.close();

}

    return 0;

}