Shell printf 命令 Shell 文件包含

Shell 输入/输出重定向

大多数 UNIX 系统命令从你的终端接受输入并将所产生的输出发送回到您的终端。一个命令通常从一个叫标准输入的地方读取输入，默认情况下，这恰好是你的终端。同样，一个命令通常将其输出写入到标准输出，默认情况下，这也是你的终端。

重定向命令列表如下：

命令	说明
command > file	将输出重定向到 file。
command < file	将输入重定向到 file。
command >> file	将输出以追加的方式重定向到 file。
n > file	将文件描述符为 n 的文件重定向到 file。
n >> file	将文件描述符为 n 的文件以追加的方式重定向到 file。
n >& m	将输出文件 m 和 n 合并。
n <& m	将输入文件 m 和 n 合并。
<< tag	将开始标记 tag 和结束标记 tag 之间的内容作为输入。

> *需要注意的是文件描述符 0 通常是标准输入（STDIN），1 是标准输出（STDOUT），2 是标准错误输出（STDERR）。*

输出重定向

重定向一般通过在命令间插入特定的符号来实现。特别的，这些符号的语法如下所示:

```
command1 > file1
```

上面这个命令执行command1然后将输出的内容存入file1。

注意任何file1内的已经存在的内容将被新内容替代。如果要将新内容添加在文件末尾，请使用>>操作符。

实例

执行下面的 who 命令，它将命令的完整的输出重定向在用户文件中(users):

```
$ who > users
```

执行后，并没有在终端输出信息，这是因为输出已被从默认的标准输出设备（终端）重定向到指定的文件。

你可以使用 cat 命令查看文件内容：

```
$ cat users
_mbsetupuser console  Oct 31 17:35
tianqixin    console  Oct 31 17:35
tianqixin    ttys000  Dec  1 11:33
```

输出重定向会覆盖文件内容，请看下面的例子：

```
$ echo "菜鸟教程: www.runoob.com" > users
$ cat users
菜鸟教程: www.runoob.com
$
```

如果不希望文件内容被覆盖，可以使用 >> 追加到文件末尾，例如：

```
$ echo "菜鸟教程: www.runoob.com" >> users
$ cat users
菜鸟教程: www.runoob.com
菜鸟教程: www.runoob.com
$
```

输入重定向

和输出重定向一样，Unix 命令也可以从文件获取输入，语法为：

```
command1 < file1
```

这样，本来需要从键盘获取输入的命令会转移到文件读取内容。

注意：输出重定向是大于号(>)，输入重定向是小于号(<)。

实例

接着以上实例，我们需要统计 users 文件的行数,执行以下命令：

```
$ wc -l users
       2 users
```

也可以将输入重定向到 users 文件：

```
$  wc -l < users
       2
```

注意：上面两个例子的结果不同：第一个例子，会输出文件名；第二个不会，因为它仅仅知道从标准输入读取内容。

```
command1 < infile > outfile
```

同时替换输入和输出，执行command1，从文件infile读取内容，然后将输出写入到outfile中。

重定向深入讲解

一般情况下，每个 Unix/Linux 命令运行时都会打开三个文件：

- 标准输入文件(stdin)：stdin的文件描述符为0，Unix程序默认从stdin读取数据。
- 标准输出文件(stdout)：stdout 的文件描述符为1，Unix程序默认向stdout输出数据。
- 标准错误文件(stderr)：stderr的文件描述符为2，Unix程序会向stderr流中写入错误信息。

默认情况下，command > file 将 stdout 重定向到 file，command < file 将stdin 重定向到 file。

如果希望 stderr 重定向到 file，可以这样写：

```
$ command 2 > file
```

如果希望 stderr 追加到 file 文件末尾，可以这样写：

```
$ command 2 >> file
```

2 表示标准错误文件(stderr)。

如果希望将 stdout 和 stderr 合并后重定向到 file，可以这样写：

```
$ command > file 2>&1

或者

$ command >> file 2>&1
```

如果希望对 stdin 和 stdout 都重定向，可以这样写：

```
$ command < file1 >file2
```

command 命令将 stdin 重定向到 file1，将 stdout 重定向到 file2。

Here Document

Here Document 是 Shell 中的一种特殊的重定向方式，用来将输入重定向到一个交互式 Shell 脚本或程序。

它的基本的形式如下：

```
command << delimiter
    document
delimiter
```

它的作用是将两个 delimiter 之间的内容(document) 作为输入传递给 command。

注意：

- 结尾的delimiter 一定要顶格写，前面不能有任何字符，后面也不能有任何字符，包括空格和 tab 缩进。
- 开始的delimiter前后的空格会被忽略掉。

实例

在命令行中通过 wc -l 命令计算 Here Document 的行数：

```
$ wc -l << EOF
    欢迎来到
    菜鸟教程
    www.runoob.com
EOF
3           # 输出结果为 3 行
$
```

我们也可以将 Here Document 用在脚本中，例如：

```
#!/bin/bash
# author:菜鸟教程
# url:www.runoob.com

cat << EOF
欢迎来到
菜鸟教程
www.runoob.com
EOF
```

执行以上脚本，输出结果：

```
欢迎来到
菜鸟教程
www.runoob.com
```

/dev/null 文件

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：

```
$ command > /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。

如果希望屏蔽 stdout 和 stderr，可以这样写：

```
$ command > /dev/null 2>&1
```

> ***注意：*** *0 是标准输入（STDIN），1 是标准输出（STDOUT），2 是标准错误输出（STDERR）。*

← Shell printf 命令 Shell 文件包含 →