今日学习心得

1. 简单的重定向

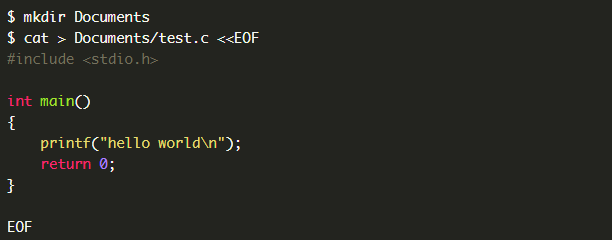
Linux 默认提供了三个特殊设备，用于终端的显示和输出，分别为stdin（标准输入,对应于你在终端的输入），stdout（标准输出，对应于终端的输出），stderr（标准错误输出，对应于终端的输出）。



文件描述符：文件描述符在形式上是一个非负整数。实际上，它是一个索引值，指向内核为每一个进程所维护的该进程打开文件的记录表。当程序打开一个现有文件或者创建一个新文件时，内核向进程返回一个文件描述符。在程序设计中，一些涉及底层的程序编写往往会围绕着文件描述符展开。但是文件描述符这一概念往往只适用于 UNIX、Linux 这样的操作系统。

默认使用终端的标准输入作为命令的输入和标准输出作为命令的输出。

将cat的连续输出（heredoc方式）重定向到一个文件：



将一个文件作为命令的输入，标准输出作为命令的输出：



将echo命令通过管道传过来的数据作为cat命令的输入，将标准输出作为命令的输出：

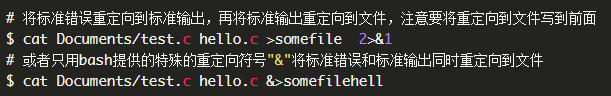


将echo命令的输出从默认的标准输出重定向到一个普通文件：



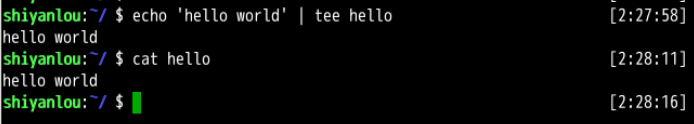
2. 标准错误重定向

另一个很实用的操作是将标准错误重定向，标准输出和标准错误都被指向伪终端的屏幕显示，所以我们经常看到的一个命令的输出通常是同时包含了标准输出和标准错误的结果的。



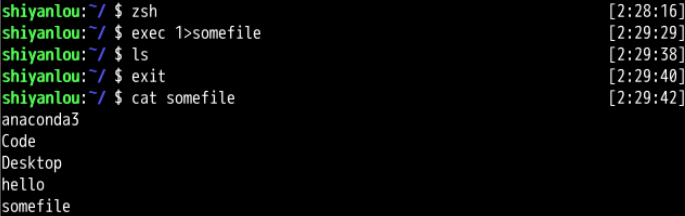
3. 使用tee命令同时重定向到多个文件

可能还有这样的需求，除了需要将输出重定向到文件,也需要将信息打印在终端。那么你可以使用tee命令来实现：



4. 永久重定向

前面的重定向操作都只是临时性的，即只对当前命令有效，要做到“永久”有效，比如在一个脚本中，需要某一部分的命令的输出全部进行重定向，可以使用exec命令实现“永久”重定向。exec命令的作用是使用指定的命令替换当前的 Shell，即使用一个进程替换当前进程，或者指定新的重定向。



5. 创建输出文件描述符

在 Shell 中有9个文件描述符。上面我们使用了也是它默认提供的0,1,2号文件描述符。另外我们还可以使用3-8的文件描述符，只是它们默认没有打开而已。你可以使用下面命令查看当前 Shell 进程中打开的文件描述符：



6. 关闭文件描述符

可以使用如下操作将操作符关闭：

