



求知

通过几个例子看sed的模式空间与保持空间

SED之所以能以行为单位的编辑或修改文本，其原因在于它使用了两个空间：一个是活动的“模式空间（pattern space）”，另一个是起辅助作用的“暂存缓冲区（holdingspace）这2个空间的使用。

sed编辑器逐行处理文件，并将输出结果打印到屏幕上。sed命令将当前处理的行读入模式空间（pattern space）进行处理，sed在该行上执行完所有命令后就将处理好的行打印到屏幕上（除非之前的命令删除了该行），sed处理完一行就将其从模式空间中删除，然后将下一行读入模式空间，进行处理、显示。处理完文件的最后一行，sed便结束运行。sed在临时缓冲区（模式空间）对文件进行处理，所以不会修改原文件，除非显示指明-i选项。

sed之所以能以行为单位的编辑或修改文本，其原因在于它使用了两个空间：一个是活动的“模式空间（pattern space）”，另一个是起辅助作用的“保持空间（hold space）这2个空间的使用。

模式空间：可以想成工程里面的流水线，数据之间在它上面进行处理，用于处理文本行。

保持空间：可以想象成仓库，我们在进行数据处理的时候，作为数据的暂存区域，用于保留文本行，是保存已经处理过的输入行，默认有一个空行。

与模式空间和暂存空间（hold space）相关的命令：

导航

- 博客园
- 首页
- 新随笔
- 联系
- 订阅 XML
- 管理

公告

昵称：生活费  
园龄：8年11个月  
粉丝：68  
关注：4  
[+加关注](#)

< 2019年5月 >						
日	一	二	三	四	五	六
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

统计

- 随笔 - 301
- 文章 - 0
- 评论 - 10
- 引用 - 0

搜索



n 输出模式空间行，读取下一行替换当前模式空间的行，执行下一条处理命令而非第一条命令。

N 读入下一行，追加到模式空间行后面，此时模式空间有两行。

h 把模式空间的内容复制到保留空间，覆盖模式

H 把模式空间的内容追加到保留空间，追加模式

g 把保留空间的内容复制到模式空间，覆盖模式

G 把保留空间的内容追加到模式空间，追加模式

x 将暂存空间的内容于模式空间里的当前行互换。

! 对所选行以外的所有行应用命令。

注意：暂存空间里默认存储一个空行。

 找找看 谷歌搜索

### 常用链接

[我的随笔](#)[我的评论](#)[我的参与](#)[最新评论](#)[我的标签](#)

### 我的标签

[angular2\(1\)](#)[linux系统\(1\)](#)

### 随笔分类

[cdn\(1\)](#)[CI\(5\)](#)[dns](#)[go\(16\)](#)[ha,nginx,lvs\(10\)](#)[java\(1\)](#)[python\(51\)](#)[redis,mysql,mq\(22\)](#)[ruby\(6\)](#)[shell\(49\)](#)[tomcat\(2\)](#)[web,js,html\(10\)](#)[安全\(5\)](#)[大数据\(4\)](#)[调试\(3\)](#)[服务调优\(4\)](#)[管理协作\(5\)](#)[技巧\(25\)](#)[监控\(20\)](#)[进程管理\(1\)](#)[开源好项目\(1\)](#)



## 例子一

sed G

在文件每一行下面输出一个空行

### 代码:

```
$ cat foo
111111111111111
22222222222222
33333333333333
444444444444444
55555555555555

$ sed G foo
111111111111111
22222222222222
33333333333333
444444444444444
```

[容器,虚拟化\(22\)](#)

[网络\(19\)](#)

[网络编程\(2\)](#)

[系统\(60\)](#)

[正则\(1\)](#)

[自动化运维\(3\)](#)

### 随笔档案

[2019年4月 \(3\)](#)

[2019年3月 \(4\)](#)

[2019年2月 \(7\)](#)

[2019年1月 \(1\)](#)

[2018年12月 \(2\)](#)

[2018年11月 \(6\)](#)

[2018年10月 \(1\)](#)

[2018年9月 \(8\)](#)

[2018年8月 \(5\)](#)

[2018年7月 \(2\)](#)

[2018年5月 \(5\)](#)

[2018年4月 \(5\)](#)

[2018年3月 \(6\)](#)

[2018年2月 \(1\)](#)

[2018年1月 \(5\)](#)

[2017年12月 \(10\)](#)

[2017年11月 \(7\)](#)

[2017年10月 \(4\)](#)

[2017年9月 \(4\)](#)

[2017年8月 \(3\)](#)

[2017年7月 \(6\)](#)

[2017年6月 \(11\)](#)

[2017年4月 \(1\)](#)

[2017年3月 \(4\)](#)

[2017年2月 \(8\)](#)

[2017年1月 \(5\)](#)

[2016年12月 \(3\)](#)



```
55555555555555
```

解释：

sed 中 G 的用法

The G function appends the contents of the holding area to the contents of the pattern space. The former and new contents are separated by a newline. The maximum number of addresses is two.

hold space：保持空间（或者叫保留空间、缓冲区），初始为空

pattern space：模式空间

在上面的例子中，将为空的hold space附加到文件的每一行后面，所以结果是每一行后面多了一个空行

引申出：

```
sed '/^$/d;G'
```

在文件的每一个非空行下面输出一个空行

```
sed '/^$/d;G;G'
```

[2016年11月 \(6\)](#)[2016年10月 \(4\)](#)[2016年9月 \(12\)](#)[2016年8月 \(6\)](#)[2016年7月 \(9\)](#)[2016年6月 \(15\)](#)[2016年5月 \(13\)](#)[2016年4月 \(3\)](#)[2016年3月 \(4\)](#)[2016年2月 \(1\)](#)[2016年1月 \(1\)](#)[2015年9月 \(1\)](#)[2015年8月 \(1\)](#)[2015年7月 \(1\)](#)[2015年5月 \(2\)](#)[2015年4月 \(7\)](#)[2015年3月 \(2\)](#)[2015年1月 \(3\)](#)[2014年12月 \(3\)](#)[2014年11月 \(1\)](#)[2014年9月 \(1\)](#)[2014年8月 \(1\)](#)[2014年7月 \(2\)](#)[2014年6月 \(1\)](#)[2014年5月 \(1\)](#)[2014年2月 \(1\)](#)[2014年1月 \(1\)](#)[2013年12月 \(1\)](#)[2013年11月 \(1\)](#)[2013年10月 \(4\)](#)[2013年8月 \(1\)](#)[2013年4月 \(1\)](#)[2013年3月 \(1\)](#)[2012年10月 \(1\)](#)[2012年7月 \(3\)](#)



## 在文件的每一个非空行下面输出两个空行

### 代码:

```
$ cat foo
```

```
1111111111111111
```

```
2222222222222222
```

```
3333333333333333
```

```
4444444444444444
```

```
5555555555555555
```

```
$ sed '/^$/d;G' foo
```

```
1111111111111111
```

```
2222222222222222
```

```
3333333333333333
```

```
4444444444444444
```

[2012年6月 \(3\)](#)

[2012年5月 \(3\)](#)

[2012年4月 \(2\)](#)

[2012年3月 \(1\)](#)

[2012年1月 \(1\)](#)

[2011年12月 \(6\)](#)

[2011年11月 \(1\)](#)

[2011年10月 \(8\)](#)

[2011年9月 \(4\)](#)

[2011年8月 \(1\)](#)

[2011年5月 \(8\)](#)

[2011年1月 \(1\)](#)

[2010年11月 \(2\)](#)

[2010年10月 \(6\)](#)

[2010年9月 \(7\)](#)

[2010年8月 \(2\)](#)

[2010年7月 \(2\)](#)

### 最新评论

[1. Re:python + django + dwebsocket 实现简单的聊天室](#)

请问下博主如何知晓用户退出登录

--11212

[2. Re:linux上ssh免密登录原理及实现](#)

很好的学习资料，谢谢分享

--B.Chiang

[3. Re:你见过的最全面的python重点](#)

hi 关注你的博客挺久了，可以加微信交流一下吗？

--laixintao

[4. Re:Linux 内存中的Cache，真的能被回收么？](#)

写的很棒~

--苦头陀

[5. Re:ntp服务的细节全解析](#)

ntpd -x OPTIONS 的效果 怎么查看，同步也是直接使用`ntpddate -x 10.10.10.1 ;/usr/sbin/hwclock -



```
55555555555555
```

注：有时会有一些由空格符或者TAB组成的空行，前面的正则式 `^$` 就不能匹配到这样的行，则可以这样

```
sed '/[[:space:]]/d;G'
```

## 例子二

```
sed '/regex/{x;p;x;}'
```

在匹配regex的所有行前面插入一个空行

### 代码:

```
$ cat foo
11111111111111
22222222222222
test333333333333
44444444444444
55555555555555

$ sed '/test/{x;p;x;}' foo
```

w" ?

--tengfei520

## 阅读排行榜

1. [shell中if做比较\(162015\)](#)
2. [Linux下rz, sz与ssh的配合使用\(45565\)](#)
3. [docker使用问题总结\(30119\)](#)
4. [Redis基础、高级特性与性能调优\(27010\)](#)
5. [Linux的shell中echo改变输出显示样式\(20781\)](#)

## 评论排行榜

1. [ubuntu安装配置指南\(1\)](#)
2. [中断处理中tasklet与工作队列的使用\(1\)](#)
3. [大前端工具集\(1\)](#)
4. [Linux 内存中的Cache, 真的能被回收么? \(1\)](#)
5. [Linux下查看Raid磁盘阵列信息的方法\(1\)](#)

## 推荐排行榜

1. [shell中if做比较\(7\)](#)
2. [ubuntu下配置vim\(4\)](#)
3. [Linux下rz, sz与ssh的配合使用\(4\)](#)
4. [大前端工具集\(3\)](#)
5. [Redis基础、高级特性与性能调优\(3\)](#)

Powered by:  
[博客园](#)  
Copyright © 生活费



```
11111111111111
```

```
22222222222222
```

```
test33333333333333
```

```
4444444444444444
```

```
55555555555555
```

解释：

sed 中 x 的用法

The exchange function interchanges the contents of the pattern space and the holding area. The maximum number of addresses is two.

即交换保持空间hold space和模式空间pattern space的内容


sed 中 p 的作用是把模式空间复制到标准输出。

分析一下该命令执行过程中保持空间和模式空间的内容

命令 保持空间 模式空间

x 执行前:null 执行后:test\n 执行前:test\n 执行后:null

p 执行前:null 执行后:test\n 执行前:test\n 执行后:null 输出一个空行

 x 执行前:test\n 执行后:null 执行前:null 执行后:test\n

(注：把test所在的行简写为test了)

引申：

可以试验一下 `sed '/test/{x;p;}' foo` 或者 `sed '/test/{p;x;}' foo` 等，看看结果，体会两个空间的变化

相应的：

`sed '/regex/G'` 是在匹配regex的所有行下面输出一个空行

`sed '/regex/{x;p;x;G;}'` 是在匹配regex的所有行前面和下面都输出一个空行

例子三

`sed 'n;G;'`

在文件的偶数行下面插入一个空行

**代码：**

```
$ cat foo
```

```
11111111111111
```





```
22222222222222
```

```
33333333333333
```

```
44444444444444
```

```
55555555555555
```

```
$ sed 'n;G;' foo
```

```
11111111111111
```

```
22222222222222
```

```
33333333333333
```

```
44444444444444
```

```
55555555555555
```

解释：

sed 中 n 的用法：将模式空间拷贝于标准输出。用输入的下一行替换模式空间。

执行 n 以后将第一行输出到标准输出以后，然后第二行进入模式空间，根据前面对 G 的解释，会在第二行后面插入一个空行，然后输出；再执行 n 将第三行输出到标准输出，然后第四行进入模式空间，并插入空行，依此类推



相应的:

sed 'n;n;G' 表示在文件的第 3,6,9,12,... 行后面插入一个空行

sed 'n;n;n;G' 表示在文件的第 4,8,12,16,... 行后面插入一个空行

sed 'n;d' 表示删除文件的偶数行

#### 例子四

sed '\$!N;\$!D'

输出文件最后2行, 相当于 tail -2 foo

#### 代码:

```
$ cat foo
```

```
111111111111111
```

```
22222222222222
```

```
33333333333333
```

```
44444444444444
```

```
55555555555555
```

```
$ sed '$!N;$!D' foo
```

```
44444444444444
```



```
555555555555555
```

解释：

D 删除模式空间内第一个 newline 字母 \n 前的资料。

N 把输入的下一行添加到模式空间中。

sed '\$!N;\$!D'：对文件倒数第二行以前的行来说，N 将当前行的下一行放到模式空间中以后，D 就将模式空间的内容删除了；到倒数第二行的时候，将最后一行附加到倒数第二行下面，然后最后一行不执行 D，所以文件的最后两行都保存下来了。

还有 N 的另外一种用法

**代码：**

```
$ sed = foo | sed N
1
111111111111111
2
```



```
22222222222222
```

```
3
```

```
33333333333333
```

```
4
```

```
44444444444444
```

```
5
```

```
55555555555555
```

```
$ sed = foo | sed 'N;s/\n/ /'
```

```
1      11111111111111
```

```
2      22222222222222
```

```
3      33333333333333
```

```
4      44444444444444
```

```
5      55555555555555
```

解释：

N 的作用是加上行号，可以用于格式化输出文件

例子五

```
sed '1!G;h;$!d'
```

```
sed -n '1!G;h;$p'
```



将文件的行反序显示，相当于 tac 命令(有些平台没有这个命令)

**代码:**

```
$ cat foo
```

```
111111111111111
```

```
222222222222222
```

```
333333333333333
```

```
$ sed '1!G;h;$!d' foo
```

```
333333333333333
```

```
222222222222222
```

```
111111111111111
```

```
$ sed -n '1!G;h;$p' foo
```

```
333333333333333
```

```
222222222222222
```

```
111111111111111
```

**解释:**

sed 中 h 用法: h



The h (hold) function copies the contents of the pattern space into a holding area, destroying any previous contents of the holding area.

意思是将模式空间的内容保存到保持空间中去

sed 中的 d 表示删除模式空间。

1!G表示除了第一行以外，其余行都执行G命令；\$!d表示除了最后一行以外，其余行都执行d命令。

看一下sed '1!G;h;\$!d'命令执行过程中保持空间与模式空间的变化：

命令 保持空间 模式空间

第一行 h;d 执行前:null 执行后:1111\n 执行前:1111\n 执行后:null

第二行 G;h;d 执行前:1111 执行后:2222\n1111\n 执行前:2222\n 执行后:null

第二行 G;h 执行前:2222\n1111\n 执行后:3333\n2222\n\n1111\n 执行前:3333\n 执行后:3333\n2222\n\n1111\n

(注：把各个行简写了)

这样输出以后就是文件的反序了。



题外话：在vi中对一个文件进行反序显示的命令是 :g/./m0，意思是按照文件正常顺序每找到一行，就把该行放到文件的最上面一行去，这样循环一下正好把文件的行反序显示了。

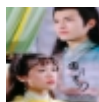
[阅读原文](#)

分类: [shell](#)

好文要顶

关注我

收藏该文



生活费

关注 - 4

粉丝 - 68

0

0

[+加关注](#)

« 上一篇: [python常用运维脚本实例](#)

» 下一篇: [Python运维自动化开发之Fabric模块](#)

posted on 2017-11-22 16:02 [生活费](#) 阅读(808) 评论(0) [编辑](#) [收藏](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#)网站首页。

**【推荐】** [超50万C++/C#源码：大型实时仿真组态图形源码](#)

**相关博文：**

- [内核空间与用户空间](#)
- [sed原理及sed命令格式,缓存区,模式空间](#)
- [今天看音的空间~~~](#)
- [MultiView空间例子](#)
- [Linux 内核空间与用户空间](#)

**最新新闻：**

- [西藏洞穴发现丹尼索瓦人化石](#)
- [中国沙漠惊现黑科技：万镜追日场面壮观](#)



- [研究证实年初的超级血狼月期间有流星撞击了月球](#)
- [Windows 10安全功能拖累Chromium浏览器：性能损失五倍](#)
- [巨头裁员风波不停，下沉互联网正C位出道](#)
- » [更多新闻...](#)