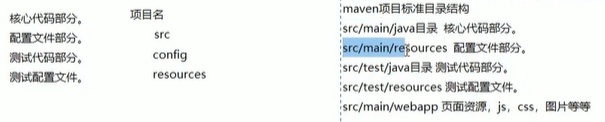
目的：解决jar包复杂和依赖问题

Maven项目结构目录

mvn clean 命令：删除本地编译的所有信息

Mvn compile 编译，会生成target文件夹 main

Mvn test 测试文件夹

Mvn package 打包 main和test 并且打包

生命周期：

-编译 compile

-测试 test

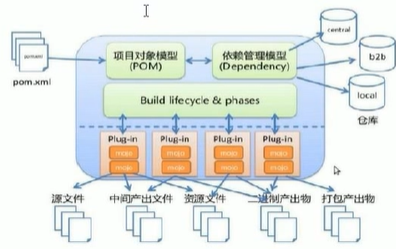
-打包 package

-安装 install

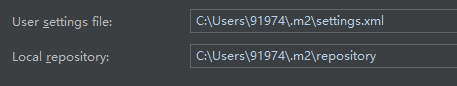
-发布 deploy

对于一个已开发项目，可以用mvn clean 清楚原有的编译信息。

Mvn模型:pom包含项目自身信息，项目依赖jar包信息，项目运行环境信息，jdk，tomcat等



Idea中mvn原始路径：D:/idea/IntelliJ IDEA 2020.2/plugins/maven/lib/maven3



-DarchetypeCatalog=internal：之前下载过的插件新建工程时在本地寻找，不用再互联网继续下载

Mvn settings文件中配置仓库位置

<localRepository>D:\maven\mavenrepository</localRepository>

Mvn的三个坐标：groupid , artifactid，version

jar包的添加

搜索maven中央仓库 搜素需要的关键词

然后复制dependency 的依赖代码，在idea中会提示maven更新，点击更新就会自动下载jar包

<scope>test</scope>

设置作用域。

作用域范围情况



Mvn创建可以选择骨架创建和非骨架创建，非骨架的文件结构更方便，因此大多选择非骨架的

Build标签 配置各种不同情况下的环境和配置

* build标签描述了如何编译及打包项目，具体的编译和打包工作是通过其中的plugin配置来实现的。当然，plugin不是必须的，即使不添加默认也会引入以下插件：