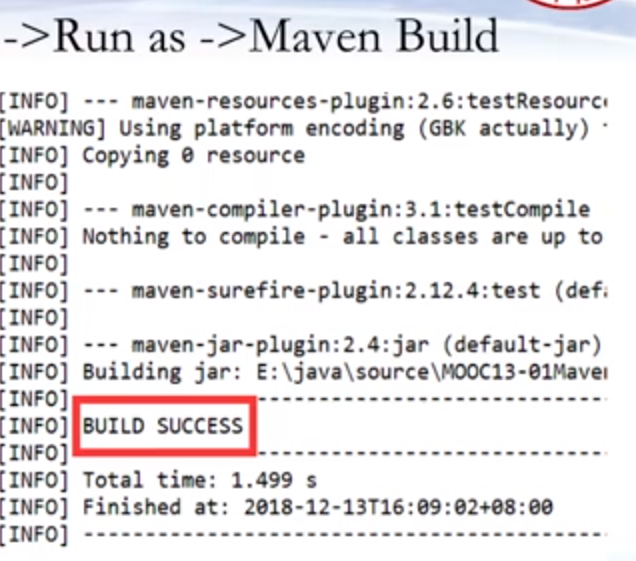
Mvn中央仓库（mvnrepository.com）

阿里云仓库

谷歌仓库 三大远程仓库

可以拿到不同版本的依赖，复制到maven，pom文件

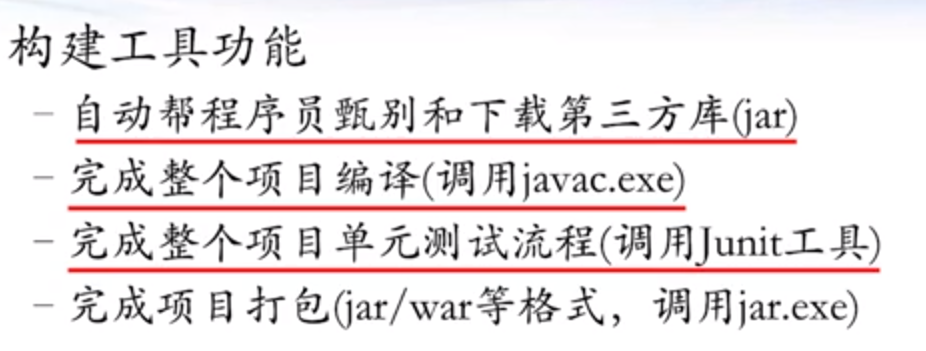
依赖复制到pom文件里，右键项目选run as maven build线进行编译，然后maven自动去帮我们把jar包下载回来（留意jar包存到哪里）



Build success然后再回到程序中run as Java Application

之后如果需要在其他电脑上使用，直接复制pom文件，然后run as maven build

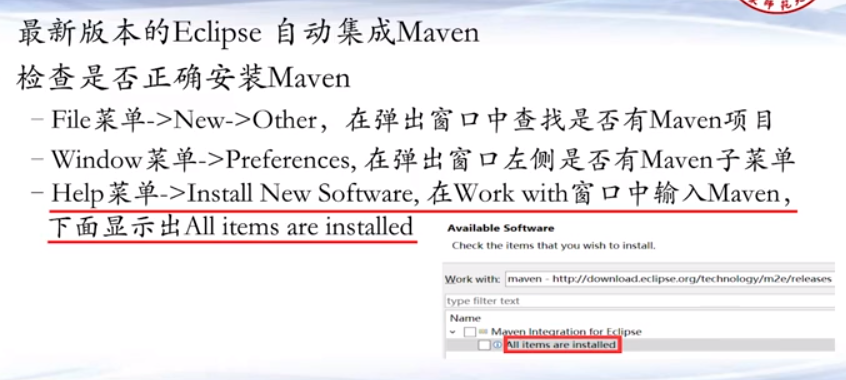
Maven（java构建工具）好处



.pom文件下

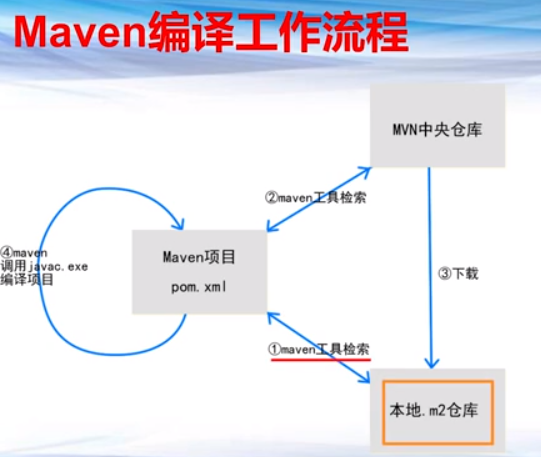
<packaging>jar</packaging>

希望打包格式为jar





**.m2本地仓库**



POM文件



**Maven项目基本目录结构**



图片什么的也放resources文件夹下

1. 新建一个maven项目（有两种方式，一种是直接，一种是先建java project，再convert maven project）
2. 写pom文件（依赖）
3. 新建sqlMapConfig.xml文件（配置jdbc）
4. 写pojo 类，也就是持久化类并实现Serializable接口（持久化类的属性要与数据库的字段相对应）
5. 构建映射器（由一个接口6和一个xml文件7构成）
6. 在dao层（数据访问层）下写UserMapper接口， 实现findAll方法
7. 在Resource文件夹下，新建UserMapper.xml配置文件（写sql语句）
8. 接下来创建测试类，在test文件夹下新建MybatisTest类

InputStream in= Resources.getResourceAsStream("SqlMapConfig.xml");

//从SqlMapConfig.xml获取数据库连接信息，从数据库读取信息

SqlSessionFactoryBuilder builder = new SqlSessionFactoryBuilder();

//声明一个工厂构造者

SqlSessionFactory factory = builder.build(in);

//建工厂

SqlSession session = factory.openSession();

UserMapper userMapper = session.getMapper(UserMapper.class);

List<User> users = userMapper.findAll();

for (User user:users){

System.out.println(user);

}

session.close();

n.close();

依赖不要用最新版本的，往往不太稳定

**Enable Auto-Import：以后更改 .pom 文件后自动下载依赖包**

groupId   
  
定义了项目属于哪个组，举个例子，如果你的公司是mycom，有一个项目为myapp，那么groupId就应该是com.mycom.myapp.   
  
artifacted   
  
定义了当前maven项目在组中唯一的ID,比如，myapp-util,myapp-domain,myapp-web等。