SHELL程序设计- 重定向

目录

目录

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。

习惯上，

标准输入（standard input）文件描述符是 0

标准输出（standard output）文件描述符是 1

标准错误（standard error）文件描述符是 2

目录

输出重定向

通常情况下，所谓**输出重定向**是指**将原本输出到标准输出的数据输出到其他文件或者设备中**。输出重定向分为

- **覆盖输出重定向**（**操作符为大于号>**）

```
<shell命令> <文件描述符> > <文件>
```

- **追加输出重定向**（**操作符为双大于号>>**）

```
<shell命令> <文件描述符> >> <文件>
```

常用的文件描述符为0、1和2。其中1和2是用来输出数据的，分别表示标准输出和标准错误。如果描述符省略，默认为标准输出（1）。

对于标准输出和标准错误的同时重定向，还有一种更为简洁的语法，操作符是 **&>**。

```
<shell命令> &> <file文件>
```

输出重定向示例

```
[root@c84n0 ~]# cat tmp10.txt
dasdas
dasdadsa
[root@c84n0 ~]# hostname
c84n0
[root@c84n0 ~]# hostname > tmp10.txt
[root@c84n0 ~]# cat tmp10.txt
c84n0
[root@c84n0 ~]# hostname >> tmp10.txt
[root@c84n0 ~]# cat tmp10.txt
c84n0
c84n0
[root@c84n0 ~]# > tmp10.txt
[root@c84n0 ~]# cat tmp10.txt
[root@c84n0 ~]#
[root@c84n0 ~]#
```

覆盖

追加

清空

输出重定向到/dev/null

如果希望执行某个命令，但又不希望在屏幕上显示输出结果，那么可以将输出重定向到 **/dev/null**：

```
<shell命令> <文件描述符>> /dev/null
```

/dev/null 是一个特殊的文件，写入到它的内容都会被丢弃；如果尝试从该文件读取内容，那么什么也读不到。但是 **/dev/null** 文件非常有用，将命令的输出重定向到它，会起到"禁止输出"的效果

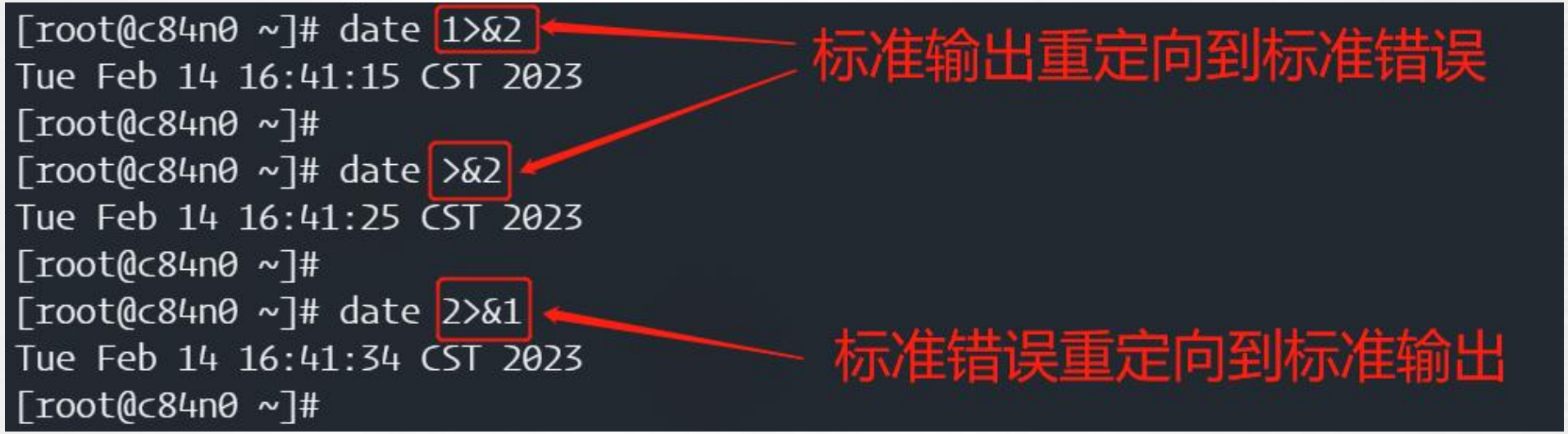
```
[root@c84n0 ~]# hostname
c84n0
[root@c84n0 ~]# hostname > /dev/null
[root@c84n0 ~]# hostname 2> /dev/null
c84n0
[root@c84n0 ~]# hostname 1> /dev/null
[root@c84n0 ~]#
```

重定向两个文件描述符

可以通过重定向操作将到一个文件描述符的输出重定向到另外一个文件描述符，即复制一个文件描述符，操作符是 **>&**。

```
<n>>&<m>
```

n 和 **m** 都是文件描述符。当 n=1，且m=2时，文件描述符1成为文件描述符2的副本，所以所有的标准输出都被重定向到标准错误；而当n=2，m=1时，文件描述符2成为文件描述符1的副本，所以所有的标准错误都被重定向到标准输出。**当n=1时，可以省略。除了使用操作符>&重定向文件描述符之外，还可以使用操作符<&来重定向文件描述符。**



目录

输入重定向

通常情况下，Shell 命令会从标准输入，即键盘读取用户输入的数据。但是Shell提供了另外一种读取用户输入的机制，即从文件中获取输入，这种机制称为输入重定向。输入重定向与输出重定向非常相似，其操作符为**小于号<**。

```
<shell命令> < <文件>
```

注意：操作符前实际上存在一个可省略的输入重定向默认的文件描述符0。如果存在文件描述符0，则0与小于号之间没有任何空格。

注意：输入重定向和输出重定向可以同时使用，表示从文件中读取输入数据，然后将执行结果输入到文件中。

```
[root@c84n0 ~]# grep tmpfs tmp2.txt
/run            tmpfs           4038329 990972
[root@c84n0 ~]# cat tmp10.txt
[root@c84n0 ~]# grep tmpfs > tmp10.txt < tmp2.txt
[root@c84n0 ~]# cat tmp10.txt
/run            tmpfs           4038329 990972
```

当前文档

输入重定向的另外一个用途是生成当前文档（here documents）。当前文档主要用在命令行中需要多行输入的情况。

```
<shell命令> << <分隔符>
…
…
<分隔符>
```

注意：<< 为输入重定向操作符，前后分隔符一摸一样，且不能含有空格或者制表符

当Shell遇到重定向操作符<<时，会一直读取用户的输入，直到遇到某一行，其中包含指定的分隔符。两个分隔符之间的行都属于SHELL命令的标准输入。最后的分隔符告诉 Shell 当前文档已经结束。如果没有后面的分隔符，Shell 将会继续读取输入，并会永远进行下去。

当前文档

```
[root@c84n0 ~]# cat << EOF
> /                nvme0n1p1       249855   166608
> /boot            nvme0n1p2       1024     15990784
> swap             nvme0n1p3       5120     65536
> /dev             devtmpfs        3946148  986537
> /dev/shm         tmpfs           3963888  990972
> /run             tmpfs           4038329  990972
> EOF
/                nvme0n1p1       249855   166608
/boot            nvme0n1p2       1024     15990784
swap             nvme0n1p3       5120     65536
/dev             devtmpfs        3946148  986537
/dev/shm         tmpfs           3963888  990972
/run             tmpfs           4038329  990972
```

当前文档可以重定向到其他文件

```
[root@c84n0 ~]# cat tmp10.txt
[root@c84n0 ~]# cat > tmp10.txt << EOF
> Initialized host personality
> NET: Registered protocol family 40
> EOF
[root@c84n0 ~]# cat tmp10.txt
Initialized host personality
NET: Registered protocol family 40
```

谢 谢