

重定向、cat与EOF

标准输入、标准输出、标准错误

运行程序会产生输出，这些输出包含两种类型，一种是程序运行的结果，一种是程序运行的状态和错误信息。与python一切皆对象的想法类似，UNIX“一切都是文件”...通过某些命令将结果显示到屏幕上(如ls)实际上该程序将运行结果发送到了一个称为标准输出(stdout)的文件中，将状态(错误)信息发送到了另一个叫标准错误(stderr)的文件中，而默认情况下这两个文件都被连接到屏幕上，因此可以在终端中直接看到这些程序的结果。同时，许多程序都从一个称作标准输入(stdin)的设备中获取输入，而默认情况下标准输入是与键盘连接的。

理解了输入输出的原理，就很容易理解I/O重定向了，键盘、屏幕相比于其他来源并没有任何特殊性，只要改变stdout, stderr, stdin文件链接的位置，就可以控制一个程序的输入与输出位置。

> 以覆盖文件的方式重定向标准输出

>> 以追加文件的方式重定向标准输出

同时，linux提供了使用文件描述符编号来重定向文件的表示法，0表示标准输入文件，1表示标准输出文件，2表示标准错误文件，因此在重定向标准错误时，可以：

2> 以覆盖的方式重定向标准错误

也可以将标准输出和标准错误重定向到同一个文件：

```
ls something 1> somefile 2>&1
```

1> 表示将ls的标准输出重定向到somefile中，2>&1 表示将标准错误重定向到标准输出中，可以观察下面一个更复杂的重定向帮助理解这一过程：

```
ls exist_file non_exist_file 1> output1.txt 2>&1 1> output2.txt
```

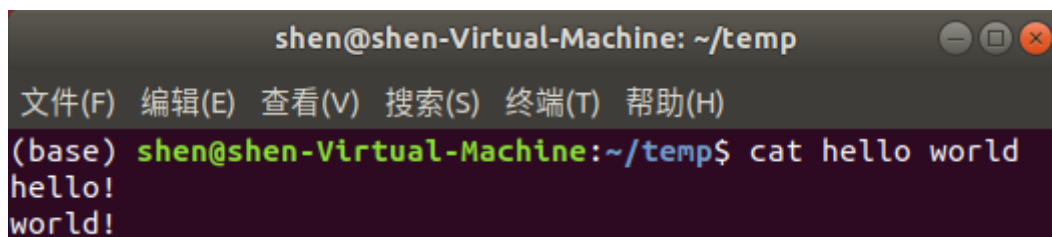
exist表示这个文件存在与否，此时linux会把标准错误发送到output1.txt文件中，而把标准输出发送到output2.txt文件中，而如果在最后再次把标准错误重定向到标准输出通道中，即在上面的命令中中再添加一个2>&1，此时错误和输出都会发送到output2.txt中。可以理解为linux线性执行命令，最后确定通道后，一起output。

有一种更简洁的方式来实现标准错误和标准输出的联合：

```
ls something &> output.txt
```

关于cat

cat 命令是concatenate(连接)的缩写，默认情况下，他接受一个文件作为参数，然后把这个文件的内容链接到标准输出上，同时接收多个文件作为参数时，可以将这些文件的内容连接到一起，输出到标准输出上：

A terminal window titled 'shen@shen-Virtual-Machine: ~/temp' with a menu bar (文件(F), 编辑(E), 查看(V), 搜索(S), 终端(T), 帮助(H)). The prompt is '(base) shen@shen-Virtual-Machine:~/temp\$'. The user enters 'cat hello world' and the terminal outputs 'hello!' followed by 'world!' on the next line.

```
shen@shen-Virtual-Machine: ~/temp
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
(base) shen@shen-Virtual-Machine:~/temp$ cat hello world
hello!
world!
```

当键入cat直接按回车时，系统会等待从标准输入获取输入，输出到标准输出上：

```
shen@shen-Virtual-Machine: ~/temp
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
(base) shen@shen-Virtual-Machine:~/temp$ cat
i type in : "hello world!"
i type in : "hello world!"
```

按下Ctrl-D，即告知cat命令已经达到了标准输入的文件尾巴(end-of-file, EOF)，结束

当然，也可以进行重定向，重定向标准输出或者标准输入

```
shen@shen-Virtual-Machine: ~/temp
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
(base) shen@shen-Virtual-Machine:~/temp$ cat > shen.txt
hello world!
i am a linux novice!
```

EOF与here文档

已经学习了`>`, `>>` 分别表示覆盖输出和追加输出，`<` 表示重定向输入，linux还有另外一种I/O重定向的方式，`<<`，称其为here文档，它的使用及功能如下所示：

```
command << token
```

此时，command命令会从标准输入中持续读取，直到读取匹配到了token，读取结束，这个用法通常在shell脚本中出现，用以向一个文件中追加多行内容，或者向一个程序传递一系列命令：

```
终端
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#!/bin/bash
# turbulence.sh
cat > object << EOF_token
Big whirls have little whirls
That feed on their velocity
And little whirls have lesser whirls
And so on to viscosity
EOF_token
~
~
```

运行这个shell脚本，得到如下结果：

```
打开(O)  object ~/temp  保存(S)
Big whirls have little whirls
That feed on their velocity
And little whirls have lesser whirls
And so on to viscosity
纯文本  制表符宽度: 8  第 3 行, 第 37 列  插入
```

