

###查看当前空间大小

```
du -h --max-depth=1
```

###挂载192.168.10.230上的共享盘

```
mount -o username=ame,password=123456 //192.168.10.230/share0 /mnt
```

###生成20万的文件，以.dat结尾的文件

```
seq 200000 | xargs -i dd if=/dev/zero of={}.dat bs=1M count=1
```

###打包当前文件以 .dat结尾的文件

```
find . -name "*.dat*" | xargs tar -rf l.tar
```

## Centos7修改默认启动级别（命令行，图形切换）

方法一：

终端输入以下命令

修改为命令行方式

```
systemctl set-default multi-user.target
```

修改为图形界面

```
systemctl set-default graphical.target
```

方法二：（Centos7不支持这种方法了，7以前的版本就支持）

修改/etc/inittab文件，增加以下语句

```
id:5:initdefault: #图形模式
```

或者

```
id:3:initdefault: #命令行模式
```

-----

## 状态码

magedu.com

专注于Linux培训

状态代码	状态描述	说明
200	OK	客户端请求成功
400	Bad Request	由于客户端请求有语法错误，不能被服务器所理解
401	Unauthorized	请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用
403	Forbidden	服务器收到请求，但是拒绝提供服务，服务器通常会在响应正文中给出不提供服务的原因
404	Not Found	请求的资源不存在，例如：输入了错误的URL
500	Internal Server Error	服务器发生不可预期的错误，导致无法完成客户的请求
503	Service	服务器当前不能够处理客户端的请求，在一段时间之后，

查看WIFI密码：

管理员: C:\Windows\system32\cmd.exe

```
C:\Users\Administrator>netsh wlan show profiles "xiaofei" key=clear
```

接口 WLAN 上的配置文件 Xiaofei:

已应用：所有用户配置文件

配置文件信息

```
=====
版本                : 1
类型                : 无线局域网
名称                : Xiaofei
控制选项            :
  连接模式          : 自动连接
  网络广播          : 只在网络广播时连接
  AutoSwitch        : 请勿切换到其他网络
  MAC 随机化        : 禁用
```

连接设置

```
=====
SSID 数目           : 1
SSID 名称           : "Xiaofei"
网络类型            : 结构
无线电类型          : [ 任何无线电类型 ]
供应商扩展名        : 不存在
```

安全设置

```
=====
身份验证            : WPA2 - 个人
```

学习nmap网站：

[https://www.cnblogs.com/nmap/p/6232207.html?tdsourcetag=s\\_pcqq\\_aiomsg](https://www.cnblogs.com/nmap/p/6232207.html?tdsourcetag=s_pcqq_aiomsg)

lsmod 《驱动名》 ##查看驱动状态

insmod 《驱动名》 ##启动驱动

rmod 《驱动名》 ##卸载驱动

学习shell脚本网站:

[https://blog.csdn.net/qq\\_36119192/article/details/82964713](https://blog.csdn.net/qq_36119192/article/details/82964713)

=====

shell 脚本:

`$@` :表示脚本所有参数内容

`$#`: 表示参数的个数

eg:

`$@`: 表示所有脚本参数的内容

`$#`:表示返回所有脚本参数的个数。

示例: 编写如下shell脚本, 保存为test.sh

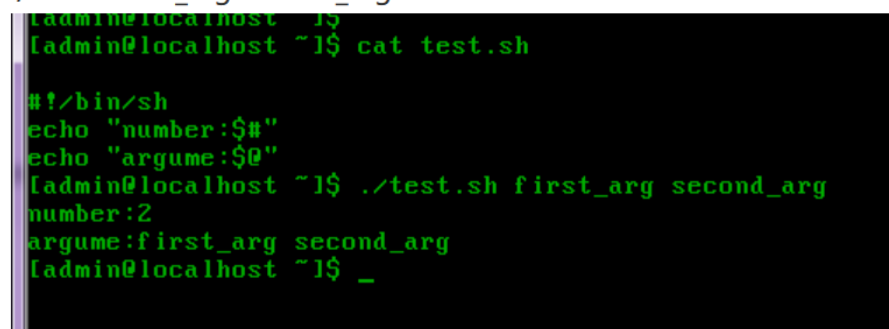
```
#!/bin/sh
```

```
echo "number:$#" 
```

```
echo "argume:$@"
```

执行脚本:

```
./test.sh first_arg second_arg
```



```
[admin@localhost ~]$ cat test.sh
#!/bin/sh
echo "number:$#"
echo "argume:$@"
[admin@localhost ~]$ ./test.sh first_arg second_arg
number:2
argume:first_arg second_arg
[admin@localhost ~]$ _
```

说明: 给脚本提供了两个参数, 所以`$#`输出的结果是2, `$@`代表了参数的内容!

## 输出

格式:printf 格式声明 与格式声明相对应的参数列表

格式声明由两部分组成：百分比符号(%)和指示符。

%d %s %c %f 格式替代符:

d : data 数字 -- 对应位置参数必须是数字型

s : str 字符串 -- 对应位置参数必须是字符串

c : char 字符 -- 对应位置是显示相对应参数的第一个字符

f : float 浮点 -- 对应位置参数必须是数字型

[https://blog.csdn.net/weixin\\_44878850](https://blog.csdn.net/weixin_44878850)

%d对应位置参数必须是整数，%f是小数。

%s对应位置参数必须是字符串，%c是相对应参数的第一个字符。

字符串 格式" " 双引号      至少包含一个字符      比如 "hello"

字符 格式' ' 单引号      有且只有一个字符      比如 't'

```
printf "%-10s %-8s %-4s\n" name sex kg
printf "%-10s %-8s %-4.2f\n" lisi nan 66.1234
printf "%-10s %-8s %-4.2f\n" zhangsan nan 48.6543
printf "%-10s %-8s %-4.2f\n" wangwu nv 47.9876
```

```
printf "%-10s %-8s %-4s\n" name sex kg
printf "%-10s %-8s %-4.2f\n" lisi nan 66.1234
printf "%-10s %-8s %-4.2f\n" zhangsan nan 48.6543
printf "%-10s %-8s %-4.2f\n" wangwu nv 47.9876
```

%-10s 指一个宽度为10个字符(-表示左对齐，没有则表示右对齐)，任何字符都会被显示在10个字符宽的字符内，如果不足则自动以空格填充，超过也会将内容全部显示出来。

%-4.2f 指格式化为小数，其中.2指保留2位小数。

```
printf "%s %s %s\n" a b c d e f g h i j
```

```
CentOS x
#!/bin/bash
printf "%-10s %-6s %-5.3f" zhangsan nan 50.6789
~
~
~
~
~
```

```
CentOS x
[root@qing jiaoben]# ./bin/bash printf.sh
zhangsan nan 50.679[root@qing jiaoben]# _
```

这里不会自动换行的，所以需要使用\n帮助去换行

```
CentOS x
#!/bin/bash
printf "%-10s %-6s %-5.3f\n" zhangsan nan 50.6789
~
~
~
~
~
```

```
CentOS x
[root@qing jiaoben]# ./bin/bash printf.sh
zhangsan nan 50.679
[root@qing jiaoben]# _
```

```
CentOS x
#!/bin/bash
printf "%-10s %-6s %-5.3f\n" zhangsan nan 50.6789
printf "%s %s %s\n" a b c d e f g h i g k l m n o p q r s t
~
~
~
~
~
```

```
CentOS x
[root@qing jiaoben]# ./bin/bash printf.sh
zhangsan nan 50.679
a b c
d e f
g h i
g k l
m n o
p q r
s t
[root@qing jiaoben]# _
```

Shell 中常见的算术运算命令

运算操作符及运算命令	意义
(( ))	用于整数运算的常见运算符，效率很高
let	用于整数运算，类似于(( ))
expr	可用于整数运算，但还有很多其他功能
bc	Linux下的一个计算器程序
\$[]	用于整数运算 <a href="https://blog.csdn.net/qq_36119192">tps://blog.csdn.net/qq_36119192</a>

变量的算术运算

Shell中常见的算术运算符

算术运算符	意义
+, -, *, /, %	加法、减法、乘法、除法、取余
**	幂运算
++, --	增加及减少
!, &&,	逻辑非（取反）、逻辑与（and）、逻辑或（or）
<, <=, >, >=	比较符号（小于、小于等于、大于、大于等于）
==, !=	比较符号（相等、不相等）
=, +=, -=	赋值运算符，例如a+=1相当于a=a+1 <a href="https://blog.csdn.net/qq_36119192">tps://blog.csdn.net/qq_36119192</a>

Shell 中常见的算术运算命令

运算操作符及运算命令	意义
(( ))	用于整数运算的常见运算符，效率很高
let	用于整数运算，类似于(( ))
expr	可用于整数运算，但还有很多其他功能
bc	Linux下的一个计算器程序
\$[]	用于整数运算 <a href="https://blog.csdn.net/qq_36119192">tps://blog.csdn.net/qq_36119192</a>

双小括号 (()) 数值运算命令的用法

双小括号 (()) 的作用是进行数值运算与数值比较，它的效率很高，用法灵活，是Linux下常用的运算操作符。其操作为：

运算操作符与运算命令	意义
((i=i+1))	此种书写方法为运算后赋值法，即将i+1的运算结果赋值给变量i。注意，不能用” echo ((i=i+1))” 的形式输出表达式的值，但可以使用 “echo \$((i=i+1))” 输出其值。
i=\$((A+5))	可以在“(())”前加\$符，表示将表达式 \$A+5 运算后的值赋值给i
((8>7&&5==5))	可以进行比较操作，还可以加入逻辑与和逻辑或，用于条件判断，if ((8>7&&5==5))
echo \$((2+1))	需要直接输出 运算表达式的运算结果时，可以在“(())”前加\$，输入3 <a href="https://blog.csdn.net/qq_36119192">https://blog.csdn.net/qq_36119192</a>

整数二元比较操作符

在[]以及test中使用的比较符号	在(())和[]中使用的比较符号	说明
-eq	== 或 =	相等，全拼为 equal
-ne	!=	不相等，全拼为 not equal
-gt	>	大于，全拼为 greater than
-ge	>=	大于等于，全拼为 greater equal
-lt	<	小于，全拼为 less than
-le	<=	小于等于，全拼为 less equal

逻辑操作符

在[]和test中使用的操作符	在(())和[]中使用的操作符	说明
-a	&&	and，与，两端都为真，才为真
-o		or，或，两端有一个为真，就为真
!	!	not，非，两端相反，则结果为真

测试表达式 test、[]、[]、(()) 的区别

测试表达式符号	test	[]	[]	(())
边界是否需要空格	需要	需要	需要	不需要
逻辑操作符	!、-a、-o	!、-a、-o	!、&&、	!、&&、
整数比较操作符	-eq、-gt、-lt、-ge、-le	-eq、-gt、-lt、-ge、-le	-eq、-gt、-lt、-ge、-le 或 =、>、<、>=、<=	=、>、<、>=、<=
字符串比较操作符	=、==、!=	=、==、!=	=、==、!=	不支持
文件操作	-d、-f、-e、-r、-s、-w、-x、-L、-nt、-ot	-d、-f、-e、-r、-s、-w、-x、-L、-nt、-ot	-d、-f、-e、-r、-s、-w、-x、-L、-nt、-ot	不支持
是否支持通配符匹配	不支持	不支持	支持	不支持

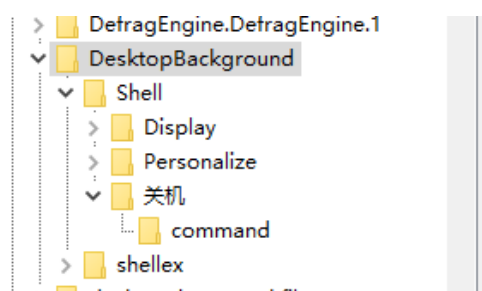
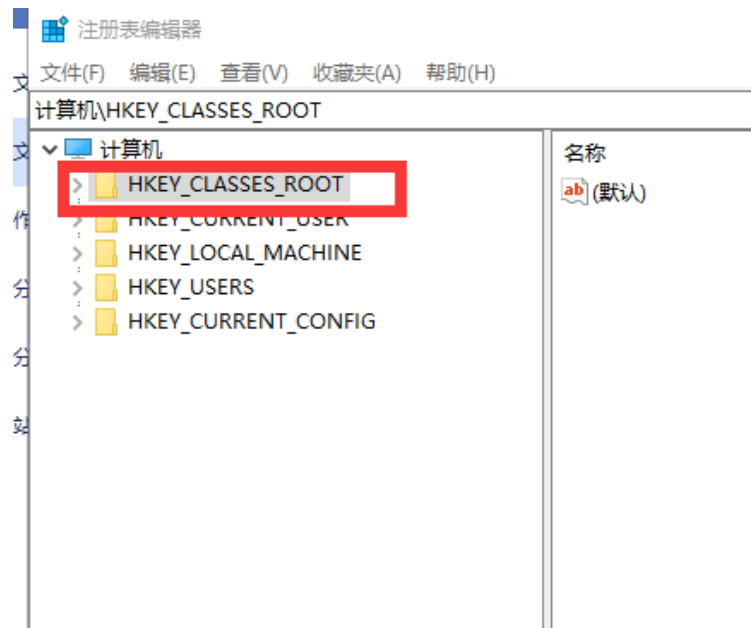
---

## nginx: 优化      隐藏nginx版本号

---

windows 右键添加关机按钮

regedit



Shutdown -s -f -t 00

---

## 软RAID相关命令

cat /proc/mdstat                      ##查看raid5信息

mdadm -D /dev/md1                    ##查看raid5详细信息

mdadm /dev/md1 --re-add /dev/sdf1 /dev/sdgl                    ##添加磁盘到raid阵列，sdf1、sdgl为磁盘设备



## 删除软RAID时，操作如下：

首先停止软raid，然后执行清理盘操作，删除/etc/mdadm.conf文件，最后重新组建软raid5

停止 `mdadm -S /dev/md1`

查看sdb1在raid信息 `mdadm -E /dev/sdb1`

清除sdb1在raid信息 `mdadm --zero-superblock /dev/sdb1`

清理后可以再执行`mdadm -E /dev/sdb1` 查看raid信息是否清理成功

## 自建仓库

前言：常用RPM的朋友们都知道，RPM简单易用，但是它的依赖关系是最头疼的！

有时候比方说A包需要B包，B包需要C包，C包需要A包，好了。这就是最常见的死锁了（类似数据库有木有？）。

这个时候有以下几种方式可以解决：

A、强制安装 ---暴力型

```
# rpm -ivh--force --nodeps gcc-c++-4.1.2-42.el5.i386.rpm
```

warning: gcc-c++-4.1.2-

42.el5.i386.rpm: Header V3 DSA signature: NOKEY, key ID 37017186

Preparing... ##### [100%]

1: gcc-c++ ##### [100%]

使用`rpm -ivh --force --nodeps` 强制安装。忽略依赖关系。这种方法你可以先装A包，再装B包，再装C包。这样还是有点隐患的，感觉不是很踏实（虽然其实目前没发现什么不好）。安装后使用成功的前提是：你要搞清楚依赖关系，并且把这些包都装好。好处是：不用管它们的具体依赖关系先后顺序。

B、一次性全装上 --- 一网打尽型

可以把依赖的几个包拷出来放在同一个文件夹里 然后 `rpm -ivn *.rpm` 这样也可以 前提也是一个都不能少。

```
# rpm -ivn *.rpm
```

```
# yum -y localinstall *.rpm
```

C、使用yum技术安装 --使用服务器方式

yum是一个服务器资源技术。通过在线下载服务器资源的方式。

缺点：太繁琐。要设置一堆的东西。优点：设置以后，很方便，需要的大多数资源都可以从服务器上找到。

```
yum deplist PACKAGE_NAME  列出一个包所有的依赖
```

如果没有yum源，我们要离线在一台服务器上安装httpd，那么可以把所有依赖打包：

```
# export LANG=en_US.utf-8
```

```
# yum deplist httpd | grep provider | awk '{print $2}' | sort | uniq
```

```
# yumdownloader $(yum deplist httpd | grep provider | awk '{print $2}' | sort | uniq)
```

```
# yumdownloader httpd
```

然后在安装服务器上：

```
# yum -y localinstall httpd
```

注：注意软件包的版本

检查系统是否已安装某个软件包：

```
# rpm -q --qf '%{NAME}-%{VERSION}-%{RELEASE} (%{ARCH})\n' PACKAGE_NAME
```

```
# yum -y install createrepo
#mkdir -p /var/ftp/pub/localrepo
#mv -f oracle_depspackage/*/var/ftp/pub/localrepo
#touch /etc/yum.repos.d/localrepo.repo
#echo "[localrepo]" >> /etc/yum.repos.d/localrepo.repo
#echo "name=Unixmen Repository" >> /etc/yum.repos.d/localrepo.repo
#echo "baseurl=file:///var/ftp/pub/localrepo" >> /etc/yum.repos.d/localrepo.repo
#echo "gpgcheck=0" >> /etc/yum.repos.d/localrepo.repo
#echo "enabled=1" >> /etc/yum.repos.d/localrepo.repo
#createrepo -v /var/ftp/pub/localrepo
#yum clean all
```

=====