1. Linux shell有三种符号

参考：<https://www.cnblogs.com/cssdongl/p/6679380.html>;

1. 由单引号括起来的字符都作为普通字符出现。特殊字符用单引号括起来以后，也会失去原有意义，而只作为普通字符解释。如例程中的 '-1 day'，再比如：

$ string=’$PATH’

$ echo $string

$PATH

可见$保持了其本身的含义，作为普通字符出现。

1. 由双引号括起来的字符，除$（美元符号）、\（反斜杠）、’（单引号）、和”（双引号）这几个字符仍是特殊字符并保留其特殊功能外，其余字符仍作为普通字符对待。对于“$”来说，就是用其后指定的变量的值来代替这个变量和$；对于“\”而言，是转义字符，它告诉shell不要对其后面的那个字符进行特殊处理，只当作普通字符即可。可以想见，在双引号中需要在前面加上“\”的只有四个字符$，\，’和”本身。而对”号，若其前面没有加“\”，则Shell会将它同前一个”号匹配。如例程中的"-$i day"，会随着循环，将$i转换成其代表的值。
2. 反引号（`）这个字符所对应的键一般位于键盘的左上角，不要将其同单引号（’）混淆。反引号括起来的字符串被shell解释为命令行，在执行时，shell首先执行该命令行，并以它的标准输出结果取代整个反引号（包括两个反引号）部分。如例程中的`date -d '-1 day' +%Y%m%d`就是把这条命令的结果赋给变量OPDATE。

= “$( )”

1. 使用yum 和rpm的安装方式

Yum和rpm是两种软件安装方式，

Yum：Yum（全称为 Yellow dog Updater, Modified）是一个在Fedora中的Shell前端软件包管理器。基於RPM包管理，能够从指定的服务器自动下载RPM包并且安装，可以自动处理 依赖性关系，并且一次安装所有依赖的软体包，无须繁琐地一次次下载、安装。，会检测软件包的依赖关系，并且自动从镜像源中下载对应包，可以一次性安装到位，与MacOS中Homebrew相似，缺点：需要配置相应的yum源；

命令: yum install xxxxx (tab) –查找镜像源中包

Yum list --列出可安装软件

yum grouplist --列出可安装组软件

yum repolist --列出已安装镜像源

Rpm：Redhad package 安装器，是Redhad类Linux系统中的软件安装器，可运行rpm包，

一般从官网下载最新软件包，检查依赖但不自动获取，需要手动使用yum安装对应依赖，有点软件一般为最新版， 缺点检查依赖繁琐

命令：rpm -ivh xxx.rpm ---安装并检查依赖源

Rpm -Uvh xxx.rpm ---先删除已有软件再安装

Rpm -e xxxx ---卸载已安装的软件

补充：常配合wget Url –从url下载文件， 配合使用

1. 更换yum源，RH中的yum源需要注册才可以使用，centos为开放源，但服务器在国外，有些软件下载比较慢，可参考使用163，阿里镜像源，方法如下

163为例（过程中使用<http://mirrors.163.com>最新地址）

1. 检查是否需要重现安装yum软件（一般不需要）

#1.查看系统版本

[root@localhost huan]# cat /etc/redhat-release

Red Hat Enterprise Linux Server release 7.4 (Maipo)

#2.查看系统本身所安装的yum 软件包

[root@localhost huan]# rpm -qa | grep yum

PackageKit-yum-1.1.5-1.el7.centos.x86\_64

yum-metadata-parser-1.1.4-10.el7.x86\_64

yum-utils-1.1.31-42.el7.noarch

yum-plugin-fastestmirror-1.1.31-42.el7.noarch

yum-langpacks-0.4.2-7.el7.noarch

yum-rhn-plugin-2.0.1-9.el7.noarch

yum-3.4.3-154.el7.centos.noarch

#3.删除自带的yum包

rpm -qa|grep yum|xargs rpm -e --nodeps （不检查依赖，直接删除rpm包）

#4.再次查看系统本身yum 软件包是否已删除

rpm -qa | grep yum

#5.下载相关安装包RPM --（其中7.4.1708可替换成最新地址）

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/PackageKit-yum-1.1.5-1.el7.centos.x86\_64.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-metadata-parser-1.1.4-10.el7.x86\_64.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-utils-1.1.31-42.el7.noarch.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-plugin-fastestmirror-1.1.31-42.el7.noarch.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-langpacks-0.4.2-7.el7.noarch.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-rhn-plugin-2.0.1-9.el7.noarch.rpm

wget http://mirrors.163.com/centos/7.4.1708/os/x86\_64/Packages/yum-3.4.3-154.el7.centos.noarch.rpm

#6.安装下载的RPM安装包,按顺序执行下面命令

rpm -ivh yum-metadata-parser-1.1.4-10.el7.x86\_64.rpm yum-3.4.3-154.el7.centos.noarch.rpm yum-plugin-fastestmirror-1.1.31-42.el7.noarch.rpm

rpm -ivh PackageKit-yum-1.1.5-1.el7.centos.x86\_64.rpm yum-utils-1.1.31-42.el7.noarch.rpm yum-langpacks-0.4.2-7.el7.noarch.rpm yum-rhn-plugin-2.0.1-9.el7.noarch.rpm

1. 备份/etc/yum.repos.d/ 中默认的源文件（具体配置可以参考已有文件进行配置）

mv /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.backup

1. 从镜像网站下载repo配置文件或自行配置

<http://mirrors.163.com/.help/> --选择对应对版本.repo

将.repo 中$releasever 修改成对应目录版本

并修改为CENTOS-BASE.REPO 或修改yum配置文件配置

1. 运行yum makecache生成缓存

yum clean all

yum makecache  
yum update

1. 补充

阿里云网络源地址：<https://mirrors.aliyun.com/centos/7/os/x86_64/Packages/>   
网易163网络源地址：<http://mirrors.163.com/>   
CentOS网络源地址：<http://centos.ustc.edu.cn/centos/>

CENTOS 源文件名称：镜像源 CENTOS-BASE.REPO

媒体源（dvd）CENTOS-MEDIA.REPO

Reference: <https://blog.csdn.net/miaoyanmm/article/details/79686695>;

<https://www.cnblogs.com/F1renze/p/8577195.html>;

<https://www.cnblogs.com/syqlp/p/6555524.html>;

<https://blog.csdn.net/hongbin_xu/article/details/79316614>;

<https://www.cnblogs.com/jym1/p/8088005.html>;

1. 从光盘加载yum源

**一、挂载iso文件到挂载点**

[root@server ~ ]# mount  -o loop /mnt/iso/CentOS5.iso /mnt/cdrom

**二、查看挂载状态**

[root@server ~ ]# df -h

Filesystem            Size  Used Avail Use% Mounted on

/dev/mapper/VolGroup00-LogVol00

                       19G  3.3G   15G  19% /

/dev/sda1              99M   12M   82M  13% /boot

tmpfs                  62M     0   62M   0% /dev/shm

/mnt/iso/CentOS5.iso   3.9G  3.9G     0 100% /mnt/cdrom

**三、修改yum的配置文件，使用本地ISO做yum源**

# cd /etc/yum.repos.d/

# mv CentOS-Base.repo CentOS-Base.repo.bak

# cp CentOS-Media.repo CentOS-Media.repo.bak

# cat  CentOS-Media.repo

[c7-media] ---根据版本变更

name=CentOS-$releasever - Media

baseurl=file:///mnt/     #表明yum源在/mnt目录下，其它的源(baseurl)注释掉

#baseurl=file:///media/CentOS/

#        file:///media/cdrom/

#        file:///media/cdrecorder/

gpgcheck=1 ------或者设为0

enabled=1                   #启用yum

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

四、清除缓存

# yum clean all

# yum list

六、开机自动挂载iso镜像文件

在/etc/fstab表中，添加最后的一行，将iso文件永久挂载到系统中

[root@server ~ ]# cat /etc/fstab

/dev/VolGroup00/LogVol00 /                    ext3    defaults        1 1

LABEL=/boot             /boot              　ext3    defaults        1 2

/dev/VolGroup00/LogVol01 swap                 swap    defaults       0 0

/mnt/iso/CentOS5.iso    /mnt/cdrom           iso9660  loop           0 0

Reference：<https://www.cnblogs.com/oftenlin/p/4325023.html>;

1. 安装Linux不要使用KDE模式，很多功能不支持，选择Gnome Desktop模式
2. Linux Shell中$ 中的应用

a. 参数类应用 $var 与${var} 相同，${ } 更准确判定范围;解析数组时建议使用${}

shell脚本中-自定义变量 可使用$var 进行取值操作

-命令行引入参数 $0 ~9 当超过10的时候 ${10}..

-特殊变量$# 表示除文件名$0 之外的参数个数 ${!#} 表示最后一个变量(${} 中不允许使用$ 符号，所以用！替代)

-$\* 将$0之外的参数当做一个字符串整体

- $@ 将$0之外的参数一组单词的数组

-$$ 脚本中传入系统分配的PID

-$? 函数、命令等执行后的返回状态码

b. 代码替换类

$( ) 与 ` ` (反引号)： 在范围中的命令可以返回命令结果

$[ ] 与 $(( )) –整数基础算数运算，

另$((16#ff)) 可以将ff转换为10进制；

[ ] 与test：条件判断语句的condition判断,判断结果为是与否， 特殊符号需要转义符

[[ ]] 与 (( )) :

[[ ]] 对于[] 文本模式的增强，增加文本匹配模式；

(( ))：1. 加强对算数判断的增强

2. 可使用自增符号重定义变量 ((a++))

3. for循环条件中C语言形式的条件 (( i=x; i<=target; i++ ))

c. ${ }高级应用

1. 先赋值一个变量为一个路径，如下：

file=/dir1/dir2/dir3/my.file.txt

命令 解释 结果

${file#\*/} 拿掉第一条 / 及其左边的字符串 dir1/dir2/dir3/my.file.txt

[root@localhost ~]# echo ${file#\*/}

dir1/dir2/dir3/my.file.txt

${file##\*/} 拿掉最后一条 / 及其左边的字符串 my.file.txt

[root@localhost ~]# echo ${file##\*/}

my.file.txt

${file#\*.} 拿掉第一个 . 及其左边的字符串 file.txt

[root@localhost ~]# echo ${file#\*.}

file.txt

${file##\*.} 拿掉最后一个 . 及其左边的字符串 txt

[root@localhost ~]# echo ${file##\*.}

txt

${file%/\*} 拿掉最后一条 / 及其右边的字符串 /dir1/dir2/dir3

[root@localhost ~]# echo ${file%/\*}

/dir1/dir2/dir3

${file%%/\*} 拿掉第一条 / 及其右边的字符串 (空值)

[root@localhost ~]# echo ${file%%/\*}

(空值)

${file%.\*} 拿掉最后一个 . 及其右边的字符串 /dir1/dir2/dir3/my.file

[root@localhost ~]# echo ${file%.\*}

/dir1/dir2/dir3/my.file

${file%%.\*} 拿掉第一个 . 及其右边的字符串 /dir1/dir2/dir3/my￼

[root@localhost ~]# echo ${file%%.\*}

/dir1/dir2/dir3/my

记忆方法如下：

# 是去掉左边(在键盘上 # 在 $ 之左边)

% 是去掉右边(在键盘上 % 在 $ 之右边)

单一符号是最小匹配;两个符号是最大匹配

\*是用来匹配不要的字符，也就是想要去掉的那部分

还有指定字符分隔号，与\*配合，决定取哪部分

2. **取子串及替换**

命令 解释 　　 结果

${file:0:5} 　　　提取最左边的 5 个字节 　　　　　　　　　　　　/dir1

${file:5:5} 　　　提取第 5 个字节右边的连续 5 个字节 　　　　　/dir2

${file/dir/path} 将第一个 dir 提换为 path 　　　　　　　　　 /path1/dir2/dir3/my.file.txt

${file//dir/path} 　　　　将全部 dir 提换为 path 　　　　　　　　　　　/path1/path2/path3/my.file.txt

${#file} 　　　　　　　　　 获取变量长度

3. **根据状态为变量赋值**

| **命令** | **解释** | **备注** |
| --- | --- | --- |
| ${file-my.file.txt} | 若 $file 没设定,则使用 my.file.txt 作传回值 | 空值及非空值不作处理 |
| ${file:-my.file.txt} | 若 $file 没有设定或为空值,则使用 my.file.txt 作传回值 | 非空值时不作处理 |
| ${file+my.file.txt} | 若$file 设为空值或非空值,均使用my.file.txt作传回值 | 没设定时不作处理 |
| ${file:+my.file.txt} | 若 $file 为非空值,则使用 my.file.txt 作传回值 | 没设定及空值不作处理 |
| ${file=txt} | 若 $file 没设定,则回传 txt ,并将 $file 赋值为 txt | 空值及非空值不作处理 |
| ${file:=txt} | 若 $file 没设定或空值,则回传 txt ,将 $file 赋值为txt | 非空值时不作处理 |
| ${file?my.file.txt} | 若 $file 没设定,则将 my.file.txt 输出至 STDERR | 空值及非空值不作处理 |
| ${file:?my.file.txt} | 若 $file没设定或空值,则将my.file.txt输出至STDERR | 非空值时不作处理 |

**tips:**

以上的理解在于, 你一定要分清楚 unset 与 null 及 non-null 这三种赋值状态. 一般而言, : 与 null 有关, 若不带 : 的话, null 不受影响, 若带 : 则连 null 也受影响.

## 4. 数组

A="a b c def" # 定义字符串

A=(a b c def) # 定义字符数组

| **命令** | **解释** | **结果** |
| --- | --- | --- |
| ${A[@]} | 返回数组全部元素 | a b c def |
| ${A[\*]} | 同上 | a b c def |
| ${A[0]} | 返回数组第一个元素 | a |
| ${#A[@]} | 返回数组元素总个数 | 4 |
| ${#A[\*]} | 同上 | 4 |
| ${#A[3]} | 返回第四个元素的长度，即def的长度 | 3 |
| A[3]=xzy | 则是将第四个组数重新定义为 xyz |  |

7. Vim **基本查找和替换**

在Vim中，可以使用:substitute (:s)命令查找和替换文本。

要在Vim中运行命令，您必须处于正常模式，这是启动编辑器时的默认模式，要从其他任何模式返回正常模式，只需按“Esc”键。

命令的一般形式如下：

:[range]s/{pattern}/{string}/[flags] [count]

该命令在[range]的每一行中搜索{pattern}，并将其替换为{string}，[count]是一个乘以命令的正整数。

如果没有给出[range]和[count]，则仅替换当前行中找到的模式，当前行是光标所在的行。

例如，要在当前行中搜索字符串“foo”的第一个匹配项并将其替换为“bar”，可以使用：

:s/foo/bar/

要替换当前行中所有出现的搜索模式，请添加g标志：

:s/foo/bar/g

如果要搜索并替换整个文件中的模式，请使用百分比字符%作为范围，此字符表示从文件的第一行到最后一行的范围：

:%s/foo/bar/g

如果省略{string}部分，则将其视为空字符串，并删除匹配的模式，以下命令删除当前行中字符串“foo”的所有实例：

:s/foo//g

除了斜杠（/）之外，还可以使用其他任何非字母数字的单字节字符作为分隔符，当您在搜索模式或替换字符串中使用“/”字符时，此选项很有用：

:s|foo|bar|

要确认每个替换，请使用c标志：

:s/foo/bar/gc

返回：

replace with bar (y/n/a/q/l/^E/^Y)?

按y替换匹配项，或按l替换匹配项并退出，按n跳过，按q或Esc退出替换，使用a选项替换匹配项和所有其余匹配项，若要向下滚动屏幕，请使用CTRL+Y，向上滚动，请使用CTRL+E。

您还可以将正则表达式用作搜索模式，下面的命令将所有以'foo'开头的行替换为'Vim is the best'：

:%s/^foo.\*/Vim is the best/gc

^（脱字符）符号与行的开头匹配，.\*与任意数量的任何字符匹配。

**区分大小写**

默认情况下，搜索操作区分大小写，搜索“FOO”将不匹配“Foo”。

要忽略搜索模式的大小写，请使用i标志：

:s/Foo/bar/gi

强制忽略大小写的另一种方法是在搜索模式后附加\c，例如，/Linux\c执行忽略大小写搜索。

如果更改了默认大小写设置，并且要执行区分大小写的搜索，请使用I标志：

:s/foo/bar/gi

模式之后的大写\C也会强制进行大小写匹配搜索。

**搜索范围**

如果未指定范围，则替换命令仅在当前行中运行。

范围可以是一行，也可以是两行之间的范围，行说明符之间用,或;字符，可以使用绝对行号或特殊符号指定范围。

例如，要在从第3行到第10行的所有行中将所有出现的“foo”替换为“bar”，可以使用：

:3,10s/foo/bar/g

该范围是包含范围的，这意味着该范围包括第一行和最后一行。

点.字符表示当前行，$美元符号表示最后一行，要在从当前行到最后一行的所有行中替换“foo”：

:.,$s/foo/bar/

也可以使用“+”或“-”符号来设置行说明符，其后是从前一个行号添加或减去的数字，如果省略符号后的数字，则默认为1。

例如，从当前行和接下来的四行开始，用“bar”替换每个“foo”，键入：

:.,+4s/foo/bar/g

**替换整个单词**

替代命令将模式作为字符串而不是整个单词查找，例如，如果您正在搜索“gnu”，则搜索结果匹配“gnu”嵌入较大词（例如“cygnus”或“magnum”）中的位置。

要搜索整个单词，请键入\>标记单词的开头，输入搜索模式，键入\>标记单词的结尾：

例如，要搜索单词“foo”，可以使用\<foo\>：

:s/\<foo\>/bar/

**替代历史（History）**

Vim会跟踪您在当前会话中运行的所有命令，要浏览历史记录以查找先前的替代命令，请输入:s并使用向上/向下箭头键查找先前的替代操作，要运行该命令，只需按Enter，您也可以在执行操作之前编辑命令。

**例子**

注释行（在行前添加#）从5到20：

:5,20s/^/#/

取消注释行从5到20，恢复以前的更改：

:5,20s/^#//

将“apple”，“orange”和“mango”的所有实例替换为“fruit”：

:%s/apple\|orange\|mango/fruit/g

删除每行末尾的空格：

:%s/\s\+$//e

注：其它类似的操作按上面的格式自由替换就行。