# Shell

交互式shell和非交互式shell?

交互式模式就是shell等待你的输入，并且执行你提交的命令。这种模式被称作交互式是因为shell与用户进行交互。这种模式也是大多数用户非常熟悉的：登录、执行一些命令、签退。当你签退后，shell也终止了。 shell也可以运行在另外一种模式：非交互式模式。在这种模式下，shell不与你进行交互，而是读取存放在文件中的命令,并且执行它们。当它读到文件的结尾，shell也就终止了。

什么是login shell 和non-login shell?

Bash应该是我们每天日常工作接触最多的东西了，就像我们最忠实的朋友，我们有必要了解一下这位朋友的“习性”。  
Bash有几种不同的运行模式，login shell与non-login shell，interactive shell与non-interactive shell（比如执行shell脚本）。这两种分类方法是交叉的，也就是说一个login shell可能是一个interactive shell，也可能是个non-interactive shell。  
在下列情况下，我们可以获得一个login shell：

1. 登录系统时获得的顶层shell，无论是通过本地终端登录，还是通过网络ssh登录。这种情况下获得的login shell是一个交互式shell。
2. 在终端下使用--login选项调用bash，可以获得一个交互式login shell。
3. 在脚本中使用--login选项调用bash（比如在shell脚本第一行做如下指定：#!/bin/bash --login），此时得到一个非交互式的login shell。
4. 使用"su -"切换到指定用户时，获得此用户的login shell。如果不使用"-"，则获得non-login shell。

login shell与non-login shell的主要区别在于它们启动时会读取不同的配置文件，从而导致环境不一样。login shell启动时首先读取/etc/profile全局配置，然后依次查找~/.bash\_profile、~/.bash\_login、~/.profile三个配置文件，并且读取第一个找到的并且可读的文件。login shell退出时读取并执行~/.bash\_logout中的命令。  
交互式的non-login shell启动时读取~/.bashrc资源文件。非交互式的non-login shell不读取上述所有配置文件，而是查找环境变量BASH\_ENV，读取并执行BASH\_ENV指向的文件中的命令。  
如果使用命令"sh"调用bash，bash会尽可能保持向后兼容。作为login shell启动时，bash依次读取/etc/profile和~/.profile配置文件。作为non-login shell启动时，bash读取环境变量ENV指向的文件。  
通常我们要定制一些配置时，将配置写在~/.bashrc中，然后在~/.bash\_profile中读取~/.bashrc，这样可以保证login shell和交互式non-login shell得到相同的配置。至于/etc/profile就不要轻易去改啦，毕竟会影响系统全局的配置。

# bashrc与profile的区别

要搞清bashrc与profile的区别，首先要弄明白什么是交互式shell和非交互式shell，什么是login shell 和non-login shell。

交互式模式就是shell等待你的输入，并且执行你提交的命令。这种模式被称作交互式是因为shell与用户进行交互。这种模式也是大多数用户非常熟悉的：登录、执行一些命令、签退。当你签退后，shell也终止了。 shell也可以运行在另外一种模式：非交互式模式。在这种模式下，shell不与你进行交互，而是读取存放在文件中的命令,并且执行它们。当它读到文件的结尾，shell也就终止了。

bashrc与profile都用于保存用户的环境信息，bashrc用于交互式non-loginshell，而profile用于交互式login shell。系统中存在许多bashrc和profile文件，下面逐一介绍：

/etc/pro此文件为系统的每个用户设置环境信息,当第一个用户登录时,该文件被执行.

并从/etc/profile.d目录的配置文件中搜集shell的设置.

/etc/bashrc:为每一个运行bash shell的用户执行此文件.当bash shell被打开时,该文件被读取。有些linux版本中的/etc目录下已经没有了bashrc文件。

~/. pro每个用户都可使用该文件输入专用于自己使用的shell信息,当用户登录时,该文件仅仅执行一次!默认情况下,它设置一些环境变量,然后执行用户的.bashrc文件.

~/.bashrc:该文件包含专用于某个用户的bash shell的bash信息,当该用户登录时以及每次打开新的shell时,该文件被读取.

另外,/etc/profile中设定的变量(全局)的可以作用于任何用户,而~/.bashrc等中设定的变量(局部)只能继承/etc/profile中的变量,他们是"父子"关系.

## 某网友总结如下：

/etc/profile，/etc/bashrc 是系统全局环境变量设定  
~/.profile，~/.bashrc用户家目录下的私有环境变量设定  
当登入系统时候获得一个shell进程时，其读取环境设定档有三步  
1首先读入的是全局环境变量设定档/etc/profile，然后根据其内容读取额外的设定的文档，如  
/etc/profile.d和/etc/inputrc  
2然后根据不同使用者帐号，去其家目录读取~/.bash\_profile，如果这读取不了就读取~/.bash\_login，这个也读取不了才会读取  
~/.profile，这三个文档设定基本上是一样的，读取有优先关系  
3然后在根据用户帐号读取~/.bashrc  
至于~/.profile与~/.bashrc的不区别  
都具有个性化定制功能  
~/.profile可以设定本用户专有的路径，环境变量，等，它只能登入的时候执行一次  
~/.bashrc也是某用户专有设定文档，可以设定路径，命令别名，每次shell script的执行都会使用它一次

# linux下配置jdk环境变量的三种方法

一、修改/etc/profile文件当本机仅仅作为开发使用时推荐使用这种方法，因为此种配置时所有用户的shell都有权使用这些环境变量，可能会给系统带来安全性问题。用文本编辑器打开/etc/profile，在profile文件末尾加入：JAVA\_HOME=/usr/share/jdk1.5.0\_05

PATH=$JAVA\_HOME/bin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME

export PATH

export CLASSPATH重新登录即可。

二、修改.bashrc文件这种方法更为安全，它可以把使用这些环境变量的权限控制到用户级别，如果需要给某个用户权限使用这些环境变量，只需要修改其个人用户主目录下的.bashrc文件就可以了。用文本编辑器打开用户目录下的.bashrc文件，在.bashrc文件末尾加入：

set JAVA\_HOME=/usr/share/jdk1.5.0\_05

export JAVA\_HOME

set PATH=$JAVA\_HOME/bin:$PATH

export PATH

set CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export CLASSPATH重新登录。

三、直接在shell下设置变量不推荐使用这种方法，因为换个shell，该设置就无效了。这种方法仅仅是临时使用，以后要使用的时候又要重新设置，比较麻烦。只需在shell终端执行下列命令：

export JAVA\_HOME=/usr/share/jdk1.5.0\_05

export PATH=$JAVA\_HOME/bin:$PATH

export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

注意：

1.要将 /usr/share/jdk1.5.0\_05jdk 改为jdk安装目录

2. linux下用冒号”:”来分隔路径

3. $PATH / $CLASSPATH / $JAVA\_HOME 是用来引用原来的环境变量的值在设置环境变量时特别要注意不能把原来的值给覆盖掉了。

4. CLASSPATH中当前目录”.”不能丢掉。

5. export是把这三个变量导出为全局变量。

6. 大小写必须严格区分。

7. 一般写成下面就可以了:

export JAVA\_HOME=/java/jdk1.8.0\_161

export CLASSPATH=.:$JAVA\_HOME/lib:$CLASSPATH

export PATH=$JAVA\_HOME/bin:$PATH

# 操作命令

## 查找文件

cd 切换工作目录； clear 清屏 windows 是 cls

cd ~或cd 切换到当前用户的主目录(/home/用户目录)

cd . 切换当前目录 配合复制和剪切文件用

cd .. 切换到上级目录

cd – 回到上一次的目录

ls “.”代表当前目录，“..”代表上一级目录，Linux文件、目录名称最长可以有265个字符

-a 显示指定目录下所有子目录与文件，包括隐藏文件

-l 以列表方式显示文件的详细信息

-h 配合 -l 以人性化的方式显示文件大小

\*文件代表文件名中所有字符 ？代表文件名中任意一个字符

ls –alh 命令后显示内容：

drwxrwxr-x 2 python python 4.0K 7月 30 2016 dbs

drwxrwxr：d代表文件夹、后面代表权限 ，2代表此文件有两个相同内容的文件 python代表用户

-rw-rw-r-- 1 python python 18 7月 30 2016 dump.rdb

ls –alh > j.txt (>:重定向)将显示在终端的内容重定向到j.txt文件中

ls \\*a查找当前的以 \*a的文件

ls [a-f]\*找到从a-f的范围内的任意一个字符

ls –help 显示帮助命令

pwd 打印当前路径 tree 以目录树方式显示

python@ubuntu:~/Documents$ pwd

/home/python/Documents 显示当前路径

python@ubuntu:~/Documents$ tree 当前Documents下以目录树方式显示所有

.

├── a

│   └── b

│   └── a

│   └── d

│   └── c

├── a.txt

├── b

│   └── a

│   └── d

│   └── c

find通常用来在特定的目录下搜索符合条件的文件

find ./ -name test.sh 查找当前目录下所有名为test.sh的文件

find ./ -name '\*.sh' 查找当前目录下所有后缀为.sh的文件

find ./ -name "[A-Z]\*" 查找当前目录下所有以大写字母开头的文件

find /tmp -size 2M 查找在/tmp 目录下等于2M的文件

find /tmp -size +2M 查找在/tmp 目录下大于2M的文件

find /tmp -size -2M 查找在/tmp 目录下小于2M的文件

find ./ -size +4k -size -5M 查找当前目录下大于4k，小于5M的文件

find ./ -perm 0777 查找当前目录下权限为 777 的文件或目录

ln 建立快捷方式 ln 源文件 链接文件

ln软链接 ：就是windows一样快捷方式，修改了等于修改了主文件

python@ubuntu:~/Documents$ ln -s a.txt cc 建立快捷方式

python@ubuntu:~/Documents$ ls –lh

-rw-rw-r-- 1 python python 13 7月 7 11:05 a.txt

lrwxrwxrwx 1 python python 5 7月 7 14:15 cc -> a.txt cc指向了a.txt

ln -s硬链接：同文件：有两个文件名以上、修改同时修改，删除一个对另一个没有影响

python@ubuntu:~/Documents$ ln b.txt dd 建立快捷方式硬链接

python@ubuntu:~/Documents$ ls -lh

-rw-rw-r-- 2 python python 35 7月 7 14:06 b.txt 显示一样的内容：2代表有两个名称

-rw-rw-r-- 2 python python 35 7月 7 14:06 dd

which 查看此文件的目标文件在哪里

python@ubuntu:~/Documents$ which python 查找python在哪里

/usr/bin/python

du -h查看此目录下所有文件的大小

## 目录、文件操作

创建

mkdir 创建目录

mkdir –p:递归创建文件目录

python@ubuntu:~/Documents$ mkdir a/b/c/d/e –p

touch 创建文件

python@ubuntu:~/Documents$ touch bb.txt

删除

rm 删除目录和名字

-I 有提示语问你删除不删除

无参数或-f 强制删除，忽略不存在的文件，不提示

-r 递归的删除目录下的内容，删除文件夹时必须加此参数

python@ubuntu:~/Documents$ rm a 未加 –r不能删除文件夹

rm: 无法删除'a': 是一个目录

python@ubuntu:~/Documents$ rm a –r 加了成功删除a文件夹下的所有文件和文件夹

拷贝

cp 是将给出的文件或目录复制到另一个文件或目录中，相当于DOS下的copy命令。

-a 通常在复制目录时使用，它保留链接、文件属性，并递归地复制目录，简单而言，保持文件原有属性。

-f 已经存在的目标文件而不提示

-i 在覆盖目标文件之前将给出提示要求用户确认

-r 复制文件目录：则cp将递归复制该目录下的所有子目录和文件

-v 显示拷贝进度

python@ubuntu:~/Documents$ cp b a 把b目录复制个a目录中

cp: 略过目录'b' 没有使用参数 –r 提示语

python@ubuntu:~/Documents$ cp b a –r 使用-r成功复制

剪切

mv mv命令来移动文件或目录，也可以给文件或目录重命名。

-f 禁止交互式操作，如有覆盖也不会给出提示

-i 如果mv操作将对已存在的目标文件的覆盖，系统会询问是否重写，要求用户回答以避免误覆盖文件

-v 显示移动进度

## 查看文件内容

more 用于内容多的话分页查看内容

python@ubuntu:~/Documents$ more a.txt

cat 查看或合并文件内容

python@ubuntu:~/Documents$ cat a.txt b.txt > c.txt 合并a和b中的内容至c.txt

python@ubuntu:~/Documents$ cat c.txt 查看c.txt

grep 搜索文件中内容 通过条件进行显示，cat是显示所有内容

-v 显示不包含匹配文本的所有行（相当于求反）

-n 显示匹配行及行号

-i 忽略大小写

python@ubuntu:~/Documents$ cat a.txt 打印此文本a.txt所有内容

lksjdf

sdf

python@ubuntu:~/Documents$ grep -n '^s' a.txt 匹配s开头的行，使用正则

4:sdf 行号第4号

## 归档管理

进行打包又压缩方式一次性完成

一次性进行打包又压缩成dd -zcvf

python@ubuntu:~/Documents$ tar -zcvf dd.tar.gz \*

python@ubuntu:~/Documents$ ls -lh

-rw-rw-r-- 1 python python 690 7月 7 15:21 dd.tar.gz

一次性进行解压成文件 -zxvf

python@ubuntu:~/Documents$ tar -zxvf dd.tar.gz

tar 只是进行打包 并没有进行压缩

-c 生成档案文件，创建打包文件

-v 列出归档解档的详细过程，显示进度

-f 指定档案文件名称，f后面一定是.tar文件，所以必须放选项最后

-t 列出档案中包含的文件

-x 解开档案文件

把当前中所有的文件打包成tar

python@ubuntu:~/Documents$ tar -cvf xixi.tar \*

把当前tar进行解包

python@ubuntu:~/Documents$ tar -xvf xixi.tar

gzip 对打包后tar文件进行压缩 变成后缀 gz

压缩

python@ubuntu:~/Documents$ gzip xixi.tar 压缩打包好的tar

python@ubuntu:~/Documents$ ls

a a.txt b bb.txt b.txt c cc c.txt dd n n.txt –p s.py xixi.tar.gz

B --- 字节 K– 千字节 1024b M—1024k

解压缩：加一个参数 –d ：注意仅仅是解压成tar还需解包

python@ubuntu:~/Documents$ gzip -d xixi.tar.gz

python@ubuntu:~/Documents$ ls -lh

-rw-rw-r-- 1 python python 20K 7月 7 15:00 xixi.tar 注意仅仅是解压成tar还需解包

注意：Linux压缩常用两种方式：xxx.tar.bz2 和xxx.tar.gz

打包方式 tar -zcvf dd.tar.gz \* tar -jcvf dd.tar.gz \*

zip 这是windows常用的，Linux也支持

压缩成zip ：zip

python@ubuntu:~/Documents$ zip n \*

adding: ni.txt (stored 0%)

python@ubuntu:~/Documents$ ls

ni.txt n.zip

解压操作 unzip –d

python@ubuntu:~/Documents$ unzip -d ./ n.zip

## 远程登录

第一 测试网络是否想通：全部系统通用命令：ping ip

ping 10.10.0.187

第二 查看ip地址命令：ifconfig windows系统:ipconfig

python@ubuntu:~/Desktop/apache-tomcat-8.5.29/bin$ ifconfig

ens33 Link encap:以太网 硬件地址 00:0c:29:d4:ca:59

命令ifconfig ens33 down关闭网卡 up开启网卡 还可以修改ip：ifconfig ens33 ip

lo Link encap:本地环回

第三 远程登录 ssh: ubuntu默认安装了ssh，windows需安装ssh软件进行path环境

ssh python@192.168.3.3 //用户名和ip 即可后续会让你输入密码，然后就会进入终端命令

第四 who：查看此系统有用户登录 (linux是多用户 ，windows是单用户！)

还有一些参数 -q

python@ubuntu:~$ who -q

python python

# 用户数=2

whoami 查看当前登录用户

## 创建用户操作

创建用户 一个用户可以有多个组，默认组（主组）一般都是用户名字

第一步 useradd 创建好后只能在终端操作：没有sudo权限和界面登录权限

-d 指定用户登录系统时的主目录，如果不适用此参数，系统自动在/home/目录下建立相同目录

-m 自动建立目录

-g 指定默认组名称,-G是添加其它组

python@ubuntu:~$ sudo useradd wenwenliang -m -d /home/wenwenliang 增加账户

第二步 passwd创建好后需要更改密码：因为没有默认密码的

python@ubuntu:~$ sudo passwd wenwenliang 更改密码

python@ubuntu:~$ su wenwenliang 更换用户

第三步 usermod –a -G 赋予账户sudo权限和界面登录权限

其实就是为用户添加组，Linux有组的概念:默认有许多组：

还未进行添加组只打印：默认的账户

root@ubuntu:~# groups wenwenliang 显示用户的组命令

wenwenliang : wenwenliang 打印默认账户

进行添加用户组操作

root@ubuntu:~# sudo usermod -a -G adm wenwenliang -G配合'-a'来完成添加其它组

root@ubuntu:~# sudo usermod -a -G sudo wenwenliang

进行添加完组后多了两个组adm sudo

root@ubuntu:~# groups wenwenliang 查看用户有那些组

wenwenliang : wenwenliang adm sudo 打印默认账户、adm、sudo

组的操作 Linux有组的概念:默认有许多组：每个组有每个组的权限

adm(admin)：有此组等于可以进行界面登录

sudo：有此组等于可以进行最高权限的操作可以使用sudo，sudo –s 可以切换root

查看系统组

方法一: cat /etc/group etc还有一个重要文件/etc/sudoers

方法二: groupmod +三次tab键

查看用户组 groups

root@ubuntu:~# groups wenwenliang

添加和删除组命令groupadd 用户名、groupdel 用户名

增加组：root@ubuntu:~# groupadd xxxx 注意这个没有用户的，创建用户时候就会默认给你创建以用户命名的默认组

修改用户所在组：usermod

usedel 删除用户

userdel abc(用户名) 删除abc用户，但不会自动删除用户的主目录

userdel -r abc(用户名) 删除用户，同时删除用户的主目录

python@ubuntu:~$ sudo userdel -r wenwenliang

[sudo] python 的密码：

userdel: wenwenliang 邮件池 (/var/mail/wenwenliang) 未找到

python@ubuntu:~$

## 进程/端口

### 检查端口

查看进程信息：ps 命令：ps –aux

-a 显示终端上的所有进程，包括其他用户的进程

-u 显示进程的详细状态

-x 显示没有控制终端的进程

-w 显示加宽，以便显示更多的信息

-r 只显示正在运行的进程

指定名称有无开启应用

命令：ps aux|grep redis //检查redis有无开启：有显示两条：一条是自己输入命令，一条是开启的端口

强制关闭端口：kill -9 pid

动态显示进程：top

M 根据内存使用量来排序

P 根据CPU占有率来排序

T 根据进程运行时间的长短来排序

U 可以根据后面输入的用户名来筛选进程

K 可以根据后面输入的PID来杀死进程。

q 退出

h 获得帮助

终止进程：kill kill命令指定进程号的进程，需要配合 ps 使用

使用格式： kill pid 终极杀死：kill -9

### 开启端口

1.测试远程主机的端口是否开启（windows命令行下执行）

telnet 192.168.1.103 80

2.查看本地的端口开启情况(ubuntu下执行)

sudo ufw status

3.打开80端口(ubuntu下执行)

sudo ufw allow 80

4.防火墙开启(ubuntu下执行)

sudo ufw enable

5.防火墙重启(ubuntu下执行)

sudo ufw reload

## 权限操作

文件夹 dbs说明

drwxrwxr-x 2 python python 4.0K 7月 30 2016 dbs

d：代表文件夹；如果是 - 证明是普通文件

rwx：用户（本人）python有什么权限：就看前三个r代表有写的权限，w有读的权限，x有执行的权限

rwx：同组者它的权限：后三个

r-x：不是同组也不是本人的权限

chmod 修改文件读、写、执行、的权限

修改本用户操作 u代表本用户意思

账户wenwenliang在文档目录创建a.py

wenwenliang@ubuntu:~/文档$ touch a.py //创建操作

wenwenliang@ubuntu:~/文档$ ls -lh //查看权限操作

-rw-rw-r-- 1 wenwenliang wenwenliang 0 7月 8 17:52 a.py //可以看到本用户只有读写操作没有执行操作

修改账户可以读写执行这个文件的权限

wenwenliang@ubuntu:~/文档$ chmod u=rwx a.py // u代表本用户

wenwenliang@ubuntu:~/文档$ ls -lh

-rwxrw-r-- 1 wenwenliang wenwenliang 0 7月 8 17:52 a.py //此时就增加了一个x

修改同组用户操作 g代表同组意思

wenwenliang@ubuntu:~/文档$ chmod g=rwx a.py 把文件a.py 同组权限修改为可读可写可执行权限

wenwenliang@ubuntu:~/文档$ ls -lh

-rwxrwxr-- 1 wenwenliang wenwenliang 0 7月 8 17:52 a.py

wenwenliang@ubuntu:~/文档$ chmod g= a.py 把g= 什么也不写就全部干掉权限

-rwx---r-- 1 wenwenliang wenwenliang 0 7月 8 17:52 a.py

修改其它用户操作 0代表其它意思

小例子 执行带命令的文件

wenwenliang@ubuntu:~/文档$ touch a.sh 第一步 创建文件a.sh并写内容 ：ls –lh

wenwenliang@ubuntu:~/文档$ ls -lh 第二步 查看权限 并为本用户有执行权限

-rw-rw-r-- 1 wenwenliang wenwenliang 7 7月 8 18:15 a.sh

wenwenliang@ubuntu:~/文档$ chmod u+x a.sh 添加执行权限

wenwenliang@ubuntu:~/文档$ ./a.sh 第三步 执行文件 a.sh 效果就是打印当前文件的详细情况

-rwx---r-- 1 wenwenliang wenwenliang 0 7月 8 17:52 a.py

-rwxrw-r-- 1 wenwenliang wenwenliang 7 7月 8 18:15 a.sh

数字表示权限

r换成数字表示是4，w换成数字表示是2，x换成数字表示是1

如果上面的操作增加本用户所有权限就可以变成

wenwenliang@ubuntu:~/文档$ chmod u=rwx a.py -🡪 chmod 7 a.py 很奇怪单写7是更改其它用户权限

改为本用户全部权限、同组用户读写权限、其它用户只有只读权限

wenwenliang@ubuntu:~/文档$ chmod u= chmod 764 a.py 4+2+1=7：7代表本用户的读写执行权限都有

修改文件夹下的所有权限：参数 –R（递归）

wenwenliang@ubuntu:~/文档$ chmod 777 aa -R aa是文件夹:这样文件夹下的文件和文件夹都有所有权限

修改文件的所有者 chown

wenwenliang@ubuntu:~/文档$ ls -lh

-rw-rw-r-- 1 wenwenliang wenwenliang 7 7月 8 18:15 a.sh 本来文件用户是 wenwenliang

wenwenliang@ubuntu:~/文档$ chown zenglifang a.sh 修改为文件用户为 zenglifang

修改文件所属组 chgrp

wenwenliang@ubuntu:~/文档$ ls -lh

-rw-rw-r-- 1 wenwenliang wenwenliang 7 7月 8 18:15 a.sh 本来文件组是 wenwenliang

wenwenliang@ubuntu:~/文档$ chgrp zenglifang a.sh 修改为文件组为 zenglifang

## 关机重启

关机重启：reboot、shutdown、init

命令 含义

reboot 重新启动操作系统

shutdown –r now 重新启动操作系统，shutdown会给别的用户提示

shutdown -h now 立刻关机，其中now相当于时间为0的状态

shutdown -h 20:25 系统在今天的20:25 会关机

shutdown -h +10 系统再过十分钟后自动关机

init 0 关机

init 6 重启

## 其它

检测磁盘空间：df

df命令用于检测文件系统的磁盘空间占用和空余情况，可以显示所有文件系统对节点和磁盘块的使用情况。

选项 含义

-a 显示所有文件系统的磁盘使用情况

-m 以1024字节为单位显示

-t 显示各指定文件系统的磁盘空间使用情况

-T 显示文件系统

检测目录所占磁盘空间：du

du命令用于统计目录或文件所占磁盘空间的大小，该命令的执行结果与df类似，du更侧重于磁盘的使用状况。

du命令的使用格式如下： du [选项] 目录或文件名

选项 含义

-a 递归显示指定目录中各文件和子目录中文件占用的数据块

-s 显示指定文件或目录占用的数据块

-b 以字节为单位显示磁盘占用情况

-l 计算所有文件大小，对硬链接文件计算多次

# 添加桌面快捷方式

第一步 在桌面，新建一个文件，以desktop为后缀

命令 touch e.desktop

第二步 打开创建的文件添加以下内容

[Desktop Entry]

Name=eclipse //名称

Exec=sh /home/du/wine/TIM.sh //执行路径

Icon=/home/du/wine/TIM.ico //图标

Type=Application

StartupNotify=true

注意：像eclipse是不用sh的：Exec= /home/wwl/eclipse/eclipse