**Java如何使用管道通讯**

上面说到，管道实际上就是一个file结构和一个VFS的索引。也就是说管道是一个虚拟的文件。那在Java中是不是就可以通过直接读写这个文件，从而实现PIPE通讯的效果呢？

答案当然是肯定的。其步骤如下：

1）首先要创建一个管道文件，这一点Java 做不到，我们要借助C/C++中的mkfifo()函数来实现。

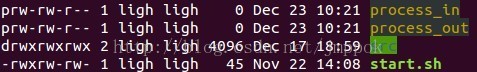
以下代码创建并打开两个管道，一个用于读取输入，一个用于输出：

[cpp]  [view plain](https://blog.csdn.net/jmppok/article/details/17500739)  [copy](https://blog.csdn.net/jmppok/article/details/17500739)

1. **char** name1[1024];
2. **char** name2[1024];
3. sprintf(name1,"./process\_in");
4. sprintf(name2,"./process\_out");
5. **if**(mkfifo(name1,0666 )<0)
6. {
7. printf("\n error when mkfifo");
8. }
9. **if**(mkfifo(name2,0666 )<0)
10. {
11. printf("\n error when mkfifo");
12. }

15. **int** fd1, fd2;
16. **if** ((fd1 = open (name1, O\_RDWR ))<0)
17. {
18. perror("Could not open named pipe.");
19. }
20. **if** ((fd2 = open (name2, O\_RDWR ))<0)
21. {
22. perror("Could not open named pipe.");
23. }

这是查看当前目录，可以看到两个Pipe文件，它和普通文件不同，具有p属性，表明是一个管道文件。



2）Java中，使用文件读写的方式打开这两个文件，即可进行读写。

[java]  [view plain](https://blog.csdn.net/jmppok/article/details/17500739)  [copy](https://blog.csdn.net/jmppok/article/details/17500739)

1. String namedPipe1=workdir+"/process\_in";
2. String namedPipe2=workdir+"/process\_out";
4. File pipe1 = **new** File(namedPipe1);
5. File pipe2 = **new** File(namedPipe2);
7. BufferedReader reader = **new** BufferedReader(**new** FileReader(pipe2));
8. //DataInputStream reader = new DataInputStream(new FileInputStream(pipe2));
9. BufferedWriter writer = **new** BufferedWriter(**new** FileWriter(pipe1));
11. writer.write("test");
13. **char** [] buf = **new** **char**[8];
14. reader.read(buf);

 3)需要注意的问题  
  
  1)Java最好用单独的线程来读数据；  
  
  2)Java写完数据后，需要flush（）；