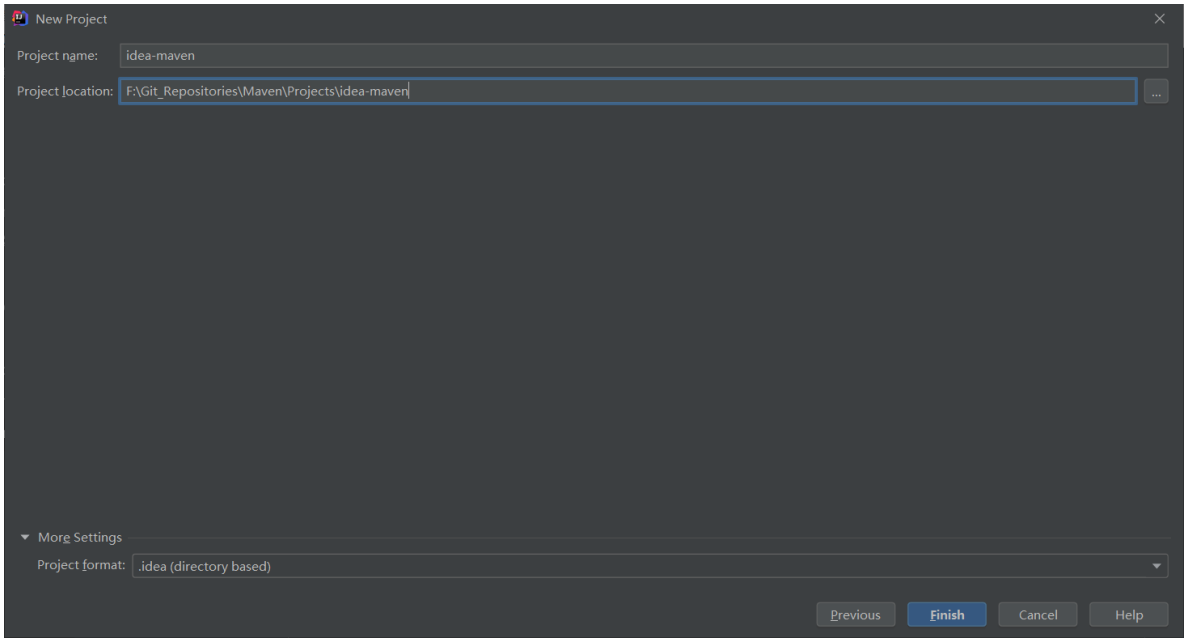
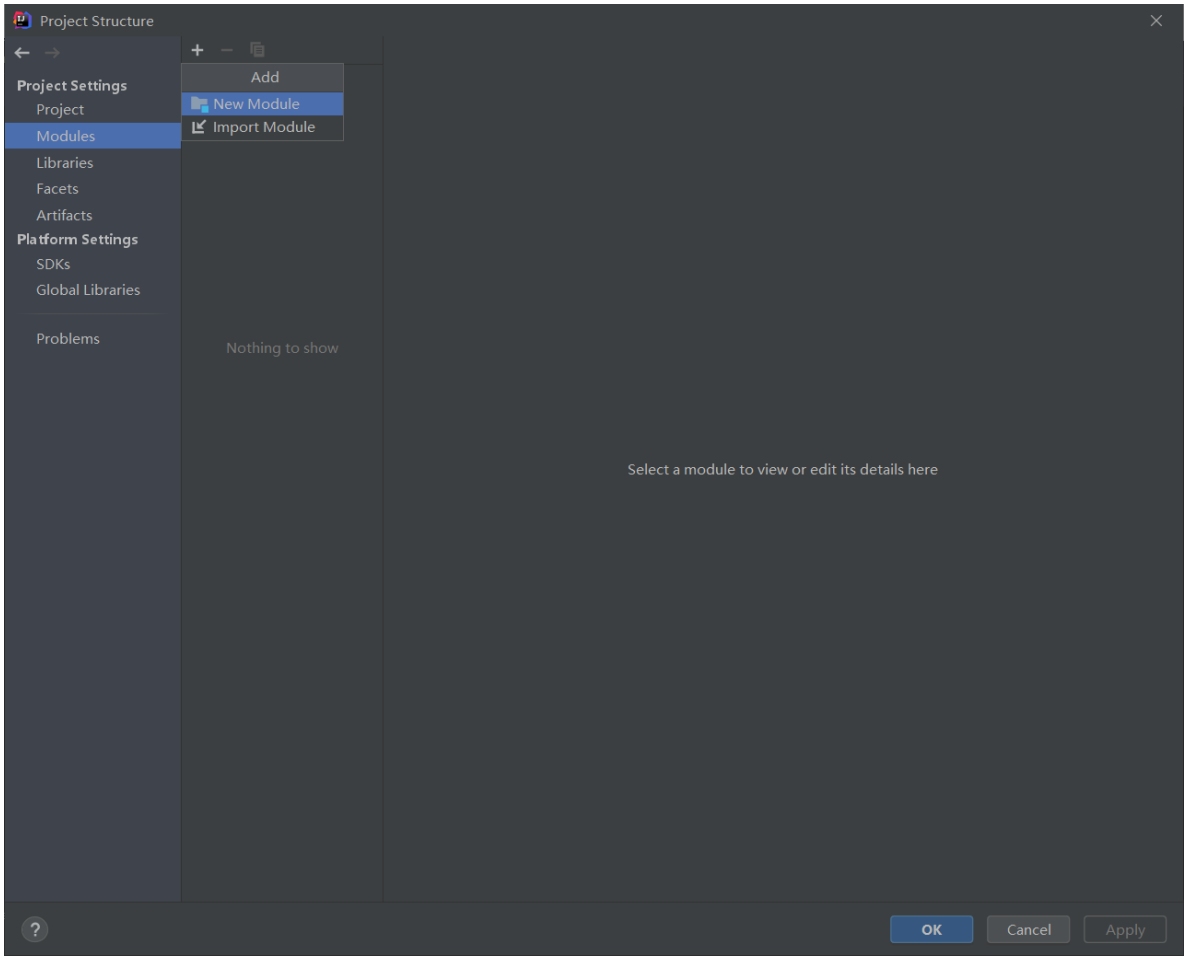
使用idea创建一个普通的maven的java工程

1、创建一个空工程，名为idea-maven

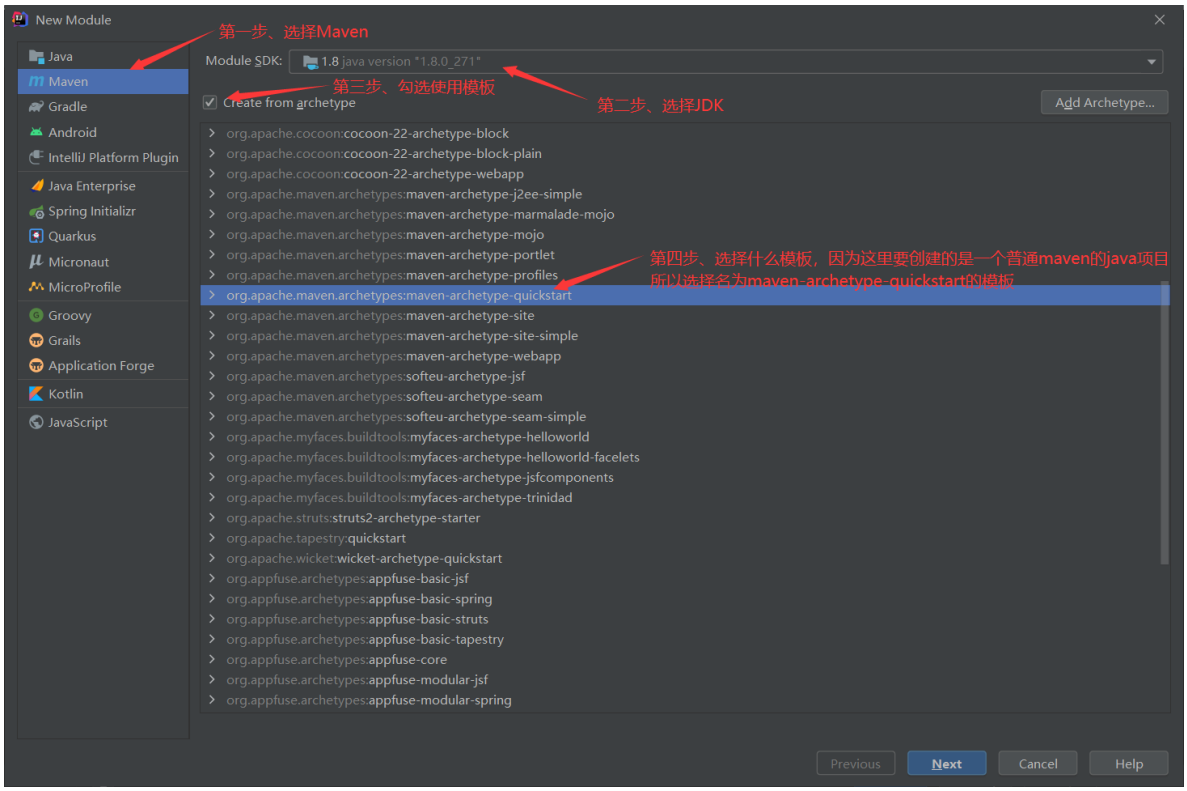


2、添加一个模块Module



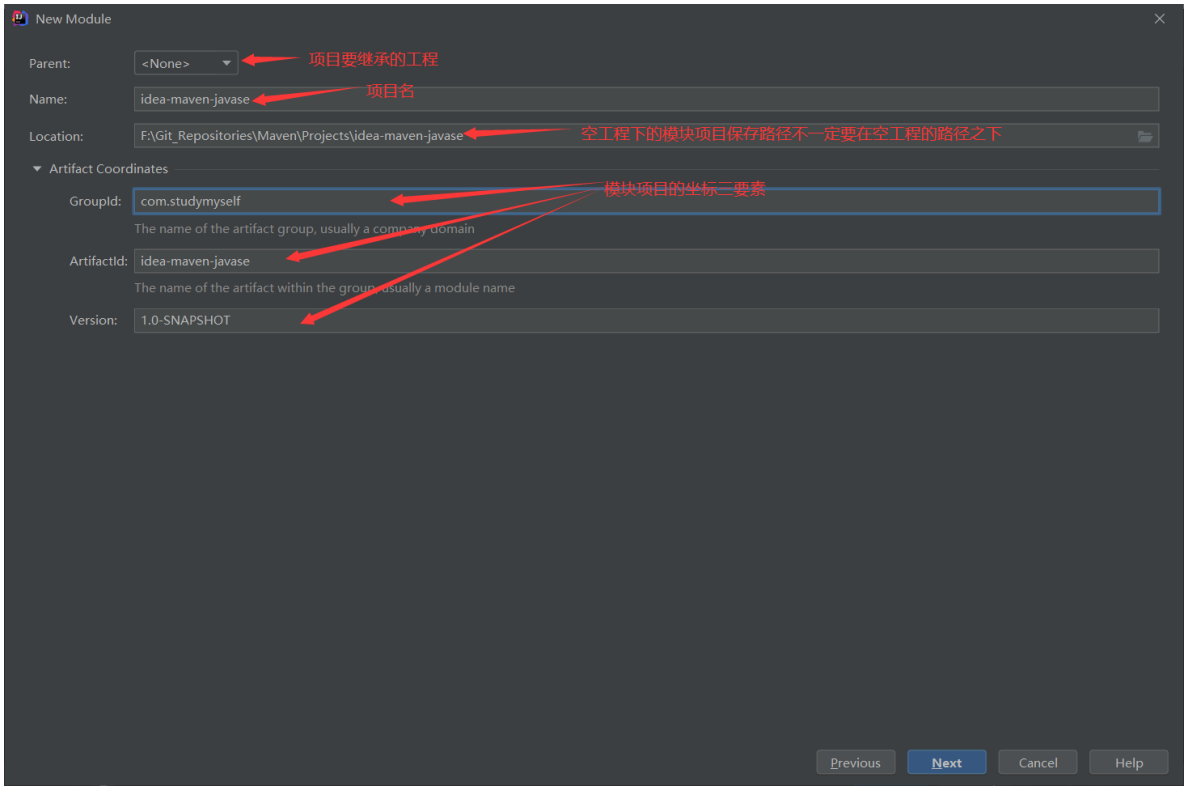
3、选择Maven，选择完毕后下一步

选择Maven选项，其中我们勾选使用模板，所陈列的都是不同种类的模板，不同模板生成的项目都有其对应的目录结构和常用文件，我们要做什么项目就选择什么模板，生成的项目目录文件就非常的适合该项目开发



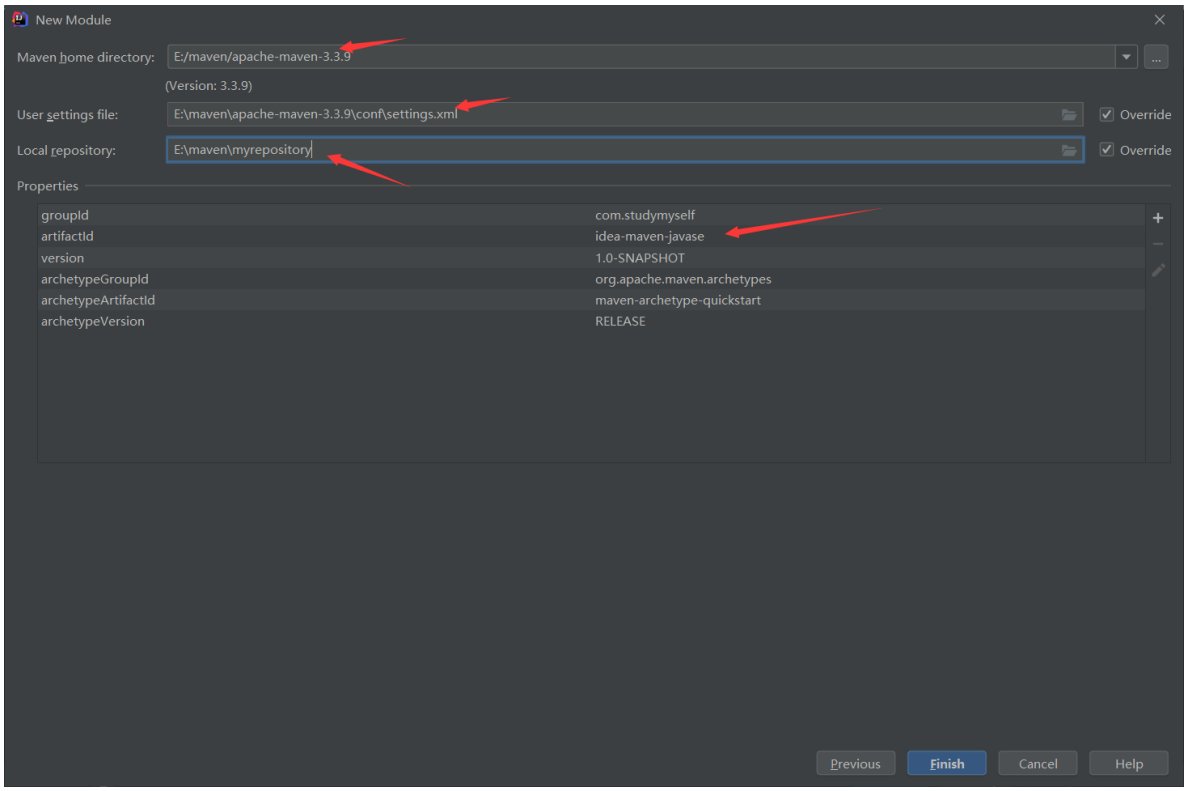
4、编写项目信息，下一步

Parent：不继承其他项目就不填
Name：填写项目名字
Location：一般不做改动
Artifact Coordinattes：项目的坐标三要素，按要求填写

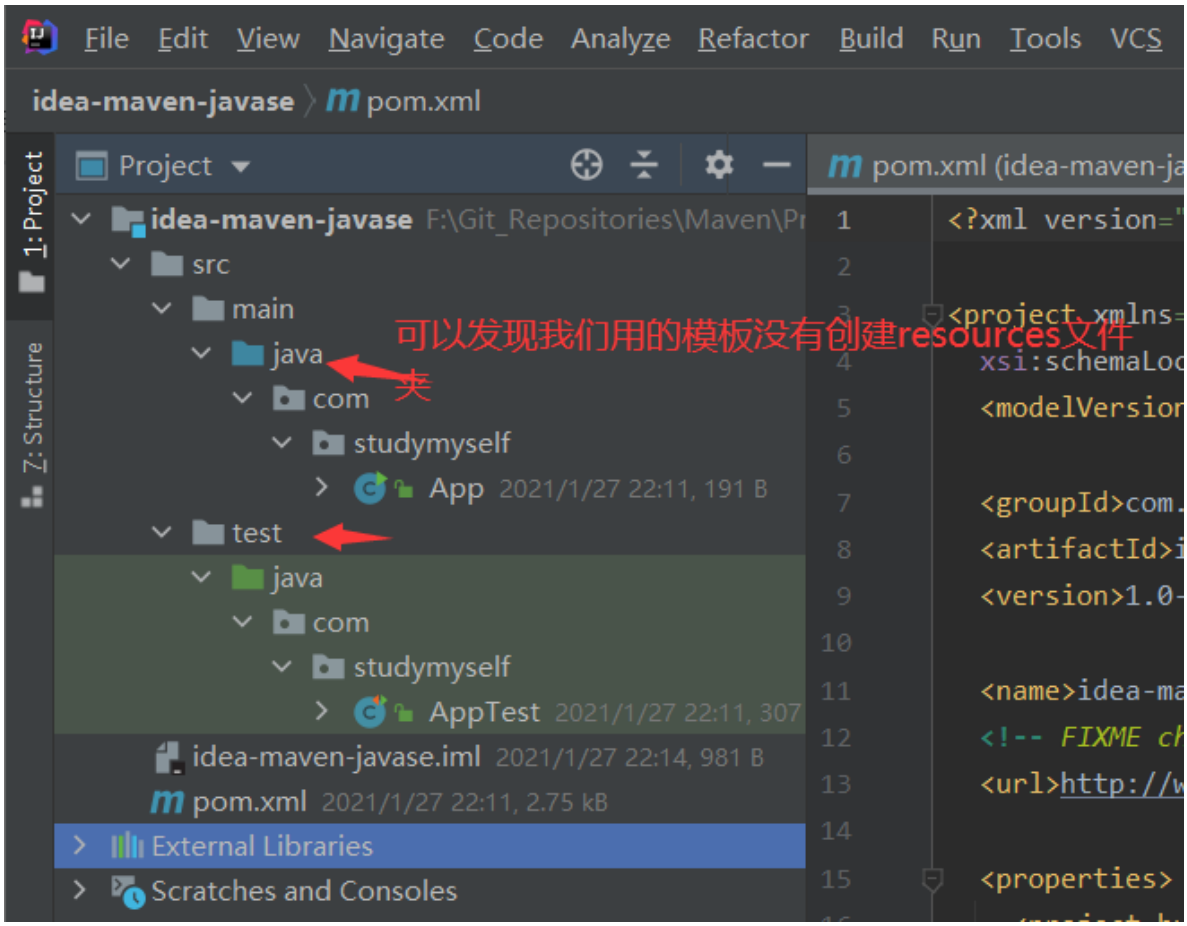


5、选择maven的配置信息

发现默认的还是idea内置的，前面在Settings中配置的没起作用，可能是创建的是空工程的原因



6、创建成功浏览项目结构，添加模板不足的缺少文件



发现所选择的模板并没有创建resources目录，那么我们自己在main和test目录下分别新建一个resources

--注意：

maven项目中的根据文件目录中存储文件的不同，分为不同种目录：

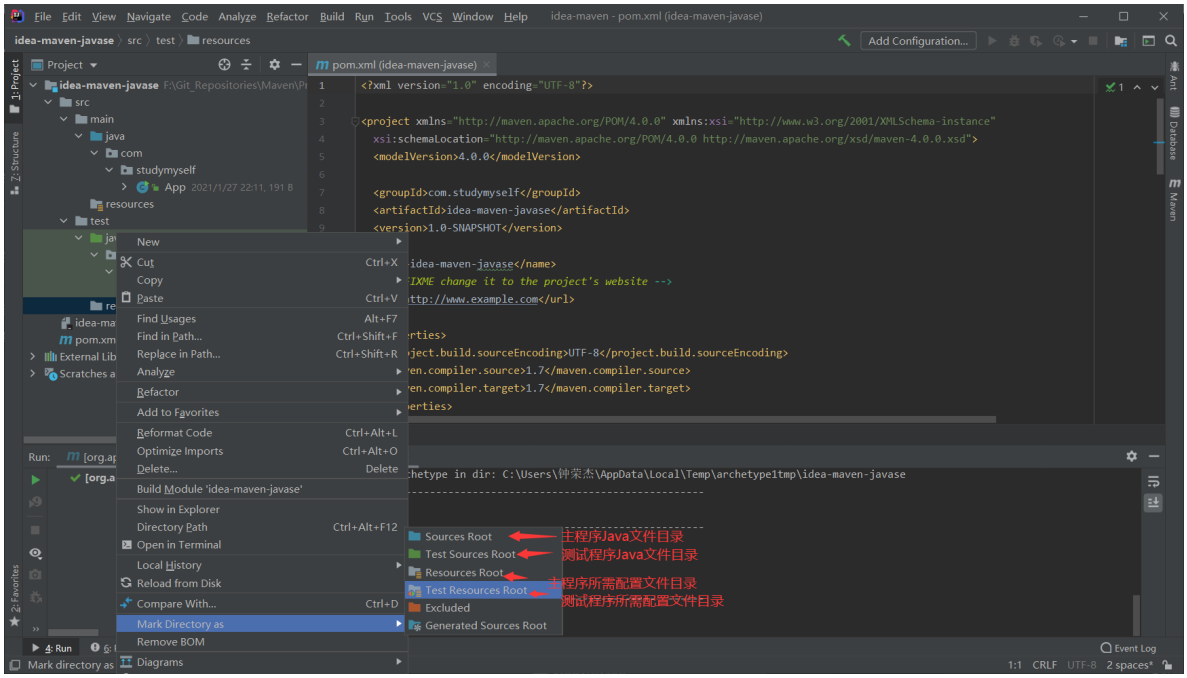
Sources Root：代表存放的是主程序的java文件

Test Sources Root：表示存放的是测试程序的java文件

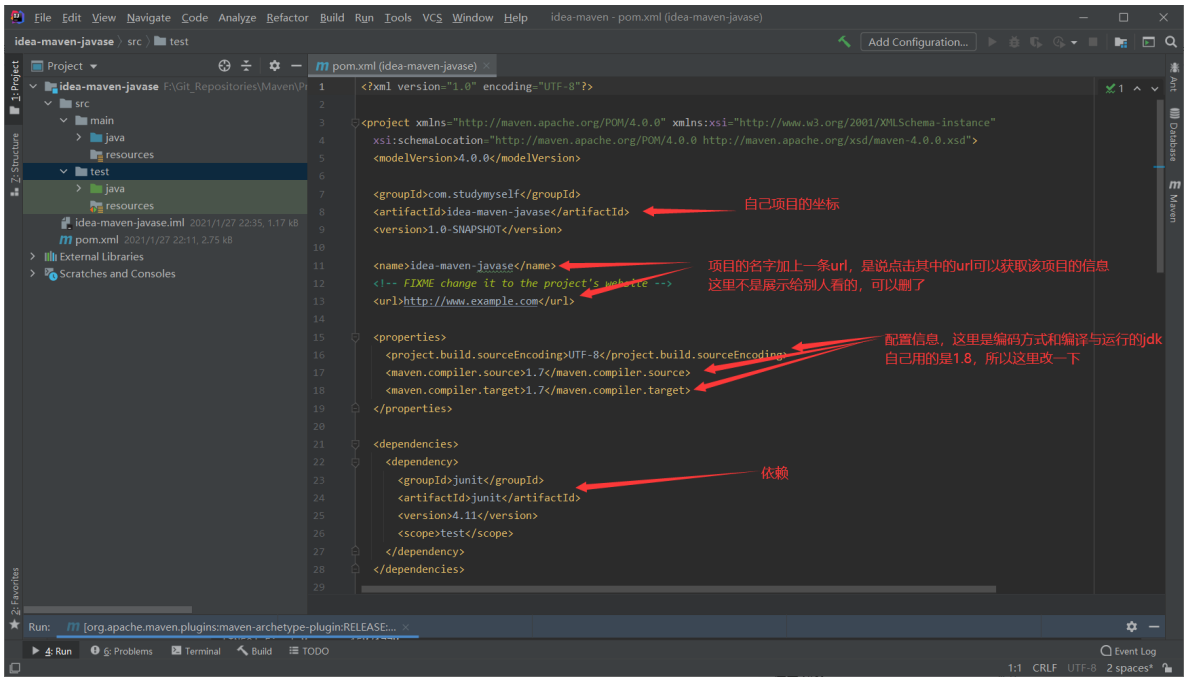
Resources Root：代表存放的是主程序的java文件所需的配置资源文件

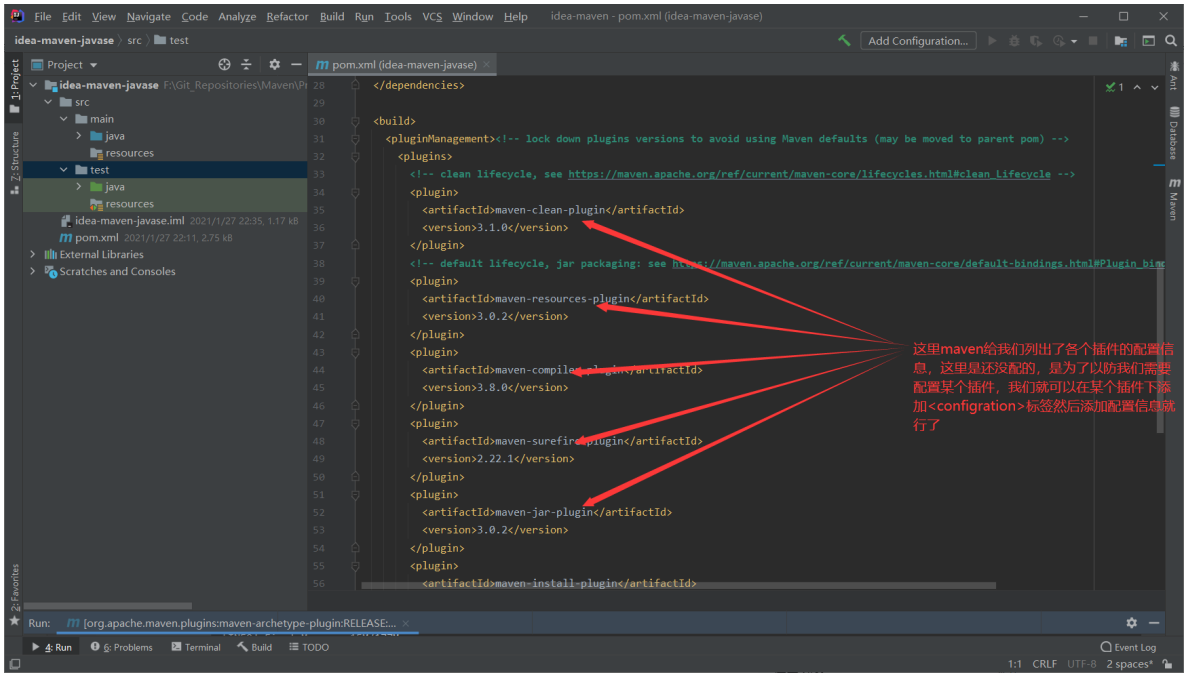
Test Resources Root：代表存放的是测试程序的java文件所需的配置资源文件

修改文件目录类型：右键文件目录->Mark Directory as->选择相应类型



7、查看pom文件





8、进行单元测试

删掉主程序和测试程序包下的两个原始java类，

src\main\com\studymyself下新建 HelloMaven.java

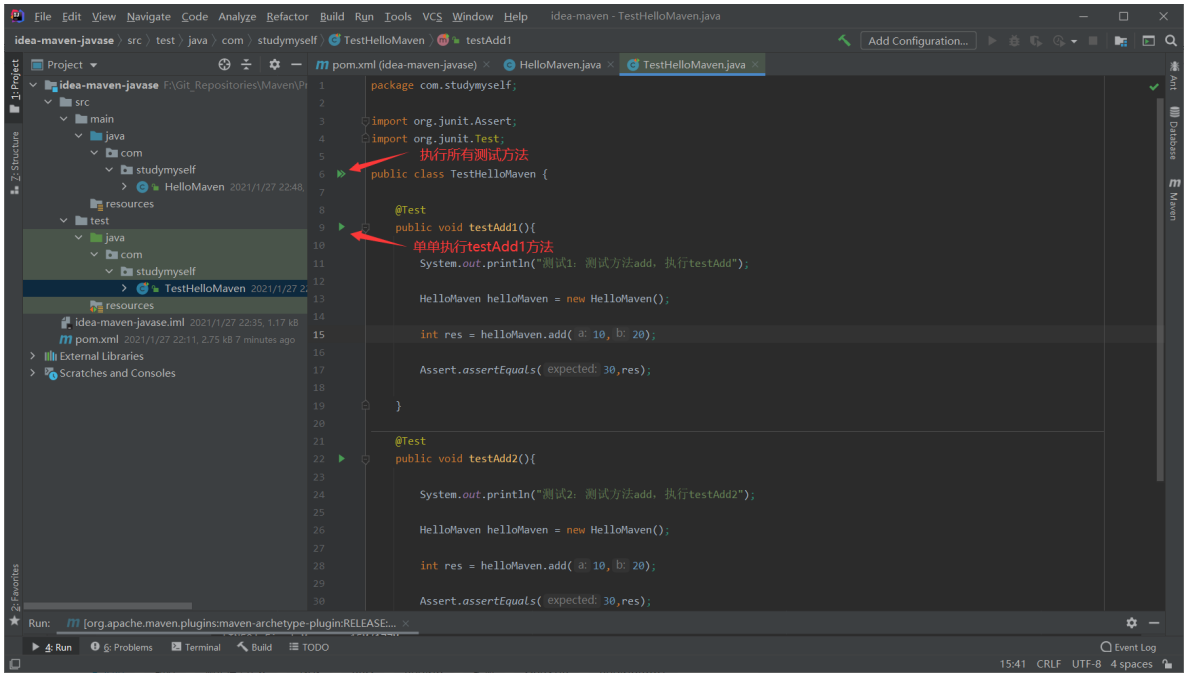
```
package com.studymyself;

public class HelloMaven {

    public int add(int a,int b){

        return a+b;

    }

}
```

src\test\com\studymyself下新建 TestHelloMaven.java

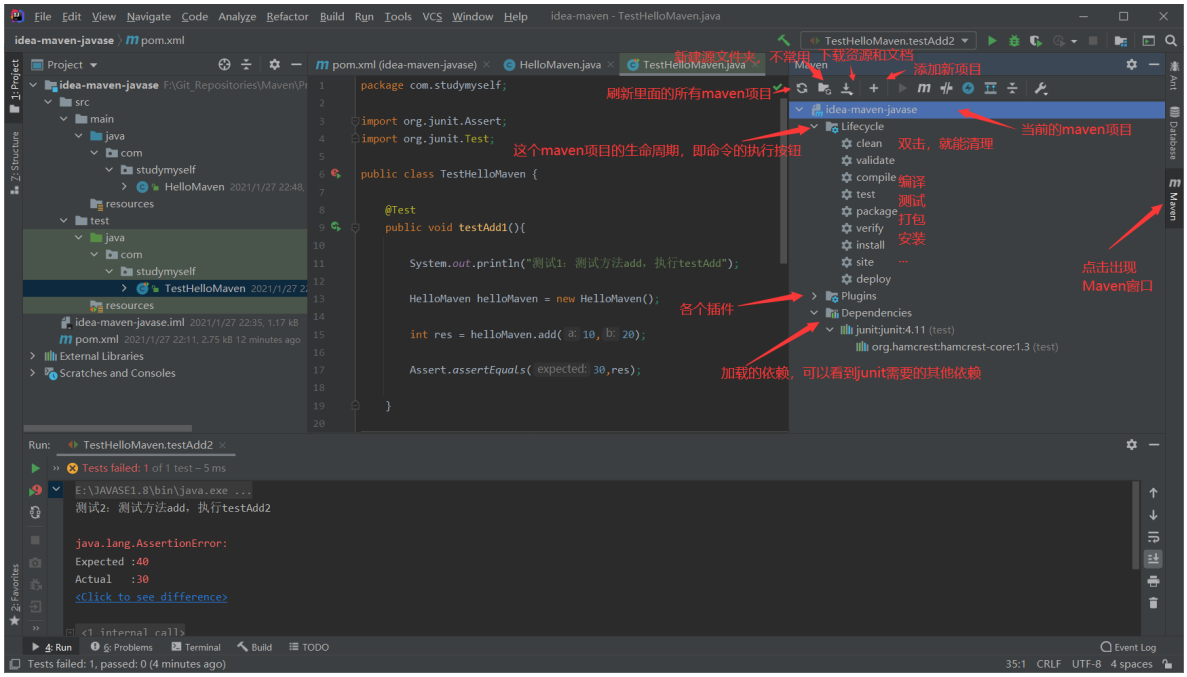
```
package com.studymyself;

import org.junit.Assert;
import org.junit.Test;

public class TestHelloMaven {

    @Test
    public void testAdd1(){

        System.out.println("测试1：测试方法add，执行testAdd");

        HelloMaven helloMaven = new HelloMaven();

        int res = helloMaven.add(10,20);
```

```
        Assert.assertEquals(30,res);

    }

    @Test
    public void testAdd2(){

        System.out.println("测试2：测试方法add，执行testAdd2");

        HelloMaven helloMaven = new HelloMaven();

        int res = helloMaven.add(10,20);

        Assert.assertEquals(30,res);

    }

}
```

点击方法前面的运行三角执行该测试方法



9、点击右边的Maven选项，弹出Maven窗口



在maven窗口中我们可以执行各种操作，我们点击其中的编译或者打包时，其弹出的控制台是使用的maven自身的而非idea的，在idea中我们设置的都是UTF-8编码，所以文件都是UTF-8，而我们知道大陆下载的maven构建项目默认是以GBK的形式，所以双击test的时候控制台输出的测试语句中文出现乱码。

--还有就是目录结构上方的设置齿轮按钮中Show Exclude Files需要打上勾，才会显示生成的target目录