目录

测试人员学习 Linux 学习方法和注意事项

1. 学习方法

多操作，多敲命令，注意看错误提示；路径问题，权限问题，命令本身敲错了。基本都是这几类问题

2. 学习注意事项

linux 基础命令，结合实际案例能够在项目中灵活运用各种基础命令

掌握测试人员常用的 linux 应用的场景中用到的命令必须掌握，比如搭建环境、查看日志、定时任务、查找文件、权限设置、vim 使用等等，也就是课本上的使用案例必须掌握

shell 部分难度较大，编程语言的学习需要一个长期的过程，所以不要求大家能够完全写出来，但是要能看懂代码。

命令的选项有很多，不可能都掌握，掌握最常用的就行，工作中用到时候可以查

1.Linux 简介

1.1. 前言

1.1.1. Linux 在软件测试中的运用

从事软件测试工程师岗位的人员都知道使用 Linux 对于测试工作的重要性，因为在工作中需要用到，面试中会被问到。现在软件测试工程师招聘要求之一是：掌握 Linux 基础知识。

对于软件测试人员来说，你需要学会一些常用的基础命令，这些命令让你能够去查看日

志，定位 bug，修改文件，搭建环境就 OK 了。那么至于 Linux 系统底层的实现对于大多数测试人员是不需要去了解的。

那么我们是如何在我们的工作中使用 Linux 呢?

相信很多人都知道，之所以我们会用到 Linux，是因为我们的产品将 Linux 系统作为我们服务器操作系统使用，当我们去测试产品时需要在 Linux 上部署产品，若产品某个功能出现错误，我们需要去排查出错的原因，基于这两个目的，我们就需要掌握一定的 Linux 命令。所以应用的场景有搭建环境和查看日志等。

下面是实际工作中更具体更详细的应用场景：

- 现在多数服务器部署 Linux，你需要学会看 Log；
- 如果没有持续集成体系，至少得会更新部署包；
- 需要更改环境变量文件；
- 需要更改配置文件内容；
- 常见的问题定位思路，需要用到的一些命令；
- 查看某个服务是否启动，执行：ps -ef|grep 服务名；
- 查看启动了哪些端口：执行：netstat –tupnl；
- 性能测试，Linux 是必须掌握的知识。因为需要搭建被测环境，部署服务，监控，以及用 shell 去写数据库备份的脚本；
- 定时清理日志文件；
- 自动化测试，Linux 必须掌握。搭建被测环境，部署服务；

1.1.2. 什么是服务器

大家玩游戏，逛淘宝，京东这些网站，你们看到的所有画面和页面都是从服务器返回过来的。有的同学可能玩过局域网游戏，例如 CS，玩的时候需要有一个人的机器作为服务器，其他人都连到这台机器，这样大家才能够一起进入一个地图进行 PK。大家第一阶段实战项目安装的 ecshop，你们用自己机器充当了服务器，访问 http://localhost:80/ecshop，这个 URL 就是本机。

从物理上来说，服务器也是一台机器，只是比普通 PC 配置更好，内存和 CPU 等硬件更好。服务器也需要操作系统，但是由于不同于个人电脑，不需要频繁操作使用，所以不用图形界面，为了节省资源，服务器一般都使用 Linux 作为操作系统。

1.1.3. 计算机组成

输入单元：键盘，鼠标，卡片阅读机，扫描仪，手写板，触摸屏
CPU：中央处理器，含有算术逻辑控制单元
输出单元：屏幕，打印机
内存：CPU 取数据直接从内存取
硬盘：持久化存储，数据库数据就存储在硬盘上

1.1.4. 进程的概念

很多同学应该都听过“进程”，但是如果让你准确描述什么是进程，可能不少同学说不清楚，大家应该都遇到过这种情况：在 windows 里面的任务管理器，电脑卡顿了我们会去杀死一些内存占用高的进程。这里我们再次详细解释什么是进程，这个概念对于大家后续学习性能测试非常重要。

进程（Process）是计算机中的程序关于某数据集合上的一次运行活动，是系统进行资源分配和调度的基本单位。

上面是官方解释，比较抽象。这句描述是站在操作系统的角度去解释的，操作系统好像一家公司的老板，公司里其他人都是统一称作“员工”，只是职位不同分工不同，都是为了管理的需要。所以"进程"这个概念也是操作系统为了管理服务器或者说 PC 机器 的硬件资源取的一个名字，类似公司老板为了管理公司统一称所有人"员工"，所以 你会发现数据库是一个进程，一个 web 服务是一个进程，任何这台机器上会占用资源的任务都是一个进程。

我们二阶段实战项目 CRM 系统，通过命令 service httpd start 和 service mysqld start 后，启动了 web 服务和数据库服务，执行 top 命令后就会看到有这 2 个进程：

```
top - 17:46:57 up 41 min,  3 users,  load average: 0.21, 0.24, 0.14
Tasks: 191 total,   2 running, 189 sleeping,   0 stopped,   0 zombie
%Cpu(s):  3.3 us,  8.6 sy,  0.0 ni, 88.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  1863224 total,  1329852 free,   279844 used,   253528 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  1381656 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 9816 root      20   0  230408   5192   3296 S   3.3  0.3   0:00.10 httpd
 9213 root      20   0  162012   2308   1592 R   1.6  0.1   0:01.44 top
    1 root      20   0  128276   6912   4212 S   1.0  0.4   0:04.03 systemd
 9770 mysql     20   0  980628  82540   7128 S   0.7  4.4   0:00.09 mysqld
 3517 root      20   0   39196   3180   2812 S   0.3  0.2   0:00.59 systemd-journal
 7211 root      20   0  225720   4820   3340 S   0.3  0.3   0:00.09 abrt-watch-log
```

1.1.5. Linux 与 Windows 的区别

基于以上的铺垫，大家现在应该对 Linux 有了一些理解了，Linux 首先是操作系统，和你们每个人的电脑里的 windows 一样，都是操作系统。操作系统是直接控制硬件的，你们买了电脑后第一件事就是装操作系统，有了操作系统，然后再部署应用服务。但是 Linux 和 windows 最大的区别就是 Linux 是命令行，windows 是图形界面。我们学习 windows 是非常简单的，操作上所见即所得。但是学习 Linux，必须在命令行进行操作，系统给你的唯一通道只有命令行，所以你要通过各种命令去发出指令。现在服务器基本都部署的 Linux，因为 Linux 相比图形界面占用系统资源少。

1.2. 工具简介和安装

1.2.1. 虚拟机安装

因为一台机器不可能同时运行多种操作系统，我们学习的机器装了 windows，所以只有在虚

拟机上面安装 Linux，虚拟机（Virtual Machine）指通过软件模拟的具有完整硬件系统功能的、运行在一个完全隔离环境中的完整计算机系统。我们安装的是 VMware Workstation Pro，大家可以自行下载安装(官网：https://www.vmware.com/)，详细步骤参考《vmware 安装文档.docx》。

1.2.2. Linux 系统简介

现在我们在虚拟机里安装 Linux，Linux 只是一个内核。一个完整的操作系统不仅仅是内核而已。所以，许多个人、组织和企业，开发了基于 GNU/Linux 的 Linux 发行版。Linux 的发行版说简单点就是将 Linux 内核与应用软件做一个打包。目前市面上较知名的发行版有：Ubuntu、RHEL、CentOS、Debian、Fedora、SuSE、OpenSUSE 等。

Centos 版本的特点和好处：

上面我们看到有各种 Linux 的发行版本，每个版本各有特点。我们学习是安装的 CentOS 版本，所以主要介绍下 CentOS 的特点和好处。

CentOS 版本的应用非常广泛，非常多的商业公司部署在生产环境上的服务器都是使用的 CentOS，CentOS 没有其他很多版本繁琐的 Linux 配置，命令行下的人性化做的比较好，而且服务非常稳定。

1.2.3. Linux 连接工具介绍

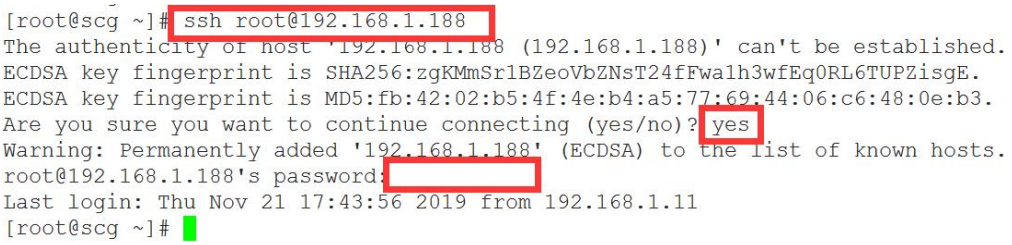
实际工作中，我们一般不直接在服务器 Linux 环境操作，而是通过工具远程连接到目标服务器，常用连接工具有：ScureCRT,putty,Xshell。

SecureCRT：SecureCRT 是一款支持 SSH（SSH1 和 SSH2）的终端仿真程序，简单地说是 Windows 下登录 UNIX 或 Linux 服务器主机的软件。SecureCRT 支持 SSH，同时支持 Telnet 和 rlogin 协议。SecureCRT 是一款用于连接运行包括 Windows、UNIX 和 VMS 的理想工具。

Xshell：Xshell 是一个强大的安全终端模拟软件，它支持 SSH1, SSH2, 以及 Microsoft Windows 平台的 TELNET 协议。Xshell 通过互联网到远程主机的安全连接以及它创新性的设计和特色帮助用户在复杂的网络环境中享受他们的工作。

Putty：PuTTY 是一个 Telnet、SSH、rlogin、纯 TCP 以及串行接口连接软件，Putty 是一个免费的、Windows x86 平台下的 Telnet、SSH 和 rlogin 客户端，但是功能丝毫不逊色于商业的 Telnet 类工具。

这些工具使用方法大同小异，我们学习的是 Xshell，但是大家以后工作中可能会用另外的连接工具，要能够快速上手，用法本质上是一样的。远程连接都是 SSH 协议，大家可以试着利用 ssh username@ip 连接到你旁边同学的环境，当然还需要输入对应用户名的密码。

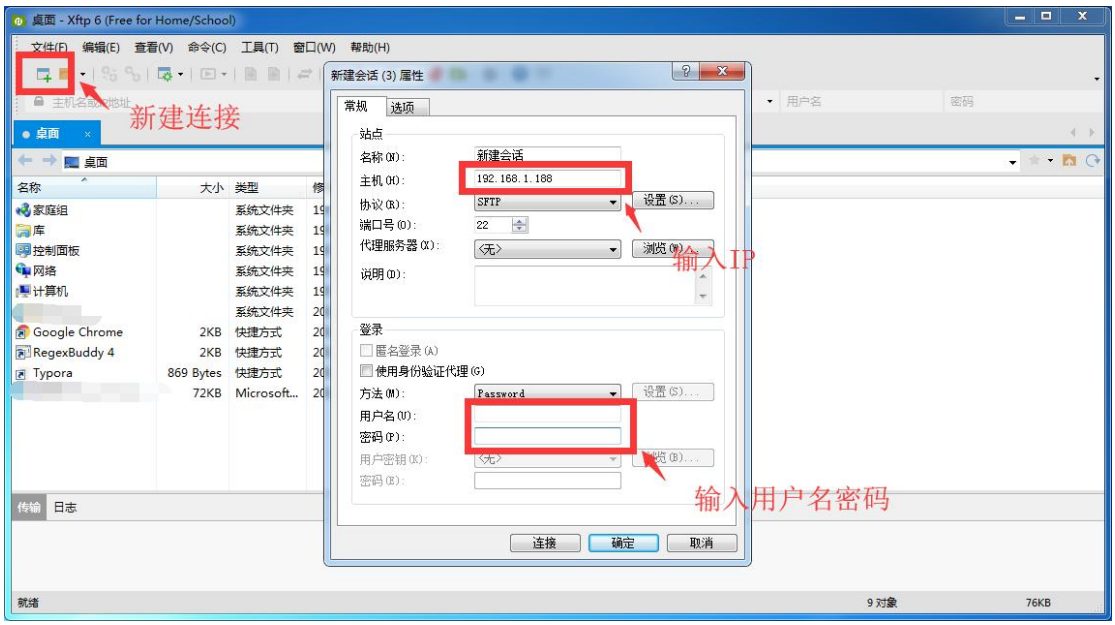


1.2.4. Windows 传文件到 Linux

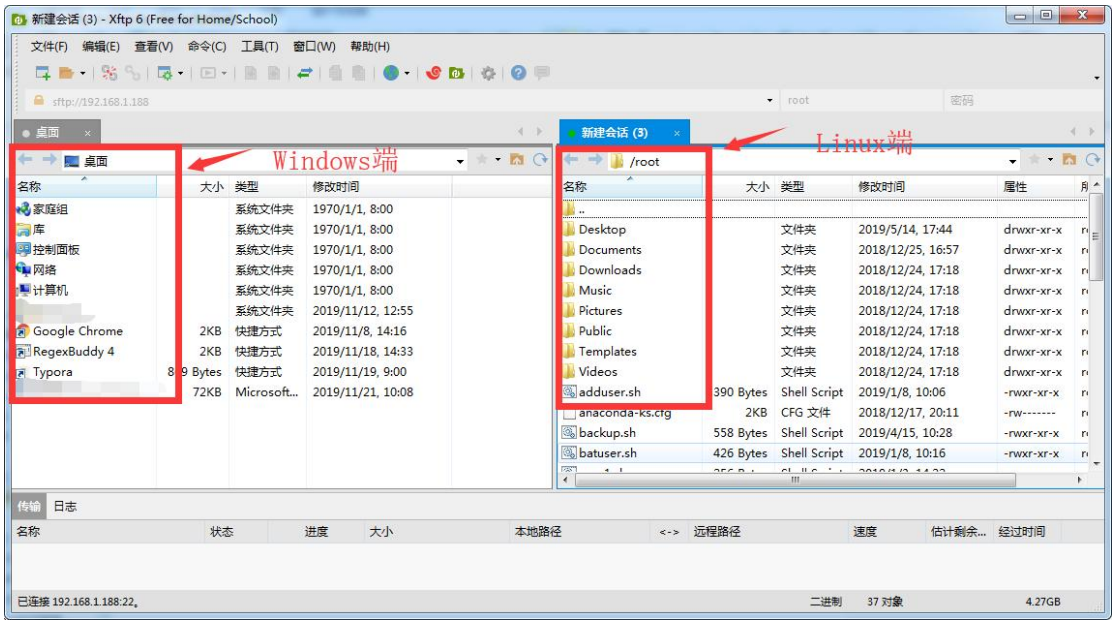
实际工作中，我们经常需要从 Windows 系统传文件到 Linux 系统。比如性能测试脚本开发，我们在自己的工作电脑上完成，是 Windows 系统，开发完成需要把脚本放到性能压测的负载机去执行，负载机一般都是 Linux 环境。再比如第二阶段项目实战 CRM 的系统包，大家肯定要放到 Linux 系统去部署，所以也需要从 Windows 传到 Linux 环境。

我们使用的工具是 Xftp，安装完成后，双击打开看到如下图，连接对应 IP 的 Linux 系统就行。

第一步：新建连接->输入 ip->输入用户名密码后点击确定



第二步：连接成功，拖动文件即可完成文件传递。

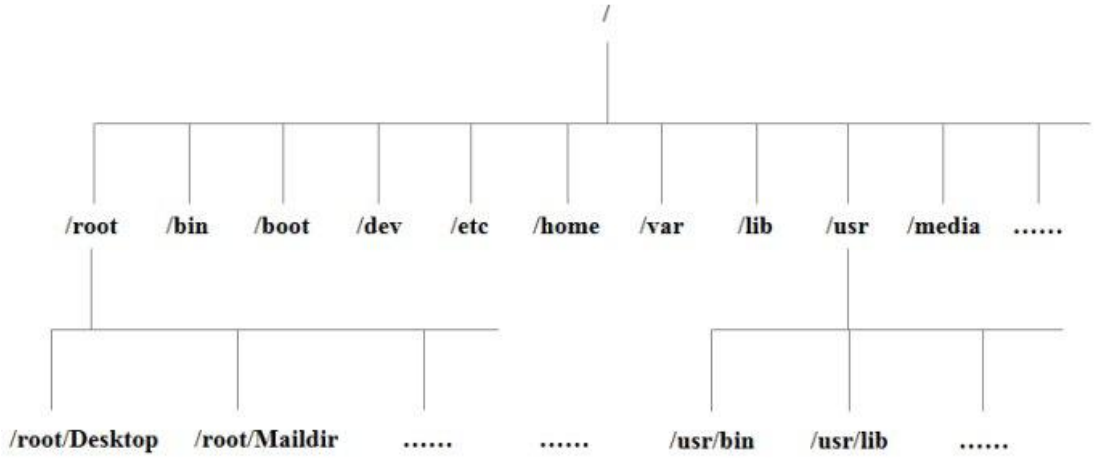


2. Linux 基础命令

2.1. Linux 常用命令

2.1.1 系统目录结构

在学习命令之前，我们先来看看 Linux 的系统目录结构。
Linux 系统的结构和 Windows 不同，在 Windows 里面磁盘可能被分成 C，D，E 在 Linux 里面任何内容都是文件，包括了硬件设备。都是以文件的形式来呈现的。/ 是根目录，就是最顶层，所有其他部分都是以文件形式“挂载”到根目录。



以下是对其中常见目录的解释：

- /bin, /sbin, /usr/bin, /usr/sbin：bin 是 Binary 的缩写，这是系统预设的执行文件的放置目录，比如 ls 就是在/bin/ls 目录下的，/bin, /usr/bin 是给系统用户使用的指令（除 root 外的通用户），而/sbin, /usr/sbin 则是给 root 使用的指令。
- /home：用户的主目录，在 Linux 中，每个用户都有一个自己的目录，一般该目录名是以用户的账号命名的。
- /root：该目录为系统管理员，也称作超级权限者的用户家目录。
- /etc：这个目录用来存放所有的系统管理所需要的配置文件和子目录。
- /var：这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下。包括各种日志文件。
- /boot：这里存放的是启动 Linux 时使用的一些核心文件，包括一些连接文件以及镜像文件。
- /usr： 这是一个非常重要的目录，用户的很多应用程序和文件都放在这个目录下，类似与 windows 下的 program files 目录。
- /opt：这是给主机额外安装软件所摆放的目录。比如你安装一个 ORACLE 数据库则就可以放到这个目录下。默认是空的。
- /tmp：这个目录是用来存放一些临时文件的。

2.1.2. 用户与身份

大家安装好登录进入 Linux 系统后，可以看看自己是什么账户。不同账户权限不同。权限是 Linux 中一个核心的概念,root 和普通用户最大的区别就是权限，root 可以理解成超级管理员，这种用户可以去到系统任何地方，但是普通用户有些地方就去不了。例如进入公司的一个测试人员维护 Linux 环境，如果只需要操作某个目录，就可以建一个普通身份账号，然后开放对应目录的三种权限（读，写，执行 看情况分配）。

- root 用户：

```
/root                          # root 用户的家目录为/root
```

- 普通用户：

```
/home/scg                      #普通用户 scg 的家目录为/home/scg
```

- 切换用户：

```
[root@localhost ~]# su - scg        # root 用户切换到普通用户，无需输入密码
[scg@localhost ~]$                  #切换成功
[scg@localhost ~]$ su - root        #普通用户切换 root 用户，需要输入密码
Password:                           #输入 root 用户的密码
[root@localhost ~]#                 #切换成功
```

2.1.3.目录的基本操作

- ls 命令

  功能：列出目录的内容。该命令类似于 DOS 下的 dir 命令。你想查看目录下的文件，就可以执行这个命令。就像 Windows 里面你双击任何一个目录，就可以看到那个目录下面的内容。

  语法：ls [选项] [目录或文件]

  常用选项：

  -a：显示目录下所有文件，包括隐藏文件

```
[root@localhost ~]# ls –a                  #列出当前目录下所有文件，包括隐藏文件
.  .bash_profile  .dbus  .lesshst  .ssh
..  .bashrc  .esd_auth  .local  .tcshrc
anaconda-ks.cfg  .cache  .history  .mozilla  .toprc
.bash_history  .config  .ICEauthority  .mysql_history  .viminfo
.bash_logout  .cshrc  initial-setup-ks.cfg  .pki  .vimrc
[root@localhost ~]#
```

  -l: 详细的显示目录下的文件信息

```
[root@localhost ~]# ls –l          #显示当前目录下文件的详细信息，不包括隐藏文件
total 8
-rw-------. 1 root root 1537 Dec 17  2018 anaconda-ks.cfg
```

```
-rw-r--r--. 1 root root 1731 Dec 17    2018 initial-setup-ks.cfg
[root@localhost ~]#
```

-R:递归显示目录下的文件

```
[root@localhost ~]# ls –R    /root        #显示/root 目录下所有文件和子目录的文件
.:
anaconda-ks.cfg    initial-setup-ks.cfg    testdir

./testdir:
test.txt
[root@localhost ~]#
```

目录中颜色的含义：

黑色→普通文件，蓝色→目录，绿色→可执行文件，红色→压缩文件，浅蓝色→链接文件

- pwd 命令

功能：此命令显示出当前工作目录的绝对路径。因为 Linux 是命令行，对于路径要求非常苛刻，有时候，在经过一系列操作后可能会出现我们忘记了当前工作目录的情况，这个时候，我们可以使用这个命令查看一下当前工作路径。

语法：pwd　[选项]

```
[root@localhost tmp]# pwd                    #显示当前工作目录的绝对路径
/tmp
[root@localhost tmp]#
```

- cd 命令

功能：改变工作目录，在 Windows 里面，我们可以利用鼠标随意切换盘符，例如从 C 盘到 D 盘，我们只需要退到最上一级，然后双击 D 盘，但是在 Linux 里面，我们只能通过命令来进行操作，就是利用 cd 命令，这个命令就是进入到某一个路径或者目录。可以退到上一级目录，也可以进入下一级目录。

语法：cd　[选项]　[目录路径]

```
[root@localhost ~]# cd /tmp          #将工作目录切换到/tmp 目录

[root@localhost tmp]# cd ~              # 切换到家目录，~代表家目录

[root@localhost ~]# cd -              # 回到上一次工作目录，-代表上一次工作目录

[root@localhost ~]# cd ..             #切换到当前目录的上一级目录，..表上一级目录
```

- mkdir 命令

功能：创建一个空目录。这个和 windows 里新建文件夹一个道理

语法：mkdir　[选项]　目录名称

常用选项：

-p：如果父目录不存在，则创建父目录

```
[root@localhost ~]# mkdir    dirname1 dirname2 dirname3    #  当前目录同时创建三个目录
[root@localhost ~]# mkdir -p /root/test1/test2/test3          #/root 目录下同时创建三级目录
├── test1
|    └── test2
|         └── test3
```

- rmdir 命令

  功能：删除一个空目录，对应 windows 里面删除，但是这里只能删空目录。

  语法：rmdir [选项] 目录名称

  常用选项：

  -p：删除目录后，如果父目录为空，则删除父目录

```
[root@localhost testdir]# rmdir dirname1 dirname2 dirname3    #  同时删除三个空文件
[root@localhost ~]# rmdir -p /root/test1/test2/test3    #删除 test3 目录，删除后如果 test2 为空，则删除 test2 以此内推
```

- 绝对路径与相对路径：

  绝对路径：从根目录开始进入到指定目录的完整路径。

```
[root@localhost ~]# cd /home/scg          #绝对路径的方式切换到/hom/scg 目录
```

  相对路径：从当前路径开始，进入目标路径。

```
[root@localhost ~]# cd ../home/scg        #相对路径的方式切换到/home/scg 目录
```

2.2. 文件管理

2.2.1 文件的基本操作

- touch 命令

  功能：新建空文件，或者改变文件的时间戳属性

  语法：touch [选项] 文件名

```
[root@localhost ~]# touch /root/test.txt       #在/root 目录中创建 test.txt 文件
```

- vim 命令

  功能：vim 是 Linux 中使用最为广泛的文本编辑器，它是 vi 命令的增强版。vim 编辑器中常见的有三种模式，分别是：命令模式，编辑模式，底行模式，进入 vim 后默认为命令模式。

  语法：vim [选项] 文件名

  命令模式：命令模式中支持众多快捷命令，来完成文本编辑时需要的常见功能，比如复制、粘贴、删除、查找等，下表是命令以及对应的功能：

快捷命令	功能
i,a	进入编辑模式(i 当前光标开始编辑，a 当前光标下一位开始编辑)

h, j, k, l	分别代表光标向：左下上下 移动一个字符
^	去到光标所在行的行首(第一个非空字符)
$	去到光标所在行的行尾
gg	去到第一行的行首
G	去到最后一行的行首
#G	指定去到第#行的行首(#为具体的数字)
yy	复制光标所在的行
#yy 或者 y#y	从光标所在行向下复制#行(#为具体的数字)
p	从光标所在行的下一行开始粘贴(如果按行复制的内容)
dd	删除光标所在的行
#dd 或者 d#d	从光标所在行向下删除#行(#为具体的数字)
dw	删除光标所在的单词
x	删除光标所在的字符
u	撤销操作
/关键字	从光标所在处向后查找关键字
:set nu	显示行号
:set nonu	取消显示行号
:set ic	忽略大小写
:%s/test/linux/g	查找替换，全文查找 test 并将其替换成 linux
:wq	保存退出
:q!	强制退出
Esc	切换到命令模式

以上就是 vim 编辑器中的常用命令，那么使用 vim 编辑文本至少需要哪几步呢？以下是一个完整的示例：

```
第一步：vim test.txt
第二步：按 i 进入编辑模式
第三步：编辑你想要编辑的内容
第四步：按 Esc 回到命令模式
第五步：按: (冒号)进入底行模式
第六步：输入 wq 保存退出
```

- rm 命令

  功能：删除一个目录中的一个或多个文件或目录，它也可以将某个目录及其下的所有文件及子目录均删除。这个命令切忌谨慎使用，如果连到了服务器上面，使用 root 账户，又进行这个操作，可能会误删除重要资料!!!

  语法：rm [选项] 文件名

  常用选项：

  -r：递归删除目录以及目录的内容

  -f：强制删除文件，删除过程中不予以二次提示

```
[root@localhost ~]# rm -rf /root/testdir/ #删除/root 目录下的 testdir 目录
```

- cat 命令

  功能：一次性查看文件的内容，适用于文件内容较少的情况

语法：cat [选项] 文件名

常用选项：

-n: 显示行号

```
[root@localhost ~]# cat -n test.txt #显示 test.txt 的内容并显示行号
    1  this is line1
    2
    3  this is line2
    4  this is line3
```

-b: 与-n 类似，但空白行不加行号

```
[root@localhost ~]# cat -b test.txt #显示 test.txt 的内容并显示行号(空白行不编号)
    1  this is line1

    2  this is line2
    3  this is line3
```

- less/more 命令：

功能: 查看文件内容，支持上下翻页，适用内容较多文件

语法：less [选项] 文件名

```
[root@localhost ~]# less test.txt #按页查看 test.txt 文件
```

more 命令用法与 less 命令类似，但默认情况下，more 不能向上翻页。

- head 命令

功能：从文件头部开始显示文件部分内容，默认 10 行

语法：head [选项] 文件名

常用选项：

-n：指定显示文件行数

```
[root@localhost ~]# head test.txt #查看 test.txt 文件的前 10 行

[root@localhost ~]# head -n 5 test.txt #查看 test.txt 文件的前 5 行
```

- tail 命令

功能：从文件尾部开始显示文件部分内容，默认 10 行

语法：tail [选项] 文件名

常用选项：

-n：指定显示文件行数

```
[root@localhost ~]# tail -n 20 test.txt #查看 test.txt 的后 20 行
```

-f：实时显示文件追加内容

```
[root@localhost ~]# tail -200f test.txt #显示 test.txt 后 200 行并实时监控此文件
```

- cp 命令

功能:文件或目录的复制,把某个文件或目录从一个地方复制一份到另外一个地方。这个和 windows 下面的复制操作一个道理。

语法：cp [选项] 源文件或目录 目标文件或目录

常用选项：

-R：递归复制

```
[root@localhost ~]# cp /root/test.txt /tmp # 将/root 下 test.txt 复制到/tmp 目录

[root@localhost ~]# cp -R /root/testdir/ /tmp #将/root 下的 testdir 目录复制到/tmp 目录，复制目录的时候需要加上-R 选项
```

- mv 命令

功能：为文件或目录改名或将文件或目录由一个目录移入另一个目录中。移动后原来的文件就没有了，类似 windows 里面的剪切。

语法：mv [选项] 源文件或目录 目标文件或目录

```
[root@localhost ~]# mv /root/test.txt /tmp 将/root 下 test.txt 移动到/tmp 目录
```

- 文件压缩与解压缩

.zip 格式：

```
[root@localhost ~]# zip test.zip test1 test2 #将 test1,test2 压缩并命名为 test.zip

[root@localhost ~]# unzip test.zip #将 test.zip 文件解压缩到当前目录
```

.tar.gz/tar.xz/tar.bz2 格式:

```
[root@localhost ~]# tar -zcvf test.tar.gz test1 test2 #将 test1,test2 压缩并命名为 test.tar.gz

[root@localhost ~]# tar -zxvf test.tar.gz #将 test.tar.gz 文件解压到当前目录

[root@localhost ~]# tar -jxvf test.tar.bz2 #将 test.tar.bz2 文件解压到当前目录

[root@localhost ~]# tar -Jxvf test.tar.xz –C /tmp 将 test.tar.xz 文件解压到/tmp 目录
```

- 管道(|)

功能：把一个命令的输出传递给另外一个命令做为输入

格式： 命令 1 | 命令 2 | 命令 3 ……

```
[root@localhost ~]# head test.txt | tail –n 2 # 显示 test.txt 的第 9、10 行
```

- cut 命令

功能：按行截取并显示指定文件内容

语法：cut [选项] 文件名

常用选项：

-d: 指定截取的字段分隔符

-f: 显示指定的字段内容

```
[root@localhost ~]# cat test.txt # 查看 test.txt 文本内容
Name;Mark;Percent
tom;69;91
```

```
jack;71;87
alex;68;98

[root@localhost ~]# cut   -d";"   -f1   test.txt   # 按；分割并显示第一个字段
Name
tom
jack
alex
```

- tr 命令

  功能：简单的替换命令，从标准输入中替换字符，并将结果写到标准输出

  语法：tr [选项] 字符串 1 字符串 2

```
[root@localhost ~]# cat test.txt | tr "m" "M"     #将 test.txt 中 m 替换成 M
```

- grep 命令

  功能：按行查找文件中符合条件的字符串

  语法：grep [选项] 文件名

  常用选项：

  -i: 忽略大小写

  -n: 显示行号

```
[root@localhost ~]#grep root /etc/passwd       #显示包含 root 的行

[root@localhost ~]#grep -n root /etc/passwd    #显示包含 root 的行并打印行号
```

  -v: 显示不包含匹配文本的所有内容

```
[root@localhost ~]#grep -v root /etc/passwd     #显示不包含 root 的行
```

  -A num：输出匹配行以及之后的 num 行

  -B num：输出匹配行以及之前的 num 行

```
[root@localhost ~]#ifconfig | grep -n -A3 -B2 "lo: "     #显示包含 lo:前 2 行以及后 3 行
```

- sed 命令

  功能：一种命令行编辑器，它能逐行处理文件(或输入)，并将输出结果发送到屏幕

  语法: sed [选项] 'AddressCommand' 文件名

```
[root@localhost ~]# sed -n '3,9p' /etc/passwd       #  只显示/etc/passwd 的第 3 到 9 行

[root@localhost ~]# sed 's/reboot/shutdown/g' /etc/passwd      #将/etc/passwd 文件中所有的 reboot 替换成 shutdown
```

- awk 命令

  功能：异常强大的文本分析工具，用于数据分析并生成报告

  语法：awk [选项] 'PATTERN{ACTION STATMENTS}' 文件名

```
[root@localhost ~]#tail   -5   /etc/fstab   |   awk   -F" "   '{print $2,$4}'   #将/etc/fstab 文件中后五行按照空格分隔，并显示第 2 和第 4 段
```

```
[root@localhost ~]#awk -F: '(NR>=2&&NR<=10){print $1}' /etc/passwd #将/etc/passwd 文件中第 2 到第 10 行安装：(冒号)分隔，并显示第 1 段
```

2.2.2 用户和组管理

- useradd 命令

  功能：新建用户(root 用户才能使用)

  语法：useradd [选项] 用户名

```
[root@localhost ~]# useradd testuser                    #创建一个叫 testuser 的用户
```

  用户创建完成后，默认是不能登录的，因为还没有设置密码，可以使用 passwd 命令给用户设置密码：

```
[root@localhost ~]# passwd testuser                     #给 testuser 修改密码
Changing password for user testuser.
New password:                                           #输入密码
Retype new password:                                    #再次输入密码
passwd: all authentication tokens updated successfully.
```

  用户的相关信息，保存在/etc/passwd 文件中，可以使用 cat /etc/passwd 命令查看：

```
[root@localhost ~]# cat /etc/passwd | grep testuser     #查看 testuser 用户信息
testuser:x:1001:1001::/home/testuser:/bin/bash
```

  /etc/passwd 中每一行代表一个用户的信息，可以按冒号将每行分割成七段，每一段的意思分别是：

  第一段：用户名（也被称为登录名）

  第二段：密码，映射到/etc/shadow 文件中；

  第三段：UID 用户 id；

  第四段：GID 用户所属主组 id；

  第五段：用户名全称，这是可选的，可以不设置；

  第六段：用户的家目录所在位置；

  第七段：用户默认 shell 的类型；

  如果要删除用户使用 userdel 命令即可，但是注意一定要加上-r 选项，不然用户的家目录等与用户相关的目录将会保留：

```
[root@localhost ~]# userdel -r testuser         #完全删除 testuser 用户
```

- groupadd 命令

  功能：用户账号可以分组，每个组里面有各自组下面的用户账号。例如 新人进入公司后，给他创建一个用户，加到对应一个安装组，这个组的权限就是可以进入存放各种工作软件的那个目录

  语法：groupadd [选项] 组名称

```
[root@localhost ~]# groupadd testgroup          #添加一个叫 testgroup 的组
```

  组添加完成后，组的相关信息存放在/etc/group 文件中。

  如果想往组内部添加和删除用户，可以使用 gpasswd 命令。

```
[root@localhost ~]# gpasswd -a scg testgroup    #将 scg 用户添加到 testgroup 组中

[root@localhost ~]# gpasswd -d scg testgroup    #将 scg 用户从 testgroup 组中移除
```

```
[root@localhost ~]# groupdel testgroup              #删除组，使用 groupdel 命令
```

- chown 命令
  功能：更改文件或目录的属主和属组
  语法：chown [选项] 用户或组 文件或目录

```
[root@localhost ~]# chown scg /root/test.txt   #将 test.txt 文件的属主修改为 scg

[root@localhost ~]# chown :scg /root/test.txt   #将 test.txt 文件的属组修改为 scg

[root@localhost ~]# chown -R   scg:scg /root/testdir   #将 testdir 目录的属主和属组都修改为 scg
```

2.2.3 文件访问权限

Linux 是一个多用户多任务操作系统。Linux 系统同时可以支持多个用户登录，每个用户对自己的文件设备有不同的权限，能够保证用户之间互不干扰。文件访问权限可以控制只有具备特定权限的人才能够进行这个操作。
上面说到了 Linux 系统里面最重要的一个概念---权限，对应有 root 用户和普通用户。从文件角度来说，每个文件和目录也有权限，分成三种：读，写，执行。

- 读---查看，只能看不能改，例如 cat
- 写--要修改, 例如 vim test.txt
- 执行--比如脚本的执行，可执行的文件，shell 脚本；用户进入某目录成为“可工作目录”至少需要具有执行权限
  用 ls -l 命令可以显示文件或目录的详细信息

```
[root@localhost ~]# ls –l test.txt        #查看 test.txt 的详细信息
-rw-r--r--   1   root   root   483997   Ju1 15 17:30 test.txt
```

最左边的一列为文件的访问权限，第一个字符指定了文件类型：在通常意义上，一个目录也是一个文件。如果第一个字符是横线，表示是一个非目录的文件（普通文件）。如果是 d，表示是一个目录。 其它位字符指定访问权限：横线代表空许可（无权限），r 代表只读，w 代表写，x 代表可执行。例如上例中：

```
-              rw-       r--       r—
普通文件   文件主   组用户   其他用户
```

- chown 命令
  功能：用于改变文件或目录的访问权限，有两种用法：一种是包含字母和操作符表达式的文字设定法，另一种是包含数字的数字设定法。
  语法(文字设定法)：chmod [who][+ | - | =][mode] 文件名...
  语法(数值设定法)：chmod [mode] 文件名...
  数值设定法，每种权限分别代表的意思是：r = 4，w = 2，x = 1，- = 0

以下是通常使用的数字值和其意义：

```
-rw------- (600) -- 只有属主有读写权限。
```

```
-rw-r--r-- (644) -- 只有属主有读写权限；而属组用户和其他用户只有读权限。
-rwx------ (700) -- 只有属主有读、写、执行权限。
-rwxr-xr-x (755) -- 属主有读、写、执行权限；而属组用户和其他用户只有读、执行权限。
-rwx--x--x (711) -- 属主有读、写、执行权限；而属组用户和其他用户只有执行权限。
-rw-rw-rw- (666) -- 所有用户都有文件读、写权限。这种做法不可取。
-rwxrwxrwx (777) -- 所有用户都有读、写、执行权限。更不可取的做法。
```

- sduo 命令

  功能：让普通用户执行一些或者全部的 root 用户才能执行的命令，root 用户在 /etc/sudoers 中设置了可执行 sudo 指令的用户。若其未经授权的用户企图使用 sudo，则会发出警告的邮件给管理员。

  语法：sudo 命令

```
[scg@localhost ~]$ sudo cd /root     #以 scg 普通用户的身份，切换到/root 目录下
[sudo] password for scg:   #输入 scg 的密码，没有在/etc/sudoers 配置，sudo 失败
scg is not in the sudoers file.    This incident will be reported.
[scg@localhost ~]$
```

2.2.4 文件查找

- find 命令

  功能：在指定目录下查找文件

  语法：find path -option 关键字

  常见选项：

  -name：通过文件名查找

```
[root@localhost ~]#find /home -name test.txt      #查找/home 下叫 test.txt 的文件
```

  -iname：通过文件名查找，但忽略大小写

```
[root@localhost ~]#find /home -iname test.txt    #查找/home 下叫 test.txt 的文件，忽略大小写
```

  -user：通过拥有者查找

```
[root@localhost ~]#find   /home   -user   scg   #查找/home 下拥有者为 scg 的文件
```

  -size：通过文件大小查找

```
[root@localhost ~]#find   /home   -size +10M      #查找/home 下文件大小超过 10M 的文件
```

  -mtime：通过文件最后修改时间查找

```
[root@localhost ~]#find /home -mtime +3 #查找/home 下文件最后修改时间超过 3 天的文件
```

  -type：通过文件类型查找

```
[root@localhost ~]#find    /home -type f            #查找/home 下所有普通文件

[root@localhost ~]#find /home  – name *.log  – type f – exec rm  – rf {} \;   #查找/home 目录下，以.log 结尾的普通文件并且删除
```

- whereis 命令

  功能：查找可执行文件（命令）查找命令的可执行文件、帮助文档、源码存放路径

  语法：whereis [选项] 命令

  常用选项：

  -b：只查找二进制文件

  -m：只查找手册页

  -s：查找源程序文件

```
[root@localhost ~]# whereis -b find     #查找 find 命令的存放路径
```

2.2.5 帮助命令

当你不知道一个命令是什么作用，或者想知道命令有多少种可用的参数，就可以使用帮助。帮助命令是官方的，英文比较多，对于大家英文要求比较高。

- help 命令：

  功能：查看 Shell 内部命令帮助信息

  语法：help 命令

```
[root@localhost ~]# help cd        #显示 cd 命令的帮助信息
```

- man 命令

  功能：查看命令的在线帮助手册

  语法：man [选项] 命令

```
[root@localhost ~]# man ls         #查看 ls 命令的帮助手册
```

2.3 系统管理

2.3.1 进程管理

我们在前言部分给大家讲解了进程的概念，现在我们来看看 Linux 系统里面关于进程管理的几个命令。

- ps 命令

  功能：显示当前运行进程的快照，是最基本同时也是非常强大的进程查看命令

  语法：ps 选项

  常用选项：

  -ef：显示系统所有进程信息

```
[root@localhost ~]# ps –ef                  #显示当前系统所有进程信息
```

```
[root@localhost ~]# ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 02:53 ?        00:00:08 /usr/lib/systemd/systemd --switched-root --sy
root           2       0  0 02:53 ?        00:00:00 [kthreadd]
root           3       2  0 02:53 ?        00:00:05 [ksoftirqd/0]
root           5       2  0 02:53 ?        00:00:00 [kworker/0:0H]
root           7       2  0 02:53 ?        00:00:00 [migration/0]
root           8       2  0 02:53 ?        00:00:00 [rcu_bh]
root           9       2  0 02:53 ?        00:00:08 [rcu_sched]
root          10       2  0 02:53 ?        00:00:00 [lru-add-drain]
```

- top 命令

功能：top 命令和 ps 命令的基本作用是相同的，显示系统当前的进程和其他状况；但是 top 是一个动态显示过程，即可以通过用户按键来不断刷新当前状态(数字键“1”按了后可以查看 CPU 每个核的使用情况，现在服务器系统都是多核，24 核，32 核，通过这个操作可以查看每个核的 CPU 使用情况，以便我们进行性能的分析)

语法：top [选项]

```
[root@localhost ~]# top                    #实时显示当前系统进程信息
```

```
top - 13:54:19 up 11:00,  2 users,  load average: 0.00, 0.01, 0.05
Tasks: 190 total,   1 running, 189 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us, 16.7 sy,  0.0 ni, 83.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  1863224 total,   972884 free,   325208 used,   565132 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  1308908 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
19316 root      20   0  162012   2236   1544 R   5.9  0.1   0:00.08 top
    1 root      20   0  128256   6968   4212 S   0.0  0.4   0:08.73 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.01 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:05.20 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    7 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 migration/0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
```

首行基本信息：

当前的时间；系统累计运行时间；登入用户数量，当前系统负载

第二行基本信息：

us---用户进程占用 cpu 资源的百分比；

sy---内核态进程占 cpu 资源的百分比；

ni---用户进程空间内改变过优先级的进程占 cpu 资源的百分比；

id---空闲 cpu 的百分比；

wa---等待输入输出的进程占 cpu 资源的百分比。

第三行内存相关的信息：

物理内存总量；使用的物理内存总量；空闲的物理内存总量；用在内核缓存的内存总量。

交换区内存总量；使用的交换区的总量；空闲的交换区总量；缓存的交换区总量

- free 命令

功能：命令相对于 top 提供了更简洁的查看系统内存使用情况

语法：free [选项]

```
[root@localhost ~]# free –h                #优雅的显示系统内存使用情况
```

```
[root@localhost ~]# free -h
              total        used        free      shared  buff/cache   available
Mem:           1.8G        317M        950M          9M        551M        1.2G
Swap:          2.0G          0B        2.0G
[root@localhost ~]#
```

free 命令显示中每一列非分别代表：

total---内存总数，物理内存总数；

used---已经使用的内存数；

free---空闲的内存数；

Swap--交换分区，虚拟内存(这个命令需要注意，正常情况应该是 0，如果不是 0，可能发生内存泄漏。因为正常情况下内存够用是不会使用虚拟内存的)

- kill 命令

功能：通过向进程发送指定的信号来结束进程，这个命令杀死进程，就像我们在 Windows 任务管理器里结束进程一样，在 Linux 里面这个命令就是：kill -9 进程号。

语法：kill [选项] 进程号

常用选项：

-s 指定需要送出的信号。既可以是信号名也可以对应数字。

-p 指定 kill 命令只是显示指定进程的 pid，并不真正送出结束信号。

-l 显示信号名称列表

-9 强行杀掉指定进程

```
[root@localhost ~]# /bin/kill -p httpd       #使用 kill 命令找到 httpd 的进程号
7836

[root@localhost ~]# kill -9 7836             #使用 kill 命令结束 httpd 服务
```

2.3.2 磁盘管理

- 系统服务

```
uname –a：查看操作系统名称及环境
cat /etc/redhat-release：查看系统的具体版本(适用于红帽系发行版)
hostname：查看操作系统名称
```

- 系统磁盘

```
df  -h：查看系统磁盘使用情况
du  -h test.txt：查看 test.txt 文件占用磁盘情况
```

- 系统启动

```
shutdown –h now : 系统关机
shutdown –r: 重启系统
reboot：同 shutdown –r，重启系统
```

- 网络相关

```
ping ipaddr : 检查与被测主机网络是否通畅
netstat –tupln：获取主机端口占用情况
```

- date 命令

功能：显示或设定系统的日期与时间，date 命令中有很多时间选项，具体参考 date 命令帮助文档

语法：date [选项]

```
[root@localhost ~]# date +"%Y-%m-%d"            #格式输出年月日
```

2.3.3 软件安装

- 源码安装：把被安装软件源码获取到后，一般只需要三步就能完成源码安装
  第一步：配置

```
./configure --prefix=自定义安装路径   执行“./configure”命令为编译做好准备；
```

  第二步：编译

```
make   执行“make”命令进行软件编译
```

  第三步：编译安装

```
make install  执行“make install”完成安装；
```

  例如：/root 目录中已存在源码文件 lrzsz-0.12.20.tar，一下是详细的安装步骤：

```
[root@localhost ~]# tar -xvf lrzsz-0.12.20.tar   #解压文件，会生成 lrzsz-0.12.20 目录

[root@localhost ~]# cd lrzsz-0.12.20     #切换工作目录

[root@localhost lrzsz-0.12.20]# ./configure --prefix=/usr/local/lrzsz   #配置信息，指定安装路径为/usr/local/lrzsz

[root@localhost lrzsz-0.12.20]# make                #编译

[root@localhost lrzsz-0.12.20]# make install       #编译安装
```

- (rpm)二进制包安装
  功能：通过 rpm 包管理工具，安装 rpm 二进制文件
  语法：rpm [选项] 包名
  常用选项：
    -ivh：安装软件
    -qa：查询所有已安装文件
    --prefix：指定包安装路径
    -e：卸载软件

```
[root@localhost ~]# rpm -ivh zsh-5.0.2-31.el7.x86_64.rpm   #安装 zsh 软件
```

- yum 命令
  功能：基于 RPM 包管理，能够从指定的服务器自动下载 RPM 包并且安装
  语法：yum [选项] 软件名

```
[root@localhost ~]# yum -y install httpd       #安装 httpd 服务

[root@localhost ~]# yum -y remove httpd         #下载 httpd 服务
```

3. Linux 项目应用案例

这个章节给大家展示 Linux 实际应用的例子，我们结合实战项目，由浅入深带着大家一起来进行操作，下面例子建议大家都实际进行操作，理解会更深刻。

3.1 安装 LAMP 环境

日常的工作中需要用到测试环境，因此搭建测试环境也是我们需要掌握的一项必备技能，那接下来我们就要完成在 CentOS 下安装 LAMP 环境。

LAMP 环境 ：Linux + Apache + MySQL + PHP，Linux 操作系统用的是 CentOS7，企业服务器一般为 Linux 操作系统，环境只需要安装 Apache（之前一直在说 Apache，其实 httpd 是服务而 apache 只是个商标,因此安装的是 httpd）,MySQL 和 PHP ，这里介绍使用 yum 来进行安装。Apache 本质上是一个 web 服务器，访问 ecshop http://localhost:80/ecshop，这是一个 http 请求，web 服务器也可以叫做 http server，就是专门处理 http 请求的，lamp 环境搭建大家安装 httpd 后(也就是 apache)，可以看到 /var/www/html 这个目录，这个目录里就是网站代码，如果通过 http://localhost:80/ecshop 这种方式来访问，http 请求就会进入到对应 html 目录下面的 ecshop 网站，对应返回的 Html 页面就是 http 的响应，用户最后看到的就是返回的 Html 网站页面。这个过程中，web 服务器也就是 apache 充当了一个容器的角色，里面放了我们的网站代码，然后让用户可以通过 http 请求找到网站并且返回对应请求给用户。整个过程都是 web 服务器在管理控制。

一般机器都带 yum 命令，并且 yum 包源都是可以用的，不用自己下载东西，直接 yum -y install。

- 安装 mysql 数据库

```
[root@localhost ~]# yum install mariadb mariadb-server mariadb-libs mariadb-devel
#安装 msyql 数据，CentOS7 中需要安装 mariadb 数据库，与 msyql 是一样的

[root@localhost ~]# systemctl start mariadb      #启动 mysql 服务

[root@localhost ~]# mysql -uroot –p                    #使用 msyql 客户端连接数据库
MariaDB [(none)]>
MariaDB [(none)]> grant all on *.* to '用户名'@'localhost' with grant option;   #用户授权
MariaDB [(none)]> flush    privileges；      #使授权生效

[root@localhost ~]# systemctl stop mariadb    #关闭数据库
```

- 安装 Apache

```
[root@localhost ~]# yum install httpd httpd-devel       #安装 httpd 服务

[root@localhost ~]# systemctl start httpd         #启动 httpd 服务

[root@localhost ~]# systemctl status httpd       #查看运行状态
```

- 安装 php

```
[root@localhost ~]# yum install -y php php-mysql php-gd php-ldap php-odbc php-pear
```

```
php-xml php-xmlrpc php-mbstring php-snmp php-soap curl curl-devel php-bcmath    #安装 php 已经常用的一些组件

[root@localhost ~]# systemctl restart httpd      #重启 httpd 服务

[root@localhost ~]# systemctl stop firewalld      #关闭防火墙
```

- 测试 LAMP 环境是否搭建成功

  在/var/www/html/新建个 test.php 文件，文件中写入<?php phpinfo();?>然后保存。再在客户端浏览器里打开 http:///IP /test.php，若能成功显示如下界面，则表示安装成功。

System	Linux localhost.localdomain 3.10.0-957.el7.x86_64 #1 SMP Thu Nov 8 23:39:32 UTC 2018 x86_64
Build Date	Nov 1 2019 16:05:03
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir	/etc/php.d

3.2 安装 ECSHOP

安装 ecshop 需要将软件安装包放的 httpd 服务的根目录中，然后做一些简单的配置即可，以下是详细步骤：

- 将软件包拷贝到/var/www/html 目录并解压：

```
[root@localhost ~]# /root/ECShopnew.zip /var/www/html/
[root@localhost ~]# cd /var/www/html/                #切换到/var/www/html 目录
[root@localhost html]# unzip ECShopnew.zip           #解压到当前目录下
```

- 在解压后生成的 ECShopnew 子目录中，找到 upload 目录并移动到/var/www/html 目录

```
[root@localhost ecmoban_V2.7.3_UTF8_20151127]# mv upload /var/www/html/

[root@localhost html]# mv upload ecshop    #将 upload 重命名为 ecshop
```

- 修改/var/www/html/ecshop/install/includes/lib_installer.php 文件，在文件最前面加

上一行：date_default_timezone_set('Asia/Shanghai');

```
<?php

/**
 * ECSHOP 安装程序 之 模型
 * ============================================================================
 * * 版权所有 2005-2012 上海商派网络科技有限公司，并保留所有权利。
 * 网站地址：http://www.ecshop.com；
 * ----------------------------------------------------------------------------
 * 这不是一个自由软件！您只能在不用于商业目的的前提下对程序代码进行修改和
 * 使用；不允许对程序代码以任何形式任何目的的再发布。
 * ============================================================================
 * $Author: liubo $
 * $Id: lib_installer.php 17217 2011-01-19 06:29:08Z liubo $
 */
date_default_timezone_set('Asia/Shanghai');
if (!defined('IN_ECS'))
{
    die('Hacking attempt');
<var/www/html/ecshop/install/includes/lib_installer.php" [dos] 1028L, 31275C 15,1
```

- 给 ecshop 目录附加所有权限

```
[root@localhost html]# setenforce 0                    #临时关闭 selinux
[root@localhost html]# chmod -R 777 ecshop
```

- 使用客户端访问 echop(http://ip/ecshop)

- 如果 ecshop 页面有报错信息，请按照文档《echop 页面报错修复.docx》修改。

3.3 安装 CRM

安装 crm 与安装 ecshop 步骤非常类似

- 将 crm 安装包解压到/var/www/html 目录中并添加权限

```
[root@localhost html]# mkdir crm          #/var/www/html 中创建 crm 目录
[root@localhost html]# cp /root/v0.5.5.zip /var/www/html/crm/    #将 crm 安装包拷贝到/var/www/html/crm 目录中
[root@localhost html]# cd crm
[root@localhost crm]# unzip v0.5.5.zip       #将安装包解压
[root@localhost crm]# cd ..
```

```
[root@localhost html]# chmod -R 777 crm        #给 crm 目录添加权限
```

- 使用客户端访问 echop(http://ip/crm)，将会看到如下配置页面：

3.4 实战项目命令详解

- 查看 CRM 日志

每个项目都会在某个目录下面存放日志文件，在 CRM 工程目录下面有一个 App，App 目录下面有一个 Runtime 目录，这个目录下就是随时访问项目都会写入一些日志。

```
[root@localhost Runtime]# pwd
/var/www/html/crm/App/Runtime
[root@localhost Runtime]# ls
Cache  Data  Logs  Temp
[root@localhost Runtime]#
```

第一次访问 CRM 后需要初始化，初始化后，我们新建一个线索，这个时候，系统会写入一个标明日期的日志文件（这里的时间是系统时间，通过命令 date 查看，不是 windows 电脑时间）。19_11_25.log 这个日志文件是每天生成一次，比如 0 点有人新建线索就会触发系统写日志，那就会新建一个日志文件，新建了这天任何操作都直接写入这个日志文件。

我们测试过程中可以执行命令 tail -f 19_11_25.log (写日志是不断写到行尾，所以查看文件最后内容就是最新的日志)动态查看日志写入。大家可以看看日志，这个操作具体做什么事情一目了解。这里大量都是 sql 操作，实际工作中，这样打开日志是一个好习惯，任何出错在日志里面都非常清楚，可以直接截图日志发给开发人员排查问题。

```
[root@localhost Runtime]# cd Logs/
[root@localhost Logs]# ls
19_11_25.log
[root@localhost Logs]# tail -f 19_11_25.log
ADD  `unit_price` FLOAT( 9, 2 ) NOT NULL COMMENT  '单价' AFTER  `tax_rate`,
ADD  `subtotal_val` FLOAT( 9, 2 ) NOT NULL COMMENT  '小计和',
ADD  `discount_price` FLOAT( 9, 2 ) NOT NULL COMMENT  '其他费用'
ERR: Duplicate column name 'last_read_time'
 [ SQL语句 ] :
ALTER TABLE  `5kcrm_user` ADD  `last_read_time` VARCHAR( 500 ) CHARACTER SET utf8 COLLATE utf8_
general_ci NOT NULL COMMENT  '商机等最后阅读时间'
ERR: Unknown column 'qq' in '5kcrm_contacts'
 [ SQL语句 ] :
ALTER TABLE `5kcrm_contacts` CHANGE `qq` `qq_no` VARCHAR(50) CHARACTER SET utf8 COLLATE utf8_ge
neral_ci NOT NULL COMMENT 'qq';
```

- 查找日志

上一小节大家都知道了 crm 项目存放的日志路径，我们可以通过 find 命令按需查找符合条件的日志：

将 crm 中一个月之前的日志找出来并且删除：

```
[root@localhost ~]# find /var/www/html/crm/App/Runtime/Logs -name "*.log" -mtime +30 -exec rm -rf {} \;
```

将 crm 中日志文件大于 100M 的找出来并且删除：

```
[root@localhost ~]# find /var/www/html/crm/App/Runtime/Logs -name "*.log" -size +100M -exec rm -rf \ ;
```

- 查看端口占用情况

之前的 crm、ecshop 都是部署在 LAMP 环境中，会启动 httpd 服务和 mariadb 服务，有的时候我们可能会遇到启动服务失败，报错说端口号被占用，我们就需要查看系统当前端口占用情况，使用 netstat 命令查看，如果能够看到指定的端口，说明此端口已经被占用：

```
[root@localhost ~]# netstat -tupln | grep httpd
tcp6       0      0 :::80                   :::*                    LISTEN      7876/httpd

[root@localhost ~]# netstat -tupln | grep mysqld
tcp        0      0 0.0.0.0:3306            0.0.0.0:*               LISTEN      8285/mysqld
```

- 配置防火墙

安装完成 crm、ecshop 之后，为了能够通过其他客户端访问，我们是把防火墙关闭了的，但实际工作中肯定是不能关闭防火墙的，所以我们需要手动去配置防火墙，将我们访问的 web 服务的 80 端口打开，执行如下命令即可：

```
[root@localhost ~]# firewall-cmd --zone=public --add-port=80/tcp –permanent      #永久打开 80 端口


[root@localhost ~]# systemctl restart firewalld          #重启防火墙
```

次时，防火墙已开启并且开放了 80 端口，在去访问 crm，同样能够访问到。

- 管道实例

如果我们想对 crm 中的日志多动态筛选，可以用 tail 命令和管道一起使用：

```
[root@localhost Logs]# tail -f /var/www/html/crm/App/Runtime/Logs/19_11_25.log |grep "ALTER"
```

```
[root@localhost Logs]# tail -f /var/www/html/crm/App/Runtime/Logs/19_11_25.log |grep "ALTER"
ALTER TABLE  `5kcrm_user` ADD  `last_read_time` VARCHAR( 500 ) CHARACTER SET utf8 COLLATE utf8_
general_ci NOT NULL COMMENT  '商机等最后阅读时间'
ALTER TABLE `5kcrm_contacts` CHANGE `qq` `qq_no` VARCHAR(50) CHARACTER SET utf8 COLLATE utf8_ge
neral_ci NOT NULL COMMENT 'qq';
```

3.5 安装禅道

首先确认 Linux 系统版本命令：getconf LONG_BIT，我的是 54 位：

```
[root@localhost ~]# getconf LONG_BIT
```

下载对应版本的禅道 (https://www.zentao.net/download/80178.html)：在下载页面找到适合的版本后，复制下载连接，在 Linux 中使用 wget 命令下载：

```
[root@localhost~]#wget
http://dl.cnezsoft.com/zentao/11.6.5/ZenTaoPMS.11.6.5.zbox_64.tar.gz
```

将安装包解压到/opt 目录：

```
[root@localhost ~]# tar -zxvf ZenTaoPMS.11.6.5.zbox_64.tar.gz -C /opt
```

修改禅道自带 apache、mysql 端口，为了不占用 server 上默认的 80、3306 端口：

```
[root@localhost ~]# /opt/zbox/zbox -ap 9000      #修改禅道自带 apache 端口

[root@localhost ~]# /opt/zbox/zbox -mp 9001      #修改禅道自带 mysql 端口

[root@localhost ~]# /opt/zbox/zbox start         #启动禅道

[root@localhost ~]# firewall-cmd --zone=public --add-port=9000/tcp –permanent    #永久打开 9000 端口

[root@localhost ~]# systemctl restart firewalld    # 重启防火墙
```

通过 http://ip:9000/即可访问成功，选择开源版(admin/123456)：

3.6 安装 JDK 并配置环境变量

环境变量的作用：环境变量（environment variables）一般是指在操作系统中用来指定操作系统运行环境的一些参数，如：临时文件夹位置和系统文件夹位置等。环境变量是在操作系统中一个具有特定名字的对象，它包含了一个或者多个应用程序所将使用到的信息。例如 Windows 和 DOS 操作系统中的 path 环境变量，当要求系统运行一个程序而没有告诉它程序

所在的完整路径时，系统除了在当前目录下面寻找此程序外，还应到 PATH 中指定的路径去找。用户通过设置环境变量，来更好的运行进程。

现在公司里面很多服务都是基于 Java,所以在 Linux 安装 JDK 和配置环境变量也是一个必备的技能。可能有些同学会问为什么要配置环境变量。我们通过在环境变量里面加入所有软件的安装路径，当我们想运行某一软件时双击其快捷方式或者在 DOS 界面输入软件名称，此时，计算机除了在其当前目录下寻找该软件的.exe 文件外，还在环境变量中搜索软件的路径，找到并运行。所以这里我们配置环境变量后，系统就会去对应配置路径下面搜寻 java 程序。

下载 JDK(去 Oracle 官网(https://www.oracle.com/index.html)找到下载连接),

新建/usr/java 目录，并将加载下来的安装包复制到这个目录：

```
[root@localhost ~]# cp jdk-8u191-linux-x64.rpm /usr/java
```

使用 rpm 工具安装 java：

```
[root@localhost java]# rpm -ivh jdk-8u191-linux-x64.rpm
```

验证安装是否成功：

```
[root@localhost ~]# /usr/java/jdk1.8.0_191-amd64/bin/java -version
```

安装完成后我们再配置环境变量,环境变量简单说是操作系统中一个特定名字的变量，它包含了应用程序所使用到的信息。编辑/etc/profile 文件加入以下内容：

```
export JAVA_HOME=/usr/java/jdk1.8.0_191-amd64
export JAVA_BIN=$JAVA_HOME/bin
export CLASSPATH=$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_BIN
```

编辑完成后，运行命令：

```
[root@localhost ~]# source /etc/profile
```

3.7 CRM 切换数据库

网站数据库切换也是我们经常会遇到的情况，下面以 CRM 系统为例给大家看看怎么切换数据库。大家首先需要知道，数据库配置信息是放到网站目录里的配置文件的。这个信息决定了网站使用哪个具体的数据库。还包含连接数据库的账户信息。

首先，找到配置文件，如下图：

```
[root@localhost Conf]# pwd
/var/www/html/crm/App/Conf
[root@localhost Conf]# ls
app_debug.php  config.php  db.php  debug.php  install.lock  tags.php  version.php
[root@localhost Conf]#
```

然后，vim db.php，如果要更改数据库，对应修改 ‘DB_NAME’=>’新数据库’

然后保存，退出

```php
<?php
return array(
'DB_TYPE'=>'mysql',
'DB_HOST'=>'localhost',
'DB_PORT'=>'3306',
'DB_NAME'=>'5kcrm',
'DB_USER'=>'root',
'DB_PWD'=>'123456',
'DB_PREFIX'=>'5kcrm_',
);
```

4. Linux Shell

4.1 Linux Shell 简介

Shell 是一种命令解释语言、程序设计语言，当一个用户登陆 linux 系统后，系统就会为该用户创建一个 shell 程序。

Shell 的版本：

Bourne Shell：是贝尔实验室开发的，unix 普遍使用的 shell，在编程方面比较优秀，但在用户交互方面没有其他 shell 优秀。

BASH：是 GNU 的 Bourne Again Shell，是 GNU 操作系统上默认的 shell，在 bourne shell 基础上增强了很多特性，如命令补全，命令历史表等等。

Korn Shell：是对 Bourne Shell 的发展，在大部分内容上与 Bourne Shell 兼容，集成了 C Shell 和 Bourne shell 优点。

C Shell：是 SUN 公司 Shell 的 BSD 版本，语法与 c 语言相似，比 bourne shell 更适合编程。

4.2 Shell 程序设计基础

4.2.1 Shell 输入输出

- echo 命令

  功能：用于字符串的输出

  语法：echo [选项] 字符串

  常用选项：

  -e：开启转义

  -n: 不输出换行符

```
[root@localhost ~]# echo -e "hello\tboy"    #\t 表示制表符
hello       boy
```

echo 命令对特殊字符敏感，如果要输出特殊字符，需要用\屏蔽其特殊含义。常用的特殊字符：双引号 “”　　反引号``　　反斜线\

```
[root@localhost ~]# echo " \" \" "          #输出双引号
```

小练习：

1 使用 echo 命令分别输出系统中 SHELL，PATH 变量的取值。

2 使用 echo 命令输出：what’s the means the sign “`v_v`” ?

3 使用 echo 命令输出如下格式的内容(注意对齐格式)：

id　　name　　msg

01　　mike　　“`hello`”

02　　john　　“hi”

- read 命令

功能：read 命令从键盘或者文件的某一行文本中读入信息，并将其赋给一个变量。如果只指定了一个变量，read 会把所有的输入赋给该变量，直至遇到第一个文件结束符或回车。

语法：read　[选项]　变量名 …

```
[root@localhost ~]# read name      #将输入赋值给 name 变量
hello i am superman
[root@localhost ~]# echo $name
hello i am superman

[root@localhost ~]# read -p"please input your name:" name      #添加提示语
please input your name:superman
[root@localhost ~]# echo $name
superman
```

- 重定向

标准输入/输出/错误：

标准输入: 对应文件描述符 0，是命令的输入，缺省键盘

标准输出: 对应文件描述符 1，是命令的输出，缺省屏幕或文件

标准错误: 对应文件描述符 2，是命令错误的输出，缺省屏幕或文件

常用文件重定向命令：

command > file：标准输出重定向到一个文件,错误仍然输出屏幕

command >> file：标准输出重定向到一个文件(追加)

command 1> file：标准输出重定向到一个文件

command 2 >> file：　标准错误重定向到一个文件(追加)

command >file 2>&1：标准输出和标准错误一起重定向到一个文件

command >>file 2>&1：标准输出和标准错误一起重定向到一个文件(追加)

command 1>file 2>file2：标准输出重定向 file，标准错误重定向到 file2 文件

command < file：以 file 做为文件标准输入

command < file1 >file2：以 file1 做为标准输入，file2 做为标准输出

- tee 命令

功能：把输出的一个副本输送到标准输出，另一个副本拷贝到相应的文件中如果想看到输出的同时，把输出也同时拷入一个文件，这个命令很合适，一般和管道符 | 结合起来使用。

语法：tee [选项] 文件

```
[root@localhost ~]# cat /etc/passwd | tee -a /root/test.txt   #查看/etc/passwd 文件，并将文件内容最近一份到/root/test.txt 文件中
```

4.2.2 Shell 后台执行命令

- & 命令

功能：当在前台运行某个作业时，终端被该作业占据；而当它在后台运行时，它不会占据终端，可以借助&命令把作业放到后台执行。注意（1 需要用户交互的命令不要放在后台执行，否则机器一直等待，2 后台程序在执行时，执行结果仍然会输出到屏幕，干扰我们的工作，建议将这样的信息重定向到某个文件）。

语法：command &

```
[root@localhost ~]# ls > out.file 2>&1 & # 将标准输出错误输出都定向到一个 out.file 的文件中
```

- at 命令

功能：at 命令允许用户向 atd 守护进程提交作业，使其在稍后的时间运行，这个时间可以是 10min 以后，也可能是几天以后。

语法：at [-f script] 时间

时间的格式可以很灵活，可以是精确时间，也可以是模糊时间，详细可以参考 at 命令的帮助手册：man at

创建一个定时任务，在 12 点 15 分执行：

```
[root@localhost ~]# at 12:15          #设定任务执行时间，开始编辑任务
at> echo "hello"                      #具体任务
at> <EOT>                             # ctrl+d 结束任务编辑
job 14 at Mon Nov 25 12:15:00 2019
```

明天下午三点运行/root/test.sh 脚本：

```
[root@localhost ~]# at -f /root/test.sh 3:00pm tomorrow
```

查看所有 at 任务：

```
[root@localhost ~]# at –l              #列出所有 at 任务，第一列为 job id
1    Mon Nov 25 12:15:00 2019 a root
2    Tue Nov 26 10:23:00 2019 a root
3    Tue Nov 26 15:00:00 2019 a root
[root@localhost ~]#
```

删除 at 任务：

```
[root@localhost ~]# at -r 1           #删除 job id 为 1 的任务

[root@localhost ~]# at –l             #查收删除成功
2    Tue Nov 26 10:23:00 2019 a root
```

```
3     Tue Nov 26 15:00:00 2019 a root
[root@localhost ~]#
```

- crontab 命令

功能：cron 是系统的调度进程，可在无人干预的情况下运行作业，通过 crontab 的命令允许用户提交，编辑或者删除相应的作业。每个用户都可以有一个 crontab 文件来保存调度信息，通过该命令运行任意一个 shell 脚本或者命令，在大的系统中，系统管理员可以通过 cron.deny 和 cron.allow 这两个文件来禁止或允许用户拥有自己的 crontab 文件。

crontab 格式： 分 小时 日 月 星期 要运行的命令

crontab 的域：

时间	取值范围
分钟	0～59
小时	0～23(0 表示子夜)
日	1～31
月	1～12
星期	0～6(0 表示星期天)

crontab 命令语法：crontab [-u user] 选项

常用选项：

-u：用户名 root 用户才能使用此选项

-e：编辑 crontab 文件

-l：列出 crontab 文件中的内容

-r：删除 crontab 文件

```
[root@localhost ~]# crontab –e              #创建 cron 定时任务

30  21  *  *  *  /apps/bin/cleanup.sh     #每天 21 点 30 分运行/app/bin 目录下的脚本 cleanup.sh，*号表示：每

*/15  8,18  *  *  *  /apps/bin/cleanup.sh  #每天的 8 点和 18 点每隔 15 分运行/app/bin 目录下的脚本 cleanup.sh，逗号表示：离散取值

0,30  18-23  *  *  *  /apps/bin/dbcheck.sh  #表示：每天的 18:00 到 23:00 之间每隔半小时运行脚本 backup.sh 脚本 ，-表示：连续取值
```

4.2.3 Shell 变量参数

- 系统变量

适用于所有用户进程，可以在命令行中设置，但用户注销时这些值将丢失，在 /etc/profile 中进行定义，传统上，所有环境变量都大写，用 export 命令导出。

设置变量并导出为环境变量：

```
[root@localhost ~]# export VAR=value          #创建环境变量
[root@localhost ~]# echo $VAR                 #查看变量
```

```
[root@localhost ~]# unset VAR                    #清除变量

[root@localhost ~]# env                          #查看当前所有环境变量
```

一般来讲，bourne shell 有一些预留的环境变量名，这些变量名不能做其他用途，通常在/etc/profile 中建立这些嵌入的环境变量，但这不绝对，取决于用户

常见 shell 变量：CDPATH; EXINIT; HOME; IFS; LOGNAME; MAIL; MAILCHECK; PATH; PS1; PS2; SHELL; TERMINFO; TERM; TZ

- 用户变量

在用户 shell 生命周期的脚本中使用，不同的用户可以定义各自的用户变量。

创建、清除用户变量，与环境变量类似，只是不需要 export 命令导出：

```
[root@localhost ~]#var=value          #创建用户变量
[root@localhost ~]# echo $var                 #查看变量
[root@localhost ~]# unset var                 #清除变量

[root@localhost ~]# set                       #查看当前所有变量
```

变量是可以直接修改的，如果想让一个变量变成只读变量，可以使用 readonly 命令：

```
[root@localhost ~]# var_name=value
[root@localhost ~]# readonly var_name       #变成只读变量
[root@localhost ~]# var_name=value1         #修改变量值失败
-bash: var_name: readonly variable
```

- 位置变量

位置变量属于只读变量，用于向 Shell 脚本传递参数，参数个数可以任意多个，每个参数访问参数前加$，脚本内部，第一个参数用$1 访问，第二部用$2 访问，以此内推，比如 test.sh 脚本如下：

```
[root@localhost ~]# cat test.sh          #查看脚本内容
#!/bin/bash
echo "第 1 个参数是：$1"
echo "第 2 个参数是：$2"
echo "第 3 个参数是：$3"
echo "第 4 个参数是：$4"
echo "第 5 个参数是：$5"
echo "第 6 个参数是：$6"
echo "第 7 个参数是：$7"
echo "第 8 个参数是：$8"
```

运行脚本并查看结果：

```
[root@localhost ~]# ./test.sh would you like to do it       #传递参数并运行脚本
第 1 个参数是：would
第 2 个参数是：you
第 3 个参数是：like
第 4 个参数是：to
第 5 个参数是：do
```

```
第 6 个参数是：it
第 7 个参数是：                                    #只传递了 6 个参数，所以 7，8 为空
第 8 个参数是：
[root@localhost ~]#
```

- 特殊变量

  同样是只读变量，它反映的是脚本运行过程中的控制信息

  常见的特殊变量：

变量名	含义
$#	传递到脚本的参数个数
$$	脚本运行的当前进程 id 号
$@	传入脚本的所有参数
$?	显示最后命令的退出状态，0 表示正确，其他任何值表示错误

4.3 Shell 程序设计流程控制

4.3.1 test 命令

Shell 中的 test 命令用于检查某个条件是否成立，它可以进行文件、字符和数值三个方面的测试。

语法：test condition　　或者 [　condition　]

- 文件测试

参数	含义
-e 文件名	如果文件存在则为真
-d 文件名	如果文件存在且为目录则为真
-f 文件名	如果文件存在且为普通文件则为真
-r 文件名	如果文件存在且可读则为真
-L 文件名	如果文件存在且为符号链接则为真
-s 文件名	如果文件存在且至少有一个字符则为真

判断 test.txt 是否为普通文件：

```
[root@localhost ~]# [ -f test.txt ]          #执行 test 命令
[root@localhost ~]# echo $?                  #查看 test 命令运行结果
0                                            #如果为 0 表示真，非 0 表示假
[root@localhost ~]#
```

- 字符串测试

参数	含义
=	等于则为真
!=	不相等则为真
\>	大于则为真
\<	小于则为真
-z 字符串	字符串的长度为零则为真

-n 字符串	字符串的长度不为零则为真

判断变量 name 是否为空串：

```
[root@localhost ~]# name=hello              #创建 name 变量
[root@localhost ~]# [ -z $name ]            #判断 name 变量是否为空串
[root@localhost ~]# echo $?
1                                           #name 变量不为空，结果为假
[root@localhost ~]#
```

- 数值测试

参数	含义
-eq	等于则为真
-ne	不等于则为真
-gt	大于则为真
-ge	大于等于则为真
-lt	小于则为真
-le	小于等于则为真

判断变量 num 是否等于 10：

```
[root@localhost ~]# num=9               #创建 num 变量
[root@localhost ~]# [ $num -eq 10 ]     #判断是否等于 10
[root@localhost ~]# echo $?
1                                       #变量 num 不等于 10，为假
[root@localhost ~]#
```

- 逻辑操作符

参数	含义
-a 或 &&	逻辑与，操作符两边均为真，结果为真，否则为假
-o 或 ||	逻辑或，操作符两边一边为真，结果为真，否则为假
!	逻辑否，条件为假，结果为真

判断 test.txt 是普通文件并且 test.txt 有读权限：

```
[root@localhost ~]# [ -f test.txt -a -r test.txt ]
[root@localhost ~]# echo $?
0                                       #判断结果为真
[root@localhost ~]#
```

使用符号的方式判断：

```
[root@localhost ~]# [ -f test.txt ] && [ -r test.txt ]   #使用符号与，注意与-a 用法区别
[root@localhost ~]# echo $?
0                                       #判断结果为真
[root@localhost ~]#
```

4.3.2 expr 命令

expr 命令一般用于整数值，也可以用于字符串。

语法：expr argument operator argument

- 做算术运算：

```
[root@localhost ~]#  expr  10 + 10      #注意空格
20
[root@localhost ~]# expr 300 /  6  /  5
10
[root@localhost ~]# expr 30  \*  3      #注意乘号必须用反斜线屏蔽其特定含义
90
[root@localhost ~]#
```

- 增量计数

expr 在循环中用于增量计算，首选，循环初始化为 0，然后循环加 1，常用的做法：从 expr 接受输出赋给循环变量

```
[root@localhost ~]# LOOP=0                  #创建 LOOP 变量
[root@localhost ~]# LOOP=`expr $LOOP + 1`       #LOOP 变量加 1
[root@localhost ~]# echo $LOOP
1
[root@localhost ~]# LOOP=`expr $LOOP + 1`       #LOOP 变量再加 1
[root@localhost ~]# echo $LOOP
2
[root@localhost ~]#
```

4.3.3 if 条件判断

- if 语法格式：

```
if 条件 1；then
    命令 1
elif 条件 2；then
    命令 2
      。
      。
      。
elif 条件 n；then
    命令 n
else
    命令
fi
```

注意：if 语句必须以 fi 来结束

例：以下脚本判断目录是否为空

```
[root@localhost ~]# cat test.sh         #查看 test.sh 脚本内容
#!/bin/bash
DIRECTORY=$1
```

```
if [ "`ls -A $DIRECTORY`" = "" ]; then
      echo "$DIRECTORY is indeed empty"
else
      echo "$DIRECTORY is not empty"
fi
[root@localhost ~]#
```

4.3.4 for 循环

- for 语法格式

```
for   变量名 in 列表
do
     命令 1
     命令 2
done
```

注意：命令 可为任何有效的 shell 命令和语句，变量名可以为任何单词 in 列表是可选的，如果没有列表，for 循环会使用命令行的位置参数 in 列表可以包含替换，字符串，文件名

例：使用数字列表

```
#!/bin/bash
for  loop1  in 1 2 4 5 6          #数字列表
do
    echo $loop1
done
```

例：字符串列表

```
#!/bin/bash
for  loop2   in “he is a tall man”            #字符串列表
do
    echo $loop2
done
```

例：替换列表

```
#!/bin/bash
for     loop3    in `ls`                      #替换列表
do
    echo $loop3
done
```

例：指定循环次数

```
#!/bin/bash
for m in {1..10..2}       # 从 1 到 10，每隔 2 循环
do
```

```
        echo "$m"
done
```

例：使用 seq 命令控制循环次数

```
#! /bin/bash
for b in $(seq 1 10 100)     #从 1 到 100，间隔 10 循环
do
        echo $b
done
```

4.3.5 while 和 until 循环

- while 语法格式

```
while   条件
do
    命令 1
    命令 2
    ……
done
```

注意：当条件为真的时候，才执行 while 语句中的命令

- until 语法格式

```
until   条件
do
    命令 1
    命令 2
    ……
done
```

注意：until 与 while 刚好相反，当条件为假的时候才执行语句中的命令

例：循环输入你追喜欢的电影名称，使用 ctrl-C 中断脚本的执行,整个循环中止：

```
[root@localhost ~]# cat test.sh       #查看脚本内容
#!/bin/bash
echo "Type    <Ctrl-C> to terminate"
echo    -n    "enter your most    liked film :"
while read    FILM
    do
        echo "Yeah ,great film the $FILM"
    done
[root@localhost ~]#
```

4.3.6 case 添加选择语句

- case 语法格式：

```
case   值   in
     模式 1)
          命令 1
          ……
          ;;
     模式 2)
          命令 2
          ……
          ;;
esac
```

注意：case 取值后面必须为 in，每个模式必须以右括号结束，取值可以为变量或者常数，找到匹配模式后，执行相关命令直到;;。 case 语句必须以 esac 来结束。

例：根据参数执行不同的分支

```
[root@localhost ~]# cat case.sh
#!/bin/bash
case $1 in
     start)
          echo "start....."
          ;;
     stop)
          echo "stop....."
          ;;
     status)
          echo "restart....."
          ;;
     *)                                    #如果都没有匹配到，走这条分支
          echo "Usage:$0 |start|stop|status|"
          ;;
esac
[root@localhost ~]#
```

4.3.7 break 和 continue

有时需要某些准则退出循环或者跳过循环步，就需要 break 和 continue 来实现

- break 语句：直接跳出循环
- continue 语句：直接跳出当前循环，继续下一次循环，本次循环 continue 语句之后的命令不再执行。

例：

```
[root@localhost ~]# cat test.sh
```

```
#!/bin/bash
while :
   do
        echo -n    "Enter any number [1..5]    :"
        read ANS
        case $ANS in
             1|2|3|4|5)
                  echo "great you entered a number between 1 and 5"
                  ;;
             *)
                  echo "wrong number..bye"
                  break                    #如果不再 1-5 之间，跳出循环
                  ;;
         esac
done
[root@localhost ~]#
```

5. Shell 项目应用案例

5.1 数据库备份

企业和网站最重要的资产就是数据，所以做好数据库备份是非常重要的。下面我们以 CRM 系统的数据库备份，给大家演示利用 shell 脚本一次性完成数据库备份。我们备份 CRM 系统的数据库。

先给大家展示整个脚本的内容，然后在逐行解释：

```
[root@localhost ~]# cat backup.sh      #查看备份数据库脚本
#!/bin/bash
DB_NAME=5kcrm
DB_USER=root
DB_PWD=123456
Now=$(date    "+%Y_%m_%d %H:%M:%S")
File=backup-$Now.sql
mkdir /root/backup
cd /root/backup
mysqldump    -u${DB_USER}    -p${DB_PWD}    ${DB_NAME} > "${File}"
echo 'this database is backup ok'
[root@localhost ~]#
```

第一行：DB_NAME---要备份的数据库的名字

第二行：DB_USER---数据库登录账号

第三行：DB_PWD---数据库登录密码

第四行：这里利用日期格式时间戳生成一个当前时间的字符串作为后续备份的 sql 文件的文件名（注意这个当前时间是服务器时间）

第五行：备份的文件的整个文件名，backup- 加上第四行的时间戳

第六行：新建一个空目录放备份的文件

第七行：进入这个空目录

第八行：mysql 备份的命令---mysqldump -u 用户名 -p 密码 数据库名 > 文件名(利用重定向)

第九行：打印备份成功

最后给脚本添加执行权限，并执行，就可以在/root/backup 目录中看到备份过后的文件：

```
[root@localhost ~]# chmod +x backup.sh      #添加执行权限
[root@localhost ~]# ./backup.sh             #执行脚本
this database is backup ok
[root@localhost ~]# ls /root/backup
backup-2019_11_25 15:44:17.sql
[root@localhost ~]#
```

如果数据库需要还原，使用 mysql 命令即可：

```
[root@localhost ~]#mysql -hlocalhost -uroot -p123456 5kcrm < backup-2019_11_25 15:44:17.sql
```

5.2 定时清理日志文件

网站运行过程中会不断产生日志文件，我们需要定期清理日志以便腾出更多可用磁盘空间。下面我们以 CRM 系统定时清理日志为例，给大家展示一下怎么利用定时任务执行脚本清理日志。

上面已经给大家介绍过了 CRM 日志目录，我们现在就是要清理这些所有.log 文件。

首先，在/root 目录下创建脚本并设置执行权限：

```
[root@localhost ~]# cat clean.sh
#!/bin/bash
path=/var/www/html/crm/App/Runtime/Logs
if [ -d ${path} ];then
    find ${path} -name '*.log' -exec rm –rf {} \;    #查找指定内容并删除
    echo "yes, we clean up all the log files"
else
    echo "there is no such directory"
fi
[root@localhost ~]#

[root@localhost ~]# chmod +x /root/clean.sh          #添加执行权限
```

然后我们把这个 shell 脚本加到定时任务中,设置执行时间：每周日早上 4 点

```
[root@localhost ~]# crontab –l          #查看已添加了定时任务
0 4 * * 0 /bin/bash /root/cleanup.sh
```

```
[root@localhost ~]#
```

5.3 自动搭建 lamp 环境

之前章节，我们手动搭建过 LAMP 环境，现在我们将命令集成在一个脚本里面，让其自动搭建 LAMP 环境：

```
[root@localhost ~]# cat lamp.sh       #查看脚本内容
#！/bin/bash
echo "start installing Apache..."
yum -y install httpd httpd-devel
echo "apache is ok"
echo "start apache"
systemctl start httpd
systemctl enable httpd                #设置 httpd 开机自启
echo "start installing mysql"
yum install mariadb mariadb-server mariadb-libs mariadb-devel
echo "MySQL is ok"
systemctl start mariadb
systemctl enable mariadb
yum install -y php php-mysql php-gd php-ldap php-odbc php-pear php-xml php-xmlrpc
php-mbstring php-snmp php-soap curl curl-devel php-bcmath
echo "PHP install ok"
systemctl restart httpd
systemctl stop firewalld
echo "yanzheng LAMP"
echo "<?php phpinfo(); ?>"    > /var/www/html/test.php
[root@localhost ~]#
```

我们逐行解释一下上面的脚本：

第一行：打印“start installing Apache...”
第二行：yum 方式安装 httpd 和 httpd-devel
第三行：打印“apache is ok”
第四行：打印“start apache”
第五行：启动 httpd 服务
第六行：设置 httpd 服务自启动
第七行：打印“start installing mysql”
第八行：yum 方式安装 mysql 相关包
第九行：打印“MySQL is ok"”
第十行：启动 mysql 服务
第十一行：开机自启 mysql 服务
第十二行：yum 方式安装 php 相关包
第十三行：重启 httpd 服务
第十四行：打印“PHP install ok”
第十五行：关闭 Linux 防火墙

第十六行：打印“yanzheng LAMP”
第十七行：新建 test 文件验证