目录

[语言 2](#_Toc495418790)

[Java 2](#_Toc495418791)

[C++ 2](#_Toc495418792)

[Js 2](#_Toc495418793)

[数据结构 2](#_Toc495418794)

[算法 2](#_Toc495418795)

[算法基础 2](#_Toc495418796)

[1bitmap 2](#_Toc495418797)

[机器学习 3](#_Toc495418798)

[1贝叶斯 3](#_Toc495418799)

[2svm 3](#_Toc495418800)

[web框架 3](#_Toc495418801)

[Spring 3](#_Toc495418802)

[Struct2 4](#_Toc495418803)

[协议 4](#_Toc495418804)

[工具 4](#_Toc495418805)

[Git 4](#_Toc495418806)

[Maven 6](#_Toc495418807)

[数据库 6](#_Toc495418808)

[数据相关操作 6](#_Toc495418809)

[常用命令 8](#_Toc495418810)

[操作系统 8](#_Toc495418811)

[VirtualBox安装centos学习 8](#_Toc495418812)

# 语言

## Java

Jar文件打开方式

$ Java –jar xxx.jar

## C++

## Js

# 数据结构

# 算法

## 算法基础

## 1bitmap

所谓的Bit-map就是用一个bit位来标记某个元素对应的Value， 而Key即是该元素。由于采用了Bit为单位来存储数据，因此在存储空间方面，可以大大节省。

来看一个具体的例子，假设我们要对0-7内的5个元素(4,7,2,5,3)排序（这里假设这些元素没有重复）。那么我们就可以采用Bit-map的方法来达到排序的目的。要表示8个数，我们就只需要8个Bit（1Bytes），首先我们开辟1Byte的空间，将这些空间的所有Bit位都置为0

然后遍历这5个元素，首先第一个元素是4，那么就把4对应的位置为1（可以这样操作 p+(i/8)|(0×01<<(i%8)) 当然了这里的操作涉及到Big-ending和Little-ending的情况，这里默认为Big-ending）,因为是从零开始的，所以要把第五位置为1

然后再处理第二个元素7，将第八位置为1,，接着再处理第三个元素，一直到最后处理完所有的元素，将相应的位置为1

然后我们现在遍历一遍Bit区域，将该位是一的位的编号输出（2，3，4，5，7），这样就达到了排序的目的

## 机器学习

### 1贝叶斯

### 2svm

# web框架

## Spring

1介绍

2原理

## Struct2

# 协议

# 工具

## Git

1初步介绍

git status

Untracked files 未跟踪文件

git add abc.txt

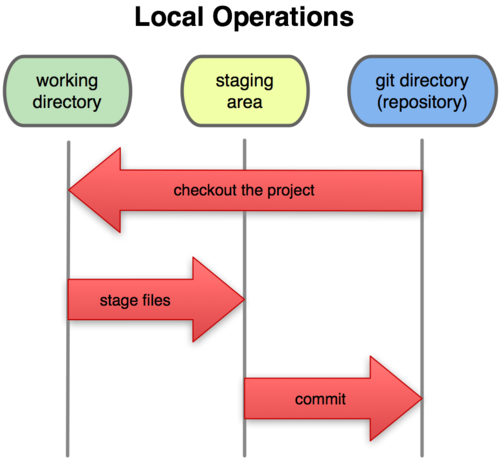
git status

Changes to be committed 在暂存区域生成了快照，等待被提交

Git commit 或git commit -m "first commit"

对于任何一个文件，在 Git 内都只有三种状态：‍‍‍‍**‍**‍**已提交（committed），已修改（modified）和已暂存（staged）**‍**‍。已提交表示该文件已经被安全地保存在本地数据库中了；已修改表示修改了某个文件，但还没有提交保存；已暂存表示把已修改的文件放在下次提交时要保存的清单中。**‍‍‍‍

由此我们看到 Git 管理项目时，文件流转的三个工作区域：**Git 的工作目录，暂存区域，以及本地仓库**



每个项目都有一个 **Git 目录**（译注：如果 git clone 出来的话，就是其中 .git 的目录；如果 git clone --bare 的话，新建的目录本身就是 Git 目录。），它是 Git 用来保存元数据和对象数据库的地方。该目录非常重要，每次克隆镜像仓库的时候，实际拷贝的就是这个目录里面的数据。

从项目中取出某个版本的所有文件和目录，用以开始后续工作的叫做**工作目录**。这些文件实际上都是从 Git 目录中的压缩对象数据库中提取出来的，接下来就可以在工作目录中对这些文件进行编辑。

所谓的**暂存区域只不过是个简单的文件，一般都放在 Git 目录中**。有时候人们会把这个文件叫做索引文件，不过标准说法还是叫暂存区域。

基本的 Git 工作流程如下：

* 在工作目录中修改某些文件。
* 对修改后的文件进行快照，然后保存到暂存区域。
* 提交更新(commit)，将保存在暂存区域的文件快照永久转储到 Git 目录中。

所以，我们可以从文件所处的位置来判断状态：如果是 Git 目录中保存着的特定版本文件，就属于已提交状态；如果作了修改并已放入暂存区域，就属于已暂存状态；如果自上次取出后，作了修改但还没有放到暂存区域，就是已修改状态。

Git push 真正推到远程代码服务器

## Maven

1初步介绍

mvn dependency:tree  查看依赖树

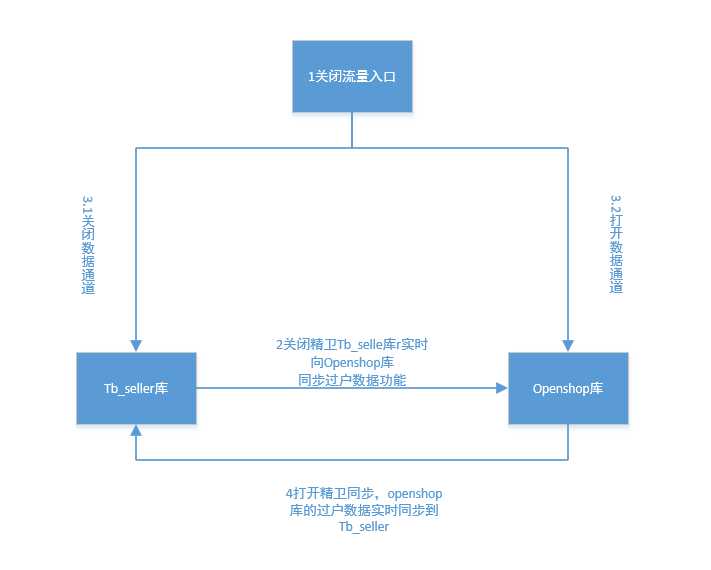
mvn dependency:tree -Doutput=\*.txt

windows 默认settings.xml地址

# 数据库

## 数据相关操作

1切DB



关闭精卫同步任务，完成数据迁移，分成以下几个步骤

1关闭精卫同步任务，开启老DB:TB\_seller向新DB:OpenShop的全量迁移，简称TB和Op

2 TB向Op迁移任务完成

3开始TB向Op的全量校验，

4完成校验，发现Op有46条缺失，12条差异，下载保存。手工抽取几条查看是否正确。

5全量订正Op，选择回源订正

6订正完成

7 Op向TB反向全量校验

8 完成全量校验，发现TB有107条数据缺失，即Op多了107条数据，应该是以前删除老数据库数据时，新数据库没有同步到删除的数据，下载保存，手工抽取几条查看正确

9 全量订正Op，此时Op是源数据。

10 订正完成

11   打开新的精卫同步任务，分为以下两个步骤

    12  时间1:25打开Op向Tb的精卫同步，将Tb作为备数据库

13  时间1:27-1:40预发分别测试过户和升级，数据流入新数据库Op，check发现能同步到老数据库Tb，即同步任务ok

# 常用命令

改完hosts刷新           ipconfig /flushdns

tail –f100 log/production.log

grep  -n 'test'  seller.log

$sed -n '234689,234900p' seller-center.log

sed -n '/2017-07-17 13:47:20/,/2017-07-17 13:47:25/p'  seller-center.log

# 操作系统

## VirtualBox安装centos学习

dky112358

1 知识点

Linux是/根目录，所以目录或文件名区分大小写

已.开头的是隐藏文件

相对路径/usr/log/sa和绝对路径../../usr/log/sa

2 常用命令

$ 当前登录用户

# root用户

命名+选项+参数

Hostname

Uanme or uname –r or uname -a  
firefox 直接打开火狐

Firefox & 后台运行火狐，命令行能继续其他操作

history 查看之前命令

!! 重复前一个命令

！字符 重复前一个已“”字符“”开头的命令

！num 按照历史记录的序号执行命令

！？abc 重复之前包含abc的命令

！-n 重复n个命令之前的那个命令

Ctrl+r 在历史记录中搜索命令

Esc . 按esc后按.键 重复调用前一个命令的参数

Su- 切换到root用户并使用新的运行环境

Su 切换到root

Sudo 使用管理员用户身份运行命令

Id 显示当前用户信息

Passwd 修改当前用户密码

Jobs bg fg管理后台作业

Jobs 当前在后台运行的所以作业

Sleep 5000 停止5000秒

Ctrl+z 暂停当前作业

Bg 1 sleep继续后台运行 1代表作业编号

Fg 1 sleep在前台运行

Pwd 显示当前工作目录（每一个shell或系统进程都有一个当前工作目录）

Touch 创建一个空白文件或者更新已有文件的时间

Ls –a 显示所有文件（包括隐藏文件）

Ls –l 显示详细信息

Ls –R 递归显示子目录结构

Ls –ld 显示目录和链接信息

File abc。Txt 查看文件类型

Cp 源文件 目标文件/目标目录 复制文件或目录

Cp –r 文件夹 目标文件/目标目录

Cp –rv 加v显示详细复制信息

或Cp –r –v

Mv 文件 目录 移动文件 如 mv a opt

Mv 文件 目录/新命名 如 mv a opt/b 将a移到opt下，重命名为b

mv a b a重命名为b，当前目录

rm xx删除当前某文件

rm –r dic 递归删除dic目录

rm –ri dic i交互式删除 每一个删除都询问

rm –rif dic 强制删除 不会有交互式询问 –f

mkdir 创建一个目录