我们这个maven课，我个人觉得最好的讲解思路是从依赖这块入手853769620

很多培训视频，上来做完一个demo之后，就是去讲解生命周期，插件，然后才是依赖

我觉得，我们用maven，看到的第一件事情就是依赖这块，怎么找到依赖，怎么声明依赖，依赖管理机制，依赖冲突（企业场景内容），先讲解了依赖这一块

依赖比较基础的一些内容都讲解完了，下面就要来进入比较重要的仓库这块，maven的依赖是从哪儿下载下来的？？？？

这个实际上是在公司里是非常重要的一块内容，因为实际上你平时去公司里开发，必须了解公司的一些maven仓库的内容和配置，否则你怎么配置你自己在公司里的开发环境呢？？

1、仓库的基本介绍

正常来说，maven会用仓库来存储所有的依赖，不会说是每个项目都放一份依赖这样子

如果我们不使用maven，那么我们就会在各种工程里建一个lib目录，里面放入这个工程的所有依赖，在打包的时候通常会用eclipse的fatjar这个插件，将工程打成一个胖jar，里面包含了所有依赖。

但是如果公司内有几十个，甚至上百个工程，那么每个工程都要维护一份依赖，第一非常浪费空间，第二非常麻烦，每次升级某个依赖，可能要将几十个工程里的lib包下的依赖，先删除旧版本，再放入新版本，很麻烦。

而maven的仓库，就是用来统一存放各种依赖的地方。哪怕是有几十个工程，但是每个工程如果有相同的依赖，那么那个依赖在仓库里只会存在一次，不会放在各个工程自己的lib包下。消除了重复，如果要升级某个依赖，直接在各个工程的pom.xml里升级依赖的版本即可，大家都会自动引用最新版本的依赖了。在仓库里所有依赖就放一次，然后每个工程在pom.xml里面声明对依赖的引用即可，打包的时候，可以将所有依赖打入发布包即可。

2、仓库的布局结构

自己windows本地的仓库，是在~/.m2/repository目录，就是maven的本地仓库

那么依赖在maven仓库里是以什么方式存放的呢？其实maven仓库的布局结构其实很简单，比如我们依赖了一个项目，它的坐标如下：

groupId：log4j

artifactId：log4j

version：1.2.15

那么这个依赖的jar包在maven仓库里的布局，就是存放在log4j/log4j/1.2.15/log4j-1.2.15.jar，log4j/log4j/1.2.15就是目录，log4j-1.2.15.jar就是依赖对应的jar包。

就这么简单，所以大家随时可以在仓库里根据坐标找到某个依赖的jar包。

所以你声明依赖之后，maven实际上就是根据依赖的坐标，先到本地仓库里面去根据坐标去一层一层的定位依赖所在的目录里面有没有这个jar包，如果没有的话，就是到远程仓库里面去根据同样的布局去找依赖的jar包。

3、一张图带你看懂maven的多层仓库架构

maven仓库的大类分成本地仓库和远程仓库两种，如果你声明了一个依赖，那么在构建打包的时候，先会去本地仓库里找，这个本地仓库的地址默认就在用户主目录下面的~/.m2/repository目录下面，里面各个依赖的存放路径就是跟上面说的那样；如果本地仓库找不到，那么就会去远程仓库找，默认是去maven自己的中央仓库里找，maven的中央仓库里几乎涵盖所有的依赖；然后会将中央仓库里的依赖下载下来放到本地仓库，缓存起来，供下次使用。

中央仓库是属于一种远程仓库，但是不只是有中央仓库一种远程仓库而已

远程仓库还有私服和其他远程仓库，私服指的是在公司内部的局域网内，架设一个服务器，上面放一个公司内私有的仓库，就是私服；其他远程仓库，就是其他公司开放出来的仓库，比如java.net maven仓库，jboss maven仓库，等等。

（1）本地仓库

本地仓库，windows默认路径是~\.m2\repository，linux默认路径是/home/root/.m2/repository

可以修改本地仓库的路径，在~/.m2/settings.xml配置文件里，可以设置：

<localRepository>某路径</localRepository>

（2）远程仓库

如果本地仓库没有某个依赖，那么maven就会从远程仓库去下载，默认就是从中央仓库去下载

（3）中央仓库

maven有一个自带的超级pom.xml文件，里面配置了一个默认的中央仓库：

<repositories>

<repository>

<id>central</id>

<name>Maven Repository Switchboard</name>

<url>http://repo1.maven.org/maven2</url>

<layout>default</layout>

<snapshots>

<enabled>false</enabled>

</snapshots>

</repository>

</repositories>

所以如果你不做任何配置，那么当你声明了某个依赖之后，就是如果本地仓库没有，就会去maven自带配置的这个中央仓库去拉取，就是http://repo1.maven.org/maven2这个地址。

（4）私服

一般正经公司都会自己假设一个maven私服，也就是在公司局域网内部，假设一个服务器，上面放一个maven私有仓库。

因为很多公司的服务器是不允许访问外网的，只能访问局域网，为了更好的安全性起见。而且依赖下载的速度也很快。

此时的话，核心的思路，就是在本地配置远程仓库为私服，如果本地仓库没有，先去私服找，如果私服没有，再去中央仓库找。在中央仓库找到后，先缓存在私服中，然后再缓存在本地仓库中。

此外，假设自己的私服，另外一个意义，就在于说，自己公司内部的一些发布包，可以放到私服上，供公司内的项目组之间使用。

（5）其他远程仓库

有些依赖可能在中央仓库没有，或者中央仓库的速度太慢，此时可能会用其他的一些远程仓库，比如jboss的仓库。java.net，google，codehaus，jboss，还有一些其他公司自己搞的Maven仓库，有少数的依赖包可能在中央仓库里找不到，只在其他仓库里。

那么私服除了连接中央仓库，还可以连接其他远程仓库。

（6）镜像仓库

比如说，像中央仓库在国外，很慢的，直接从中央仓库下载的话，是很慢的

所以一般国内的一些大型的互联网公司，阿里云，会搞一个镜像仓库，完全跟中央仓库一模一样的，代理了中央仓库所有的请求

你可以直接从阿里云镜像仓库去请求，如果有就直接返回了，国内网络的速度很快的，上百倍；阿里云如果自己没有，就会去从国外的中央仓库去下载

你就知道了什么啊，pom.xml里面声明了各种依赖之后，是怎么去找的，仓库，仓库的布局，多层仓库（本地、私服、镜像、中央、其他）。

下面我们就要用几讲的时间，把整个maven的仓库架构搭建起来，让依赖的下载走我们图里画的那一套复杂的仓库架构