**shell 编程之 sed命令**

sed： stream editor

s e d是一个非交互性文本流编辑器。它编辑文件或标准输入导出的文本拷贝。标准输入可 能是来自键盘、文件重定向、字符串或变量，或者是一个管道的文本。

注意： s e d 并不与初始化文件打交道， 它操作的只是一个拷贝，然后所有的改动如果没有重定向到一个文件，将输出到屏幕。

处理逻辑：处理时，把当前处理的行存储在临时缓冲区中，称为“模式空间”（pattern space），接着用sed命令处理缓冲区中的内容，处理完成后，把缓冲区的内容送往屏幕。接着处理下一行，这样不断重复，直到文件末尾。文件内容并没有改变，除非你使用重定向存储输出。

调用sed有三种方式：在命令行键入命令；将 sed 命令插入脚本文件，然后调用 sed；将sed 命令插入脚本文件，并使 sed脚本可执行。

**sed命令格式**

sed [nefri] 'command' file(s)

**使用s e d脚本文件，格式为：**

sed [选项] -f sed 脚本文件 输入文件

要使用第一行具有 s e d命令解释器的s e d脚本文件，其格式为：

s e d脚本文件 [选项] 输入文件

注意：不管是使用 s h e l l 命令行方式或脚本文件方式，如果没有指定输入文件， s e d 从标准输入中 接受输入，一般是键盘或重定向结果。

常用选项:

-n　　使用安静（silent）模式。在一般sed的用法中，所有来自stdin的资料一般都会被列出到屏幕，但如果加上-n参数后，则只有经过sed特殊处理的那一行（或者command）才会被列出来。

-e　　允许多点编辑。

-f　　直接将sed的动作写在一个档案内，-f filename 则可以执行filename内的sed动作。

-r　　sed 的动作支援的是延伸型正规表示法的语法。(预设是基础正规表示法语法)

-i　　直接修改读取的档案内容，而不是由屏幕输出。

sed编辑命令:

|  |  |
| --- | --- |
| 命令 | 功能描述 |
|  | 常用command |
| a\ | 新增， a 的后面可以接字串，而这些字串会在新的一行出现(目前的下一行)～ |
| c\ | 取代， c 的后面可以接字串，这些字串可以取代 n1,n2 之间的行！ |
| d | 删除，因为是删除啊，所以 d 后面通常不接任何咚咚； |
| i\ | 插入， i 的后面可以接字串，而这些字串会在新的一行出现(目前的上一行)； |
| p | 列印，亦即将某个选择的资料印出。通常 p 会与参数 sed -n 一起运作～ |
| s | 取代，可以直接进行取代的工作哩！通常这个 s 的动作可以搭配正规！例如 1,20s/old/new/g 就是啦！ |
|  | 高级command |
| h | 拷贝pattern space的内容到holding buffer(特殊缓冲区)。 |
| H | 追加pattern space的内容到holding buffer。 |
| g | 获得holding buffer中的内容，并替代当前pattern space中的文本。 |
| G | 获得holding buffer中的内容，并追加到当前pattern space的后面。 |
| n | 读取下一个输入行，用下一个命令处理新的行而不是用第一个命令。 |
| P | 打印pattern space中的第一行。  //大写 |
| q | 退出sed。第一个模式匹配完成后退出或者立即退出 |
| r | 从另一个文件中读文本 |
| w file | 写并追加pattern space到file的末尾。 |
| ! | 表示后面的命令对所有没有被选定的行发生作用。 |
| s/re/string | 用string替换正则表达式re。 |
| = | 显示文件行号。 |
| l | 显示与八进制ASCII代码等价的控制字符 |
| {} | 在定位行执行的命令组 |
| 替换标记 |  |
| g | 行内全面替换，如果没有g，只替换第一个匹配。 |
| x | 互换pattern space和holding buffer中的文本。 |
| y | 把一个字符翻译为另一个字符(但是不能用于正则表达式)。 |

使用sed在文件中定位文本的方式：

x                 x为一行号，如 1

x , y             表示行号范围从x到y，如2，5表示从第2行到第5行

/pattern/     查询包含模式的行。例如 /disk/或/[a-z]/

/pattern/pattern/        查询包含两个模式的行。例如 /disk/disks/

/pattern/,x                  在给定行号上查询包含模式的行。如 /ribbon/ ,3

x,/pattern/                  通过行号和模式查询匹配行。3,/vdu/

x,y!                              查询不包含指定行号 x和y的行。1,2!

---------------------------------------

**基本sed编程举例：**

打印范围

eg:打印testfile.log文件的2到3行。 （如果后面的行号比前面的小，）

$ sed -n '2,3P' testfile.log

msmd 04 23 sdfrwt 99 23 45

ch 05 12 dgesdd 101 45 88

---------------------

使用模式和行号进行查询

eg: 在第三行查询模式sdd,进行匹配。

$ sed -n '3,/sdd/p' testfile.log

ch 05 12 dgesdd 101 45 88

注：这里'3,/sdd/p'和'3,/sdd/'p都行

---------------------

匹配元字符

匹配元字符 $前，必须使用反斜线 \屏蔽其特殊含义。模式为 /\$/ p 。

eg: 匹配 含有特殊字符$的行。

$ sed -n '/\$/'p testfile.log

345$1233

--------------------

显示整个文件 ：只需将行范围设为第一行到最后一行 1 , $。$意为最后一行。

$ sed -n '1，$/'p testfile.log

--------------------

任意字符

匹配任意字母，0次或多次重复，并以 i n g 结尾，模式为 /.\*ing/。可以使用 这个模式查询以ing结尾的任意单词。

eg：

$ echo "testing,tell me" >> testfile.log

$ sed -n '/.\*ing/'p testfile.log

testing,tell me

--------------------

打印行号

打印第一行  ：$ sed -n '1p' testfile.log

打印最后一行 ：$ sed -n '$p' testfile.log

要打印行号，使用等号 =。打印模式匹配的行号，使用格式 /pattern/=。

eg：$ sed -e '/ing/=' testfile.log

打印出所有行，只显示匹配行的行号。

eg：$ sed -n '/ing/=' testfile.log

只显示匹配行的行号。

如果打印行号及匹配行，必须使用两个sed命令，并使用e选项。第一个命令打印模式 匹配行，第二个使用=选项打印行号，格式为 sed -n -e /pattern/p -e /pattern/=。

eg: 显示匹配ing的行及行号。

$ sed -n -e '/ing/p' -e '/ing/=' testfile.log

testing,tell me

7

---------------------------------------------------

删除文本

sed删除文本格式： [ address [，address ] ] d

删除第一行：$ sed '1d' testfile.log

删除最后一行：$ sed '$d' testfile.log

删除匹配ing的一行：$ sed '/ing/d' testfile.log

--------------------------------------------------

替换文本

替换命令用替换模式替换指定模式，格式为：

[address [，address]] s/ pattern-to-find /replacement-pattern/[g p w n]

s 选项通知 s e d这是一个替换操作，并查询 pattern - to - find，成功后用replacement - pattern替 换它。

替换选项如下：

g 缺省情况下只替换第一次出现模式，使用 g选项替换全局所有出现模式。

p 缺省s e d 将所有被替换行写入标准输出，加 p选项将使 - n选项无效。 - n选项不打印输出

结果。

w 文件名 使用此选项将输出定向到一个文件。

eg:

$ sed 's/msmd/SSS/' testfile.log     把 msmd  替换为SSS

$ sed 's/\$//' testfile.log          删除$符号，用空白替换掉$即可。

缺省是替换一次出现模式

$ sed 's/h/HH/' testfile.log

test HHh 1234 && 8762 wrq

加上参数g,全局替换

$ sed 's/h/HH/g' testfile.log

test HHHH 1234 && 8762 wrq

将替换结果写入一个文件用 w选项，下面的例子将h替换为HH的替换结果 写入文件replace.out中：

$ sed 's/h/HH/w replace.out' testfile.log

-----------------------------------------

使用替换修改字符串

如果要附加或修改一个字符串，可以使用（ &）命令， &命令保存发现模式以便重新调用 它，然后把它放在替换字符串里面。这里给出一个修改的设计思路。先给出一个被替换模式， 然后是一个准备附加在第一个模式后的另一个模式，并且后面带有 &，这样修改模式将放在 匹配模式之前。

eg: 在msmd前面加上HELLO。

 $ sed -n 's/msmd/"HELLO" &/p' testfile.log

"HELLO" msmd 04 23 sdfrwt 99 23 45

原句为：msmd 04 23 sdfrwt 99 23 45

---------------------------------------------

将sed结果写入文件命令

像使用 >文件重定向发送输出到一个文件一样，在sed命令中也可以将结果输入文件。格 式有点像使用替换命令：

[address [，address]]w filename        ‘w’选项通知sed将结果写入文件。filename是自解释文件名。

eg:

$ sed 'w file.txt' testfile.log         把testfile.log内容写入到file.txt文件中。

$ sed '/ms/w file.txt' testfile.log         把testfile.log文件中匹配ms的内容写入到file.txt文件中。

------------------------------------------

从文件中读文本

处理文件时， s e d允许从另一个文件中读文本，并将其文本附加在当前文件。此命令放在 模式匹配行后，格式为：

address r filename            这里r通知s e d将从另一个文件源中读文本。 f i l e n a m e是其文件名。

eg:

$ sed '/ms/r append.sed' testfile.log    在匹配到ms后，读取append.sed文件的内容。

$ sed 'r append.sed' testfile.log          在testfile.log的每一行后，读取append.sed文件的内容。

-------------------------

匹配后退出

有时需要在模式匹配首次出现后退出sed，以便执行其他处理脚本。退出命令格式为：

address q

eg:

$ sed '/h/q' testfile.log

msmd 04 23 sdfrwt 99 23 45

ch 05 12 dgesdd 101 45 88

匹配到h后，不再匹配，选择退出

-----------------------------

显示文件中的控制字符

当从其他系统下载文件时，有时要删除整个文件的控制字符（非打印字符） ，从菜单中捕 获一个应用的屏幕输出有时也会将控制字符输出进文件，怎样知道文件中是否有控制字符？

使用cat -v filename 命令，屏幕会乱叫，且到处都是一些垃圾字符，这可以确知文件中包含有 控制字符，如果有兴趣可以观察一下这些字符以便于更加确认它们是控制字符。

一些系统中使用cat filename 而不是c a t - v来查看非打印字符。

s e d格式为：

[address，[ address ] ] l

‘l’意为列表。

一般情况下要列出整个文件，而不是模式匹配行，因此使用 l 要从第一到最后一行。模式 范围1，$即为此意。

如果c a t一个文件，发现实际上包含有控制字符。

现在运行s e d命令，观察输出结果。

$ sed -n '1,$l' append.sed           --前面是数字1，后面是字母l.

$

#!/bin/sed -f$

/company/ a\\$

Then suddenly it happened.$

$

\033OP$

s e d找到并显示了两个控制字符.\033代表退格键， O P为F1键值，放在退格键后。

各系统控制字符键值可能不同，主要取决于其映射方式（例如使用 terminfo 或 termcap） 。

如果要在文本文件中插入控制字符 F1键，使用vi查看其键值，操作如下：

 启动v i。

 进入插入模式。

 按下< C t r l >键，然后按 < v >键 （出现a ^） 。

 释放上述两个键。

 按下F 1键（显示[ O P ] 。

 按下< E S C >键（显示F 1键值）