

# 14. Linux基本使用和程序部署

## 本节目标

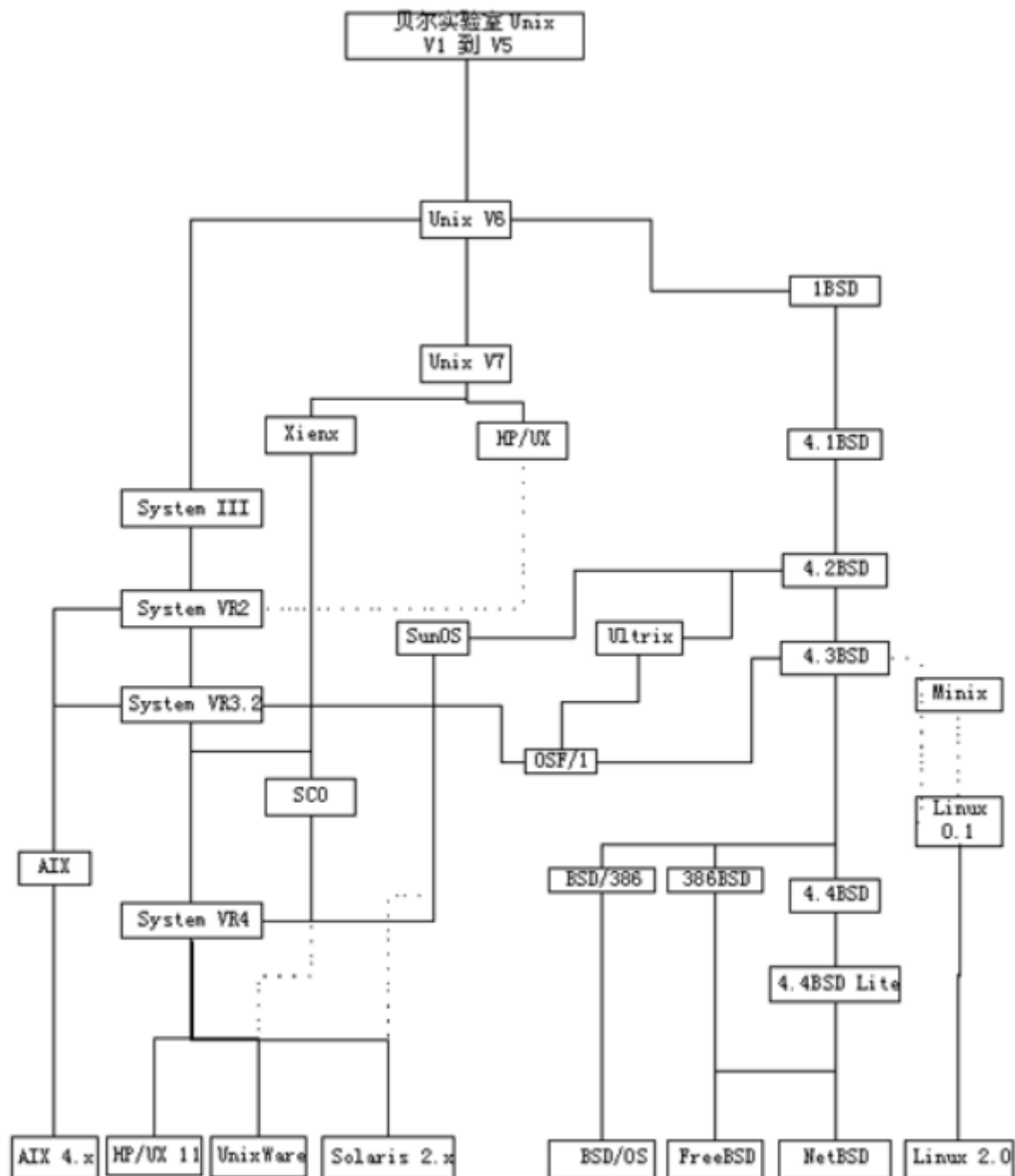
1. 了解Linux的常用命令
2. 了解Linux系统常见软件的安装, 以及独立在Linux服务器上部署项目
3. 学习SpringBoot多平台配置的方式

## 1. Linux 背景知识

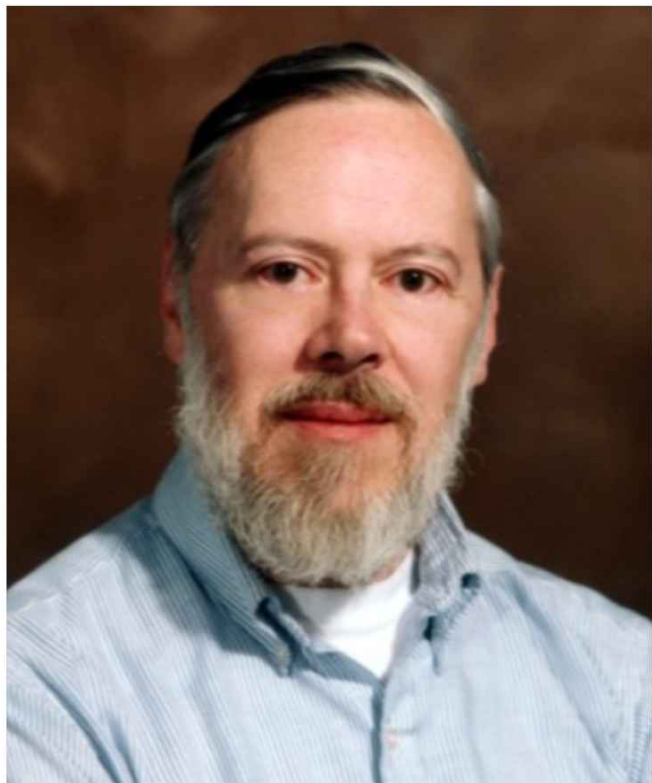
### 1.1 Linux 是什么

Linux 是一个操作系统. 和 Windows 是 "并列" 的关系.

**Unix & Linux 发展历程图**



1. 1969—1970 年, 贝尔实验室的 Dennis Ritchie (左) 和 Ken Thompson (右) 开发了 Unix 操作系统.



他们本来是想开发一个操作系统, 在开发的过程中, 发现现有的编程语言不太好用, 于是两个人便先开发了一个编程语言, 于是 **C 语言** 就诞生了.

Dennis Ritchie 在2011年就去天堂写代码了. Ken Thompson 后来在 google 又参与发明了 Go 语言.

2. Unix 火了之后, 衍生出很多的分支. 其中有一支为 "Minix". 由荷兰的Andrew S. Tanenbaum 教授开发. 源代码开放给大学教学和研究工作. (MINIX 含义为 "小型的UNIX")



3. 1991 年, 还在读大学的 芬兰人 Linus Benedict Torvalds (1969年12月28日出生, 当时 22 岁) 基于 Minix 的启发, 开发出了 Linux 的第一个版本.



值得一提的是, Linus 在开发 Linux 的时候, 觉得现有的 版本管理工具 不太方便. 于是开发了一个新的版本管理工具. 也就是现在最广泛使用的 git .

Linus 大佬现在 50 多岁了, 但是仍然在互联网上非常活跃. 经常会拍一些视频(IT届的活化石). 在B站上就可以找到. <https://www.bilibili.com/video/BV1qv411b79d?from=search&seid=3828654573220073441>

经过这么多年的发展, Linux 已经成为 **世界第一大操作系统**.

有的同学可能表示, Linux 都是世界第一大操作系统了, 我咋还没听说过? 其实安卓系统就是基于 Linux 进行的开发

虽然日常生活中, 大家对Linux的感知不高, 但是作为一个程序员, 在实际的工作中, 无论你从事哪方面的开发: 前端/后端/运维/算法/测试...多多少少都需要接触Linux开发

## 1.2 Linux 发行版

Linux 严格意义来说只是一个 "操作系统内核".

一个完整的操作系统 = 操作系统内核 + 配套的应用程序.

由于 Linux 是一个完全开源免费的内核, 因此有些公司/开源组织又基于 Linux 内核, 提供了不同的配套程序. 这就构成了不同的 "发行版".



企业中主要使用的发行版是 RedHat (红帽), CentOS(RedHat的社区免费版本)和ubuntu. 但是 RedHat 是收费的, CentorOS官方在2020年12月份通知: 2021年底停止维护CentOS8, 2024年6月30日停止维护CentOS7, 后续将无法获得官方升级和补丁. 出于以上考虑, 咱们课堂上使用的是Ubuntu.

## 1.3 Linux的优势

1. 开源(意味着免费, 便宜)
2. 稳定(Linux 可以运行很多年, 都不会发生重大问题)
3. 安全(Linux 只有管理员或者特定用户才能访问Linux内核)
4. 自由(不会被强加商业产品和服务)
5. 社区支持(Linux 在全球社区都非常活跃和使用广泛, 有很多志愿者在线帮大家解决问题)

## 1.4 关于 Linux 我们学习什么

### 基础命令

Linux 虽然也有图形化界面, 但是在 服务器 / 嵌入式设备上往往都是通过命令行的方式操作的. 因此学习 Linux 命令就是使用 Linux 的重要基础.

使用命令相比于使用图形界面主要有以下好处:

- 节省系统资源: 运行图形界面需要让系统付出一些额外的资源开销. 尤其是对于配置比较低的嵌入式设备, 这一点至关重要.
- 节省网络带宽: 如果通过网络访问服务器, 使用图形界面需要传输一帧一帧的图像, 而使用命令只需要传输简单的字符串.
- 便于批量执行任务: 可以通过一些 "脚本" 代码 (比如 Linux Shell) 来批量执行一些任务, 完成一些简单的编程工作. (比如定时备份文件, 删除文件等).

Windows 也有命令(也就是 cmd), 只是对于普通用户来说很少使用.

## 系统编程 & 网络编程

Linux 自身提供了一些 API, 供程序猿调用来完成一些更复杂的编程任务(比如文件操作, 多线程编程, socket 编程等).

但是由于 Java 跨平台的特性, 这部分功能已经被 Java 自身封装好了 (流对象, Thread 对象, Socket 对象等). 所以这部分内容我们不必再学习了.

## 部署 JavaWeb 项目

我们自己写的 web 程序, 要想让其他的用户能够访问, 就需要发布到服务器上. 这是我们接下来重点学习的内容.

所以, 对于 Java 程序员来说, 关于 Linux 重点学习 **基础命令** 和 **项目部署** 即可.

# 2. Linux 环境搭建

要想学习 Linux , 需要先有一个 Linux 的环境.

## 2.1 环境搭建方式

主要有四种:

1. 直接安装在物理机上. 但是Linux 桌面使用起来非常不友好. 所以不建议. **[不推荐]**.
2. 使用虚拟机软件, 将 Linux 搭建在虚拟机上. 但是由于当前的虚拟机软件(如 VMWare 之类的)存在一些 bug , 会导致环境上出现各种莫名其妙的问题, 比较折腾. **[非常不推荐]**
3. 使用 WSL (Windows Subsystem for Linux). 这个是 Windows 近几年开发的新功能, 在 Windows 系统内集成了一个 Linux. 但是目前这个技术还不够成熟. **[暂时不推荐]**.
4. 使用云服务器, 可以直接在 腾讯云, 阿里云或华为云 等服务器厂商处直接购买一个云服务器. **[推荐]**

腾讯云阿里云等为在校学生提供了优惠, 只要通过学生认证, 最低可以 10 块钱一个月. 还是非常划算的.

甚至在学习的过程中, 同学们可以 4, 5 个人共用一台服务器, 平均下来一个人一个月 2 块钱.

使用云服务器不仅环境搭建简单, 避免折腾, 同时还有一个最大的好处, 部署在云服务器上的项目可以直接被外网访问到, 这个时候就和一个公司发布一个正式的网站没有任何区别. 也就能让我们自己写的程序真的去给别人去使用.

能够在外网被访问是非常有意义的. 这样我们以后面试的时候就可以提前部署好项目, 现场给面试官演示了.


## 2.2 使用云服务器

我们以腾讯云为例, 其他的服务器厂商也是类似.


**注意:** 由于腾讯云官网一直在改版, 同学们实际看到的页面可能和课件上略有差别. 如果遇到某个环节搞不定, 随时可以咨询腾讯云的客服人员. (腾讯云的客服是非常热情的, 尤其是在你付钱之前~~).


1. 进入官方网站 <https://cloud.tencent.com/act/campus>, 并登录(直接在百度上搜 "腾讯云校园计划")

  
校园优惠套餐

  
云从业者培训及认证

  
在线学习中心 小程序

  
玩转人工智能

  
云开发实验室

校园优惠套餐 25岁以下免学生认证

云服务器 限量

1核  
CPU

2G  
内存

1Mbps  
带宽

50GB  
高性能云盘

✓ 套餐包含特价云服务器、域名(可选), 限购1台

✓ 完成 [学生认证](#) 获得2次购买续费资格, 需在本页面优惠续费

✓ 适用于建站、学习等轻量级web应用场景 [购买规则>](#)

☐ 加16元可选购1年.cn域名使用权

活动地域  

广州三区

北京三区

操作系统  

CentOS 7.6 64位

购买时长  

1年

6月

3月

2月

1月

120元 1398元

立即购买

2. 右侧的操作系统选择 Ubuntu 22.04 LTS. 购买时长根据需要选择(建议1年), 点击立即购买即可. 这个步骤需要实名认证, 否则会提示

温馨提示

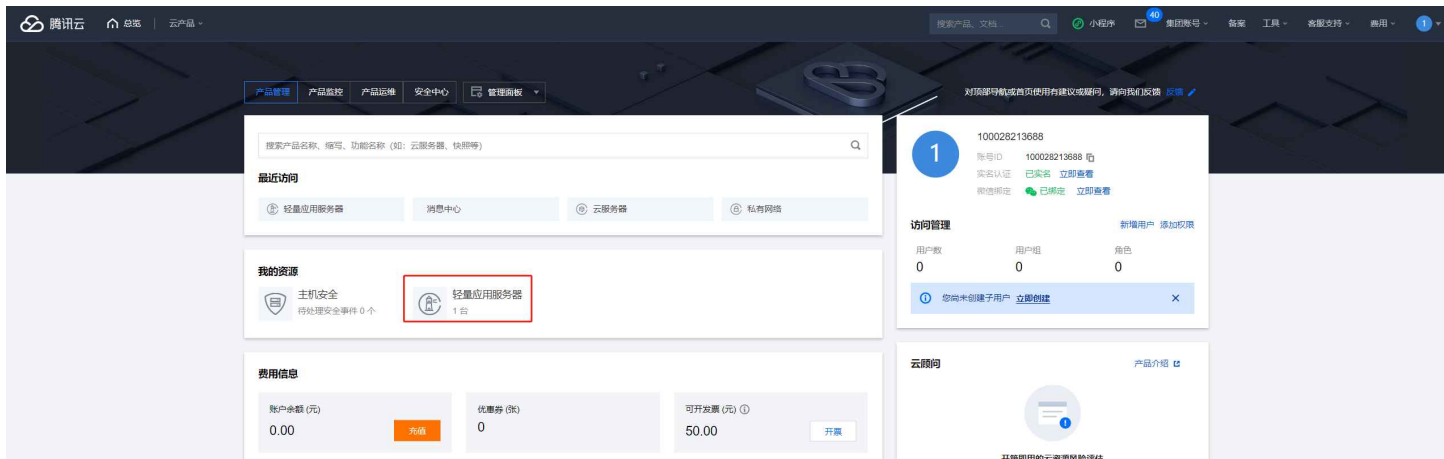
抱歉, 您未完成实名认证, 无法参与该活动。

立即认证

点击立即认证, 按照系统提示, 完成实名认证即可(认证速度很快).

3. 购买完成后, 可以在控制台中找到自己买的服务器. 点进去能够看到服务器的 IP 地址.





右下角为公网 ip 地址, 稍后我们会使用这个 ip 登陆服务器.

4. 设置 root 密码: 点击更多 -> 重置密码(这个环节可能需要手机短信验证).

root 密码建议设置的稍微复杂一些, 否则容易被黑客入侵.





小结:

在这个环节我们最重要的是得到三个信息:

1. 服务器的外网 IP
2. 服务器的管理员账户 (固定为 root)
3. 管理员账户密码(在腾讯云网站上设置的)

这三个信息是我们登陆到 Linux 上的必要条件.

## 2.3 使用终端软件连接到 Linux

### 2.3.1 什么是终端软件

终端软件是一类工具软件, 可以和远程的主机建立网络连接, 从而对主机进行一些操作.

常见的终端软件:

- XShell
- Putty
- MobaXTerm
- Iterm2
- SecureCRT
- .....

我们课堂上使用的是 XShell.

## 2.3.2 下载安装 XShell

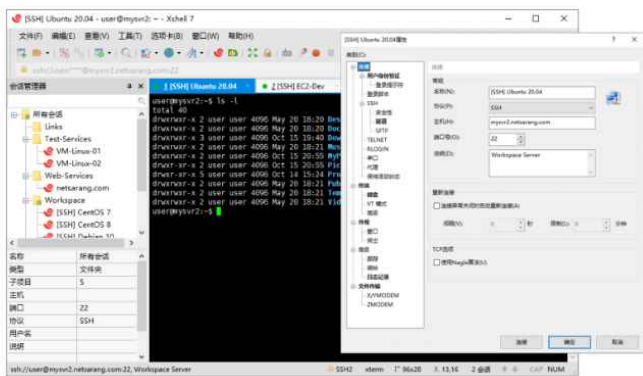
下载地址: <https://www.xshell.com/zh/free-for-home-school/>



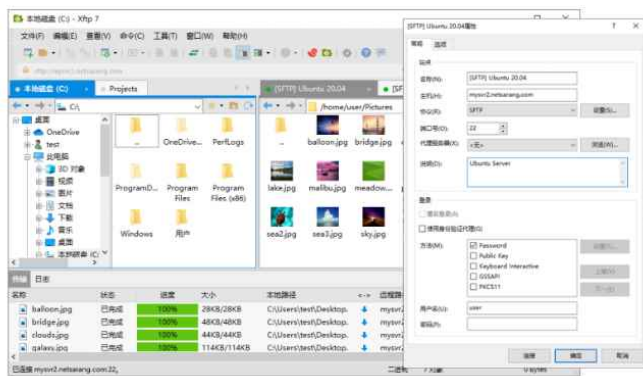
业界最强大的SSH客户端



轻松地通过网络传输文件



下载



下载

XShell 提供了两种授权方式.

- 企业版: 收费.
- 家庭/教育版: 免费.

我们选择 家庭/教育 版即可. 上面链接就是免费版



## 2.3.3 使用 XShell 登陆主机

在 XShell 终端下输入

```
1 ssh 服务器外网ip地址
```

ip 为腾讯云后台页面中看到的外网 IP.

如果网络畅通, 将会提示输入用户名密码. 输入即可正确登陆.

备注: 这里的用户名为 ubuntu, 密码是在最初购买服务器的时候设置的密码.

个别同学可能会出现无法输入密码的情况, 解决方案参考 <https://www.cnblogs.com/lemon-le/p/11168609.html>

登录失败的原因

1. 未输入用户名和密码(会提示输入用户名和密码)
2. IP输入错误
3. 确认安装的是Ubuntu (安装其他操作系统, 比如windows server, 确实无法登录)
4. 是否设置了初始密码
5. 服务器是否启动
6. 通过控制台登录
7. 以上都没问题, 咨询客服

### 关于XShell 下的复制粘贴

**复制:** ctrl + insert (有些同学的 insert 需要配合 fn 来按)

**粘贴:** shift + insert

ctrl + c / ctrl + v 是不行的.

可以重新设置快捷键 工具 => 选项 => 键盘和鼠标 => 编辑

## 3. Linux 常用命令

### ls

**语法:** ls [选项] [目录或文件]

**功能:** 对于目录, 该命令列出该目录下的所有子目录与文件。对于文件, 将列出文件名以及其他信息。

**常用选项:**

- -a 列出目录下的所有文件, 包括以 . 开头的隐含文件。
- -d 将目录象文件一样显示, 而不是显示其下的文件。如: ls -d 指定目录
- -k 以 k 字节的形式表示文件的大小。ls -alk 指定文件
- -l 列出文件的详细信息。
- -r 对目录反向排序。
- -t 以时间排序。
- -R 列出所有子目录下的文件。(递归)

**举例:**

```
1 ls -l
```

## pwd

**语法:** pwd

**功能:** 显示用户当前所在的目录

**举例:**

```
pwd
```

## cd

Linux系统中，磁盘上的文件和目录被组成一棵目录树，每个节点都是目录或文件。

**语法:** cd 目录名

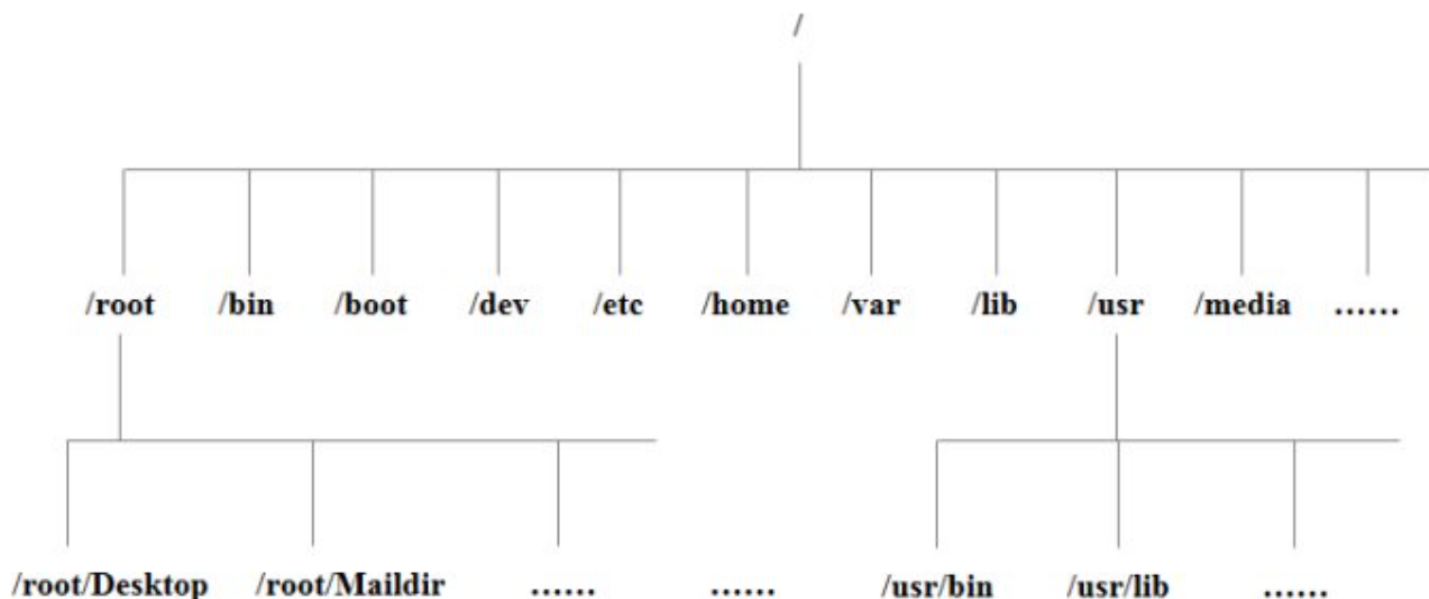
**功能:** 改变工作目录。将当前工作目录改变到指定的目录下。

**举例:**

```
1 # 返回上级目录
2 cd ..
3
4 # 进入用户家目录
5 cd ~
6
7 # 返回最近访问目录
8 cd -
```

## 认识 Linux 目录结构

Linux 是一个树形目录结构。



几个特殊的目录:

- / 称为根目录
- . 称为当前目录
- .. 称为当前目录的上级目录

### 绝对路径 vs 相对路径

形如: `/usr/share/tomcat/logs/` 以根目录开头的, 称为绝对路径.

形如: `./logs` 以 `.` 或者 `..` 开头的, 称为相对路径.

### 使用 tab 键补全

我们敲的所有的 Linux 命令, 都可以使用 tab 键来尝试补全, 加快效率.

### 使用 ctrl + c 重新输入

如果命令或者目录敲错了, 可以 ctrl + c 取消当前的命令.

## touch

**语法:** touch [选项]... 文件...

**功能:** touch命令参数可更改文档或目录的日期时间, 包括存取时间和更改时间, 或者新建一个不存在的文件。

**举例:**

```
1 touch test.txt
```

## cat

**语法:** cat [选项] [文件]

**功能:** 查看目标文件的内容

**常用选项:**

- -n 对输出的所有行编号

```
1 cat test.txt
```

## mkdir

**语法:** mkdir [选项] dirname...

**功能:** 在当前目录下创建一个名为 “dirname” 的目录

**常用选项:**

- -p, --parents 可以是一个路径名称。此时若路径中的某些目录尚不存在,加上此选项后,系统将自动建立好那些尚不存在的目录,即一次可以建立多级目录

**举例:**

```
1 # 递归建立多个目录
2 mkdir -p test/test1
```

## rm

**语法:** rm [-f-i-r-v] [dirName/dir]

**功能:** 删除文件或目录

**常用选项:**

- -f 即使文件属性为只读(即写保护),亦直接删除
- -i 删除前逐一询问确认
- -r 删除目录及其下所有文件

**举例:**

```
1 rm test.txt
```

**重要注意事项:**

千万不要运行 `rm -rf /` ,尤其是在公司的生产服务器上.

理解递归删除的过程:

先手动创建如下目录结构:

```
test
├── a
│   ├── a1
│   │   ├── 1.txt
│   │   └── 2.txt
│   └── a2
├── b
│   ├── b1`
│   │   ├── 1.txt
│   │   └── 2.txt
│   └── b2
└── c
```

使用 `rm -ri` 命令删除 `test`, 观察删除的顺序.

## cp

**语法:** `cp [选项] 源文件或目录 目标文件或目录`

**功能:** 复制文件或目录

**说明:** `cp`指令用于复制文件或目录, 如同时指定两个以上的文件或目录, 且最后的目的地是一个已经存在的目录, 则它会把前面指定的所有文件或目录复制到此目录中。若同时指定多个文件或目录, 而最后的目的地并非一个已存在的目录, 则会出现错误信息

**常用选项:**

- `-f` 或 `--force` 强行复制文件或目录, 不论目的文件或目录是否已经存在
- `-i` 或 `--interactive` 覆盖文件之前先询问用户
- `-r`递归处理, 将指定目录下的文件与子目录一并处理。若源文件或目录的形态, 不属于目录或符号链接, 则一律视为普通文件处理
- `-R` 或 `--recursive`递归处理, 将指定目录下的文件及子目录一并处理

**举例:**

```
1 cp test1.txt test2.txt
```

## mv



**语法:** mv [选项] 源文件或目录 目标文件或目录

**功能:**

1. 视mv命令中第二个参数类型的不同（是目标文件还是目标目录），mv命令将文件重命名或将其移至一个新的目录中。
2. 当第二个参数类型是文件时，mv命令完成文件重命名，此时，源文件只能有一个（也可以是源目录名），它将所给的源文件或目录重命名为给定的目标文件名。
3. 当第二个参数是已存在的目录名称时，源文件或目录参数可以有多个，mv命令将各参数指定的源文件均移至目标目录中。

**常用选项**

- -f：force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖
- -i：若目标文件 (destination) 已经存在时，就会询问是否覆盖！

**举例**

```
1 mv test1.txt test2.txt
```

## tail

**语法:** tail [必要参数] [选择参数] [文件]

**功能:** 用于显示指定文件末尾内容，不指定文件时，作为输入信息进行处理。常用查看日志文件。

**选项:**

- -f 循环读取
- -n<行数> 显示行数

**举例:**

```
1 tail -10 test1.txt
```

## vim

vim 是一个知名的文本编辑器. 使用 vim 可以进行文本编辑了.

vim 就相当于 Windows 的记事本. 只不过功能比记事本强大一些.

**1) 创建文件 / 打开文件**

```
1 vim [文件名]
```

## 2) 进入插入模式

vim 打开文件后默认是**普通模式**. 普通模式下键盘的按键表示一些特殊功能的快捷键. (例如按下 j 并不是输入字母 "j", 而是表示光标往下移动一行). 需要进入插入模式才能进行文本编辑.

使用 i 键可以进入到**插入模式**. (左下角提示 --INSERT-- ) 然后就可以像记事本一样正常编辑了.

## 3) 保存

在插入模式下不能保存文件, 需要先回到 普通模式 . 按下 `Esc` 回到普通模式.

在普通模式下输入 `:w` , 再按下回车, 即可保存文件.

## 4) 退出

在插入模式下不能退出, 需要先回到 **普通模式**.

退出时, 分以下情况:

- ① 文件未修改: 输入 `:q` , 再按下回车, 即可退出
- ② 文件修改并保存: 使用 `:wq` 同时执行保存和退出.
- ③ 文件修改, 但不希望保存: 使用 `:wq!` 进行强制退出

关于 Vim 的用法还有很多. 此处不做过多介绍了. 有兴趣的同学可以参考 vimtutor (直接在终端输入 `vimtutor` 即可进入官方教程) 或者相应书籍来进行深入学习.

## grep

**语法:** `grep [参数]... [文件]...`

**功能:** 用于查找文件中是否包含指定字符串, 并显示对应的行.

**选项:**

- `-n<行数>` 显示的行数
- `-w` 全字匹配. 要求整个单词都完全相同的结果才能匹配出来, 而不仅仅是一个单词的一部分.
- `-r` 递归查找. 可以搜索多级目录下的所有文件.
- `--color` 高亮查找到的结果
- `--include` 指定查找某些文件
- `--exclude` 指定排除某些文件

**举例:**

```
1 grep "hello" Hello.java
```

## ps

**语法:** ps [参数]...

**功能:** 用于查看当前系统上运行的进程

**选项:**

- a 显示一个终端的所有进程
- u 以用户为主的格式来显示程序状况
- x 显示所有程序, 不止是会话中的进程
- e 显示所有进程, 包括系统守护进程
- f 显示完整格式输出

**举例:**

```
1 ps aux    # 显示系统上所有的进程
2 ps aux | grep "进程名"
3 ps aux | grep "进程id"
4
```

## netstat

**语法:** netstat [参数]...

**功能:** 查看系统上的网络状态.

**选项:**

- -a 显示所有正在或不在侦听的套接字
- -n 显示数字形式地址而不是去解析主机、端口或用户名
- -p 显示套接字所属进程的PID和名称

**举例:**

```
1 netstat -anp
2 netstat -anp | grep "进程名"
3 netstat -anp | grep "端口号"
```

## 管道

管道是一种古老的 "进程间通信" 方式. 在 Linux 指令中可以使用 `|` 作为管道标记.

意思是将前一个指令标准输出的内容, 作为第二个指令的标准输入内容.

**举例:**

```
1 ps -ef|grep "java"
2 ps -ef|head -10
3 tail -f log.txt|grep "Exception"
```

## 4. 搭建 Java 部署环境

### 4.1 apt

apt(Advanced Packaging Tool), Linux软件包管理工具. 用于在Ubuntu、Debian和相关Linux发行版上安装、更新、删除和管理deb软件包.

大多数apt命令必须以具有sudo权限的用户身份运行.

#### apt常用命令

列出所有软件包

```
1 apt list
```

这个命令输出所有包的列表, 内容比较多, 可以使用grep命令过滤输出.

```
1 apt list |grep "java"
```

#### 更新软件包数据库

```
1 sudo apt-get update
```

apt实际上在可用软件包的数据库上工作. 如果数据库没有更新, 系统将不知道是否有更新的软件包可用. 这就是为什么在安装任何Linux系统之后, 第一件事应该是更新apt数据库

运行此命令时, 您将看到从各种服务器检索到的软件包信息.

如果切换到root用户, 命令前就不需要加 sudo了

切换root用户

```
1 sudo su
```

## 安装软件包

```
1 sudo apt install package_name
```

## 移除软件包

```
1 sudo apt remove package_name
```

remove命令将卸载给定的软件包，但可能会留下一些配置文件。如果要删除包含所有配置文件的软件包，请使用purge而不是remove

apt remove 和 apt purge 的区别

- `apt remove` 删除包的二进制文件，它留下了残留的配置文件。
- `apt purge` 删除与包相关的所有内容，包括配置文件。

如果弄乱了程序的配置，希望从系统中完全清除它的痕迹再重新开始，可以使用 `apt purge`

通常使用 `apt remove` 就足够了

## 4.2 JDK

### 1. 更新软件包

```
1 sudo apt-get update
```

## 执行结果

```
1 ubuntu@VM-24-3-ubuntu:~$ sudo apt-get update #更新软件包
2 Hit:1 http://mirrors.tencentyun.com/ubuntu jammy InRelease
3 Hit:2 http://mirrors.tencentyun.com/ubuntu jammy-updates InRelease
4 Hit:3 http://mirrors.tencentyun.com/ubuntu jammy-security InRelease
5 Reading package lists... Done
6
```

## 2. 安装openjdk

```
1 #查找jdk包
2 apt list |grep "jdk"
3
4 #安装jdk
5 sudo apt install openjdk-8-jdk
```

### 执行结果

```
1 root@bite:~# apt list |grep "jdk" #查找jdk
2 openjdk-8-doc/focal-updates,focal-updates,focal-security,focal-security 8u382-ga
3 openjdk-8-jdk-headless/focal-updates,focal-security,now 8u382-ga-1~20.04.1 amd64
4 openjdk-8-jdk-headless/focal-updates,focal-security 8u382-ga-1~20.04.1 i386
5 openjdk-8-jdk/focal-updates,focal-security,now 8u382-ga-1~20.04.1 amd64 [install
6 openjdk-8-jdk/focal-updates,focal-security 8u382-ga-1~20.04.1 i386
7 openjdk-8-jre-headless/focal-updates,focal-security,now 8u382-ga-1~20.04.1 amd64
8 openjdk-8-jre-headless/focal-updates,focal-security 8u382-ga-1~20.04.1 i386
9 openjdk-8-jre-zero/focal-updates,focal-security 8u382-ga-1~20.04.1 amd64
10 openjdk-8-jre-zero/focal-updates,focal-security 8u382-ga-1~20.04.1 i386
11 openjdk-8-jre/focal-updates,focal-security,now 8u382-ga-1~20.04.1 amd64 [install
12 openjdk-8-jre/focal-updates,focal-security 8u382-ga-1~20.04.1 i386
13 openjdk-8-source/focal-updates,focal-updates,focal-security,focal-security 8u382
14 ubuntu@VM-24-3-ubuntu:~$ sudo apt install openjdk-8-jdk #输入安装命令
15 Reading package lists... Done
16 Building dependency tree... Done
17 Reading state information... Done
18
19 ...
20 0 upgraded, 126 newly installed, 0 to remove and 154 not upgraded.
21 Need to get 102 MB of archives.
22 After this operation, 400 MB of additional disk space will be used.
23 Do you want to continue? [Y/n] Y #输入Y继续下一步
24
25 ...
26
27 done.
28 done.
29 Processing triggers for libglib2.0-0:amd64 (2.72.4-0ubuntu2.2) ...
30 Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
31 Setting up at-spi2-core (2.44.0-3) ...
32 Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.8+dfsg-1ubuntu0.2) ...
```

**注意:** 此处安装的是OpenJDK, OpenJDK是一个开源版本的JDK, 和 Oracle 官方的JDK 略有差别. 此处我们就使用 OpenJDK 即可. 安装 Oracle JDK 比较麻烦.

使用 `java -version` 验证是否安装成功.

如果提示 "java 命令找不到" 则说明安装失败.

```
1 ubuntu@VM-24-3-ubuntu:~$ java -version #查看jdk版本
2 openjdk version "1.8.0_382"
3 OpenJDK Runtime Environment (build 1.8.0_382-8u382-ga-1~22.04.1-b05)
4 OpenJDK 64-Bit Server VM (build 25.382-b05, mixed mode)
5
```

## 4.3 MySQL

### 1. 使用apt安装MySQL

```
1 #查找安装包
2 apt list |grep "mysql-server"
3
4 #安装mysql
5 sudo apt install mysql-server
```

执行结果

```
1 root@bite:~# apt list |grep "mysql-server" #查找mysql的安装包
2
3 WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
4
5 default-mysql-server-core/focal,focal 1.0.5ubuntu2 all
6 default-mysql-server/focal,focal 1.0.5ubuntu2 all
7 mysql-server-8.0/focal-updates,focal-security 8.0.35-0ubuntu0.20.04.1 amd64
8 mysql-server-8.0/focal-updates,focal-security 8.0.35-0ubuntu0.20.04.1 i386
9 mysql-server-core-8.0/focal-updates,focal-security 8.0.35-0ubuntu0.20.04.1 amd64
10 mysql-server-core-8.0/focal-updates,focal-security 8.0.35-0ubuntu0.20.04.1 i386
11 mysql-server/focal-updates,focal-updates,focal-security,focal-security 8.0.35-0u
12 ubuntu@VM-24-3-ubuntu:~$ sudo apt install mysql-server #输入安装命令
13 Reading package lists... Done
14 Building dependency tree... Done
15 Reading state information... Done
16 ...
17
18 After this operation, 243 MB of additional disk space will be used.
```



```
19 Do you want to continue? [Y/n] Y #输入Y确认
20
21 emitting matrix      : 100% |#####|
22
23 done!
24 update-alternatives: using /var/lib/mecab/dic/ipadic-utf8 to provide /var/lib/me
25 Setting up libhtml-parser-perl:amd64 (3.76-1build2) ...
26 Setting up libhttp-message-perl (6.36-1) ...
27 Setting up mysql-server (8.0.35-0ubuntu0.22.04.1) ...
28 Setting up libcgi-pm-perl (4.54-1) ...
29 Setting up libhtml-template-perl (2.97-1.1) ...
30 Setting up libcgi-fast-perl (1:2.15-1) ...
31 Processing triggers for man-db (2.10.2-1) ...
32 Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
```

## 2. 查看MySQL状态

```
1 sudo systemctl status mysql
```

### 执行结果

```
1 ubuntu@VM-24-3-ubuntu:~$ sudo systemctl status mysql
2 ● mysql.service - MySQL Community Server
3     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset:
4     Active: active (running) since Tue 2023-10-31 15:05:59 CST; 13min ago
5     Process: 70739 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=e
6     Main PID: 70747 (mysqld)
7     Status: "Server is operational"
8     Tasks: 37 (limit: 2220)
9     Memory: 369.2M
10    CPU: 2.360s
11    CGroup: /system.slice/mysql.service
12            └─70747 /usr/sbin/mysqld
13
14 Oct 31 15:05:58 VM-24-3-ubuntu systemd[1]: Starting MySQL Community Server...
15 Oct 31 15:05:59 VM-24-3-ubuntu systemd[1]: Started MySQL Community Server.
16
```

## 3. 设置密码

### 连接mysql服务器

```
1 sudo mysql
```

使用alter user 命令修改密码

```
1 ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '123456';
```

## 5. 部署 Web 项目到 Linux

### 5.1 什么是部署

工作中涉及到的 "环境"

- **开发环境**: 开发人员写代码用的机器.
- **测试环境**: 测试人员测试程序使用的机器.
- **生产环境**(线上环境): 最终项目发布时所使用的机器. 对稳定性要求很高.

把程序安装到生产环境上, 这个过程称为 "部署". 也叫 "上线".

一旦程序部署成功, 那么这个程序就能被外网中千千万万的普通用户访问到.

换句话说, 如果程序有 BUG, 这个 BUG 也就被千千万万的用户看到了.

部署过程至关重要, 属于程序开发中最重要的一环. 一旦部署出现问题, 极有可能导致严重的事故(服务器不可用之类的).

为了防止部署出错, 一般公司内部都有一些自动化部署工具(如 Jenkins 等). 当前我们先使用手工部署的方式来完成部署.

### 5.2 环境配置

程序正常运行, 需要保证环境和程序都要正确, 我们需要先设置环境

#### 数据准备

按照之前的数据库建表脚本, 在服务器上运行, 建立相同的表结构.

#### 程序配置文件修改

实际工作中, 开发环境, 测试环境以及生产环境的配置都是不一样的

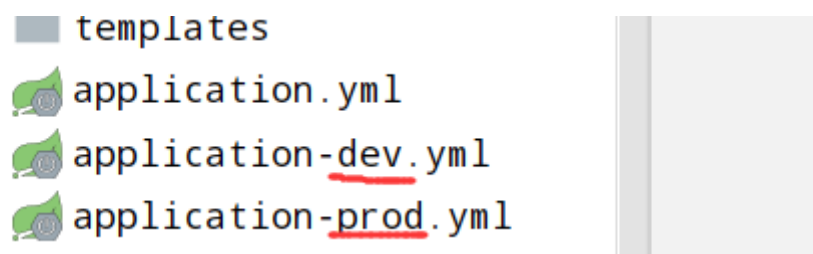
比如mysql的用户名和密码

我们可以针对不同的环境, 设置不同的配置

#### 多平台文件配置

针对不同平台创建不同的配置文件, 要求名字为application-XXX.yml或者application-XXX.properties

以下以application-XXX.yml为例



固定格式, 只有后面的字母可以修改

在配置文件里写不同的内容

application-dev.yml

```
1 spring:
2   datasource:
3     url: jdbc:mysql://127.0.0.1:3306/java_blog_spring?characterEncoding=utf8&use
4     username: root
5     password: root
6     driver-class-name: com.mysql.cj.jdbc.Driver
```

application-prod.yml

```
1 spring:
2   datasource:
3     url: jdbc:mysql://127.0.0.1:3306/java_blog_spring?characterEncoding=utf8&use
4     username: root
5     password: 123456
6     driver-class-name: com.mysql.cj.jdbc.Driver
```

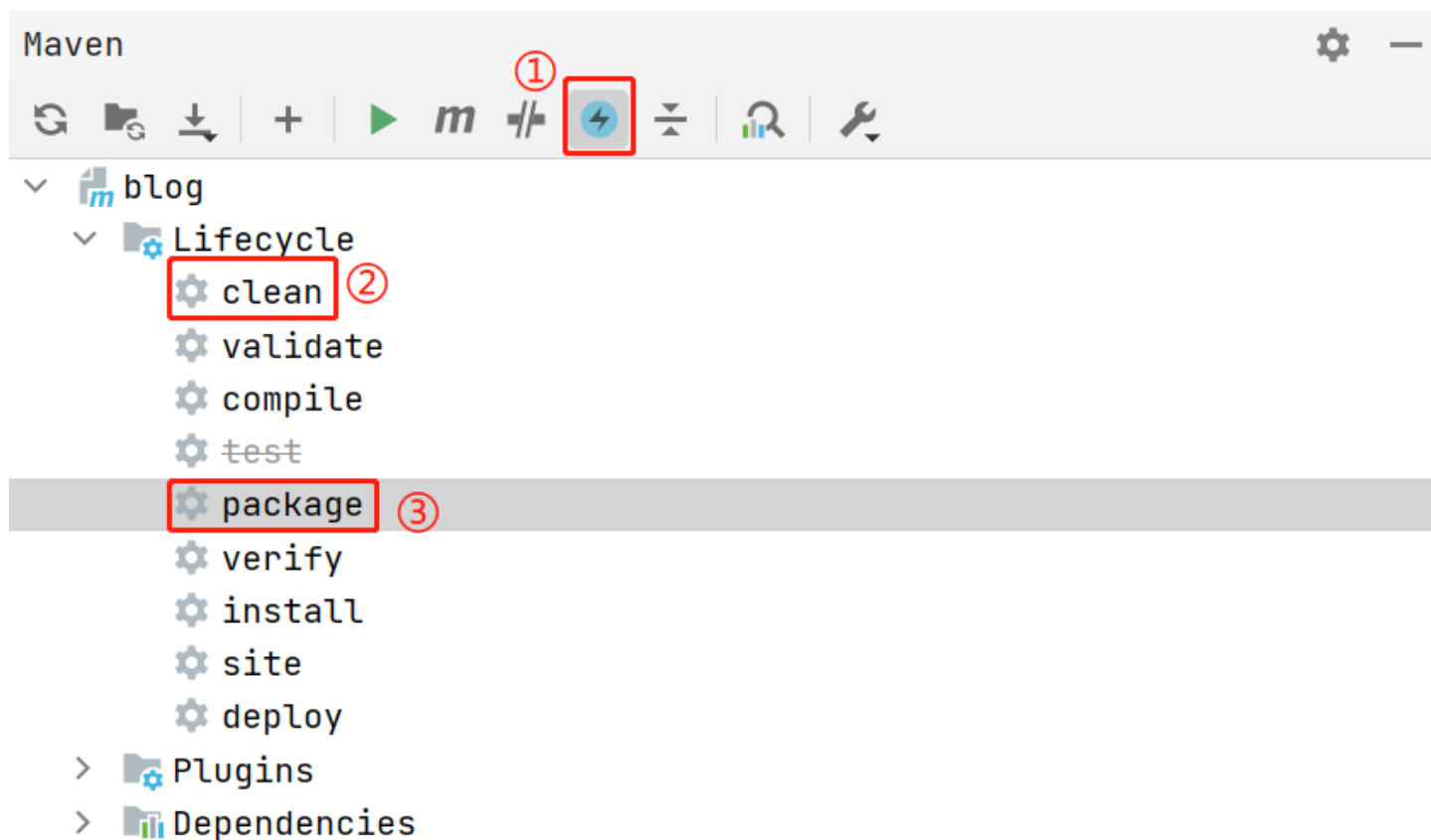
在主配置文件 `application.yml` 中指定配置文件, 并删除数据库相关配置

```
1 spring:
2   profiles:
3     active: prod
```

## 5.3 构建项目并打包

在本地使用maven进行打包

1. 如果Test代码中有与环境配置相关的操作(比如数据库相关的操作), 打包会失败, 点击下图①处的图标, 可以跳过测试
2. 点击clean->package



```
[INFO] Building jar: D:\Git\JavaEE课件相关资料\后端代码\bite-blog\target\blog-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.7.17:repackage (repackage) @ blog ---
[INFO] Replacing main artifact with repackaged archive
[INFO] -----
[INFO] BUILD SUCCESS 打包成功
[INFO] -----
[INFO] Total time: 5.067 s
[INFO] Finished at: 2023-10-31T18:30:58+08:00
[INFO] -----
```

## 5.4 上传Jar包到服务器, 并运行

### 1. 上传Jar包

直接拖动打好的jar包到xshell窗口即可完成文件的上传

```
ubuntu@VM-24-3-ubuntu:~/java$ ls
blog-0.0.1-SNAPSHOT.jar
```

Xshell 可以直接拖动文件到窗口, 达到上传文件的目的, 如果使用其他客户端, 不支持文件的上传, 需要借助lrzsz命令

### 1. 上传文件

```
1 sz filename
```

### 2. 下载文件

```
1 rz
```

执行该命令后, 在弹出框中选择要上传的文件即可, 上传的速度取决于网络.

问题: 如果执行上述命令之后, 提示 `Command 'XX' not found`, 表示当前云服务器未安装lrzsz命令, 需要先进行安装

### 3. 安装lrzsz

```
1 apt-get install lrzsz
```

## 2. 运行程序

```
1 nohup java -jar blog-spring-0.0.1-SNAPSHOT.jar &
```

nohup: 后台运行程序. 用于在系统后台不挂断地运行命令, 退出终端不会影响程序的运行.

**语法格式:**

```
1 nohup Command [ Arg ... ] [ & ]
```

**参数说明:**

Command: 要执行的命令。

Arg: 一些参数, 可以指定输出文件

&: 让命令在后台执行, 终端退出后命令仍旧执行

比如:

```
1 nohup java -jar blog-0.0.1-SNAPSHOT.jar >/logs &
```

Linux 可以通过 `>` 把需要输出的内容写到指定文件中. 这样的操作称为 "重定向".

### 3. 开放端口号

如果外网需要访问该服务, 需要先服务器防火墙开放对应的端口号

本着服务器安全的原则, 云服务器上的端口非必要不开启.

比如常见端口号: 数据库 3306 , Redis 6379, 尽可能避免开放, 而是采用其他方式来连接, 比如配置隧道的方式

以腾讯云服务器举例:

#### 1) 进入防火墙管理页面



#### 2) 添加规则

添加规则

对轻量应用服务器实例的入流量进行控制。

应用类型	来源	协议	端口	策略	备注
<div><div></div>自定义</div>	不填默认所有IPv4地址	TCP	8080	允许	tomcat

新增一条

您还可增加 93 条

确定

取消

端口号写需要开放的端口号, 多个端口号以逗号分割.

## 5.5 验证程序

1. 访问项目: `http://IP:Port/blog_login.html`

IP改为云服务器的IP, Port改为项目的端口号.

2. 按照项目的功能进行验证

- 验证账户注册登录
- 验证展示博客列表
- 验证新增博客
- 验证展示博客内容
- .....

## 5.6 常见问题

一个程序的正常运行, 需要程序的正确和环境正确. 同样的代码在Windows上可以运行成功, 不一定在Linux上运行成功. 不同的系统对代码的理解和支持是不同的. 比如Windows系统对MySQL不区分大小写, Linux区分大小写

服务不能正常访问的原因有很多, 主要分以下几方面

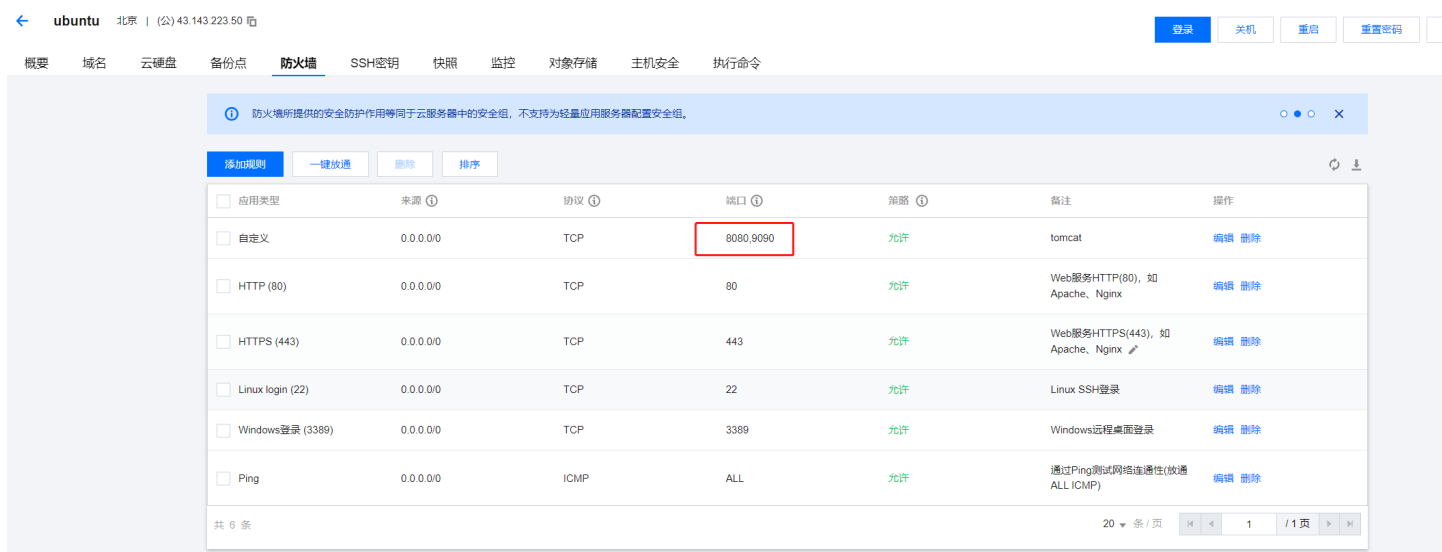
1. 服务未启动

- 使用 `ps -ef | grep java` 查看程序是否在运行
- 使用 `curl http://127.0.0.1:8080/blog_login.html` 看下是否有返回html页面, 如果有返回, 说明程序启动成功了, 考虑端口未开放
- 如果未启动成功, 需要查看对应的日志, 根据原因来分析.
  - 数据库不存在
  - 表不存在(区分大小写)
  - 数据库密码不正确
  - Jdk 安装版本不对, 或者未安装
  - mysql未设置密码
  - ....

2. http端口未开放

检查云服务器防火墙/安全组是否开放相应端口(如8080)





## 5.7 杀掉进程

如果我们需要重启服务, 或者重新部署等, 都需要先停止之前的服务.

### 1. 查看当前服务的进程

```
1 ps -ef|grep java
```

```
root@VM-24-3-ubuntu:/home/ubuntu/java# ps -ef|grep java
root      35443   34744  0 18:59 pts/2    00:00:12 java -jar blog-0.0.1-SNAPSHOT.jar
root      41144   34744  0 19:22 pts/2    00:00:00 grep --color=auto java
root@VM-24-3-ubuntu:/home/ubuntu/java#
```

上图35443 就是该服务的进程

### 2. 杀掉进程

```
1 kill -6 PID
```

也可以使用kill -9 强制杀掉进程

## 6. 总结

1. 连接Linux服务器的方式有很多, xshell只是其中一种. xshell是一个客户端, 而非服务器.
2. Ubuntu 软件管理工具是apt, 其他的linux发行版本软件包管理工具不同, 比如CentOS是使用yum来管理软件的
3. SpringBoot可以使用多个配置文件来完成不同平台的配置.

4. 在Window上可以运行成功的代码, 在Linux上不一定能运行成功.
5. 启动程序需要使用nohup后台运行, 需要停止服务时, 使用kill命令