链滴

SpringBoot(一) 、基于 Maven 构建多模块项目

作者：TOJing

原文链接：https://ld246.com/article/1604633919222

来源网站：链滴

SpringBoot(一）、基于Maven构建多模块项目

分布式项目搭建规范

规范地搭建微服务项目将有助于应用的开发、维护以及代码的理解。目前应用比较广泛的是基于Mave构建多模块的方式，这种方式搭建的每个模块各司其职，负责应用中不同的功能，同时每个模块采用级依赖的方式，最终构建成一个聚合型的Maven项目。

本项目将结构划分成了4层，结构如下图。当然，随着实际情况，后续也可按照所需自由划分。

技术选用

- jdk1.8

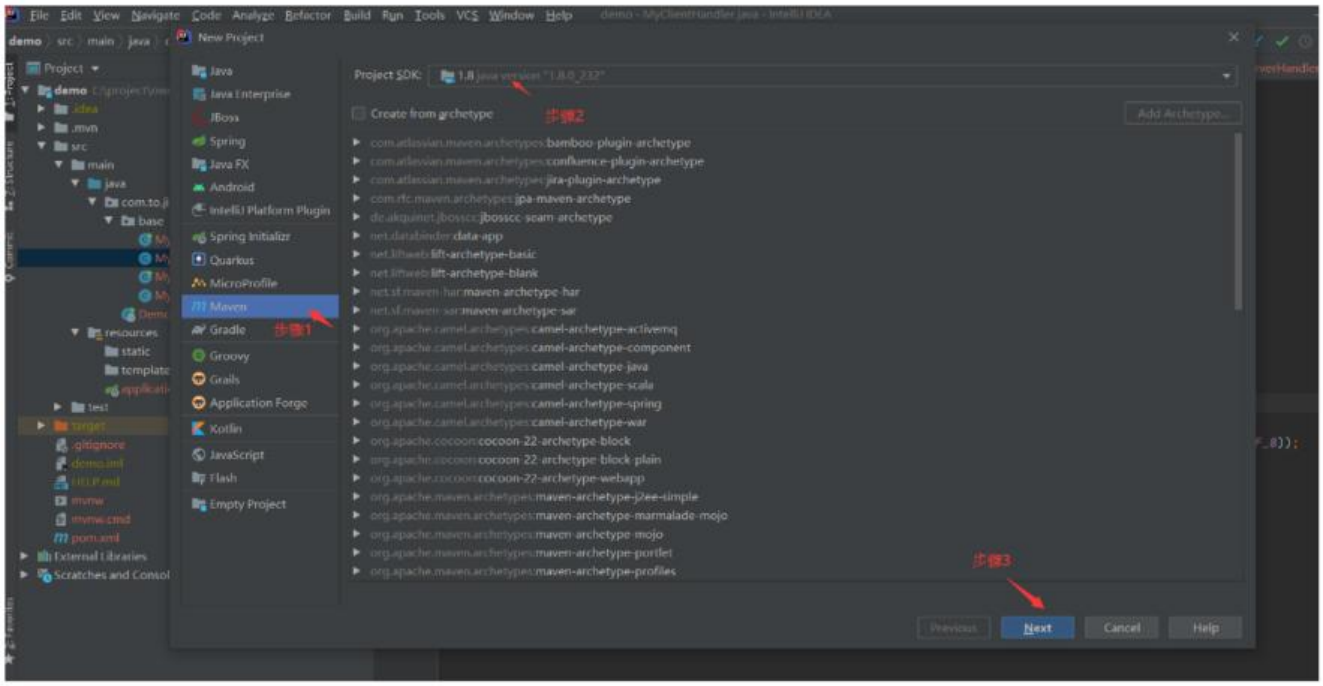
- Idea2020
- Maven
- Springboot
- fastjson
- lombok

相关工具的安装和使用这里不再赘述。

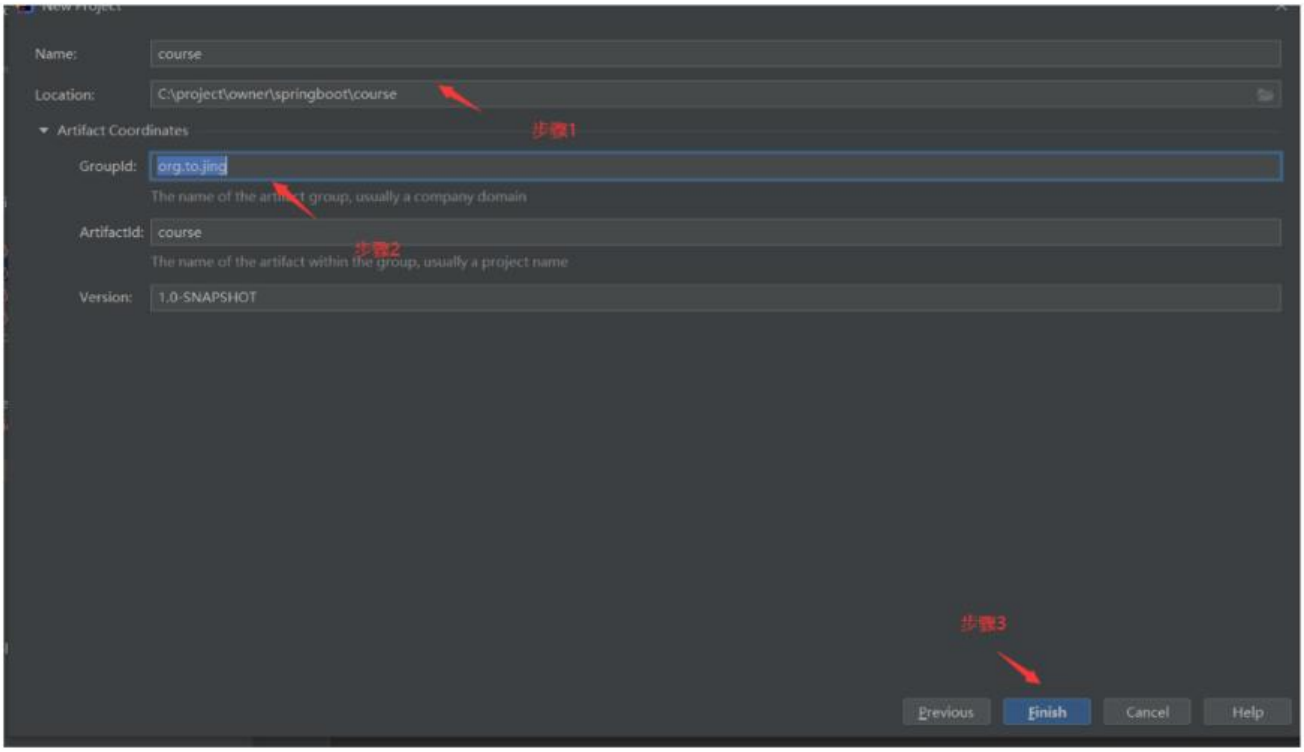
使用idea构建Maven父项目

首先，我们先通过Idea构建父项目course，步骤如下：

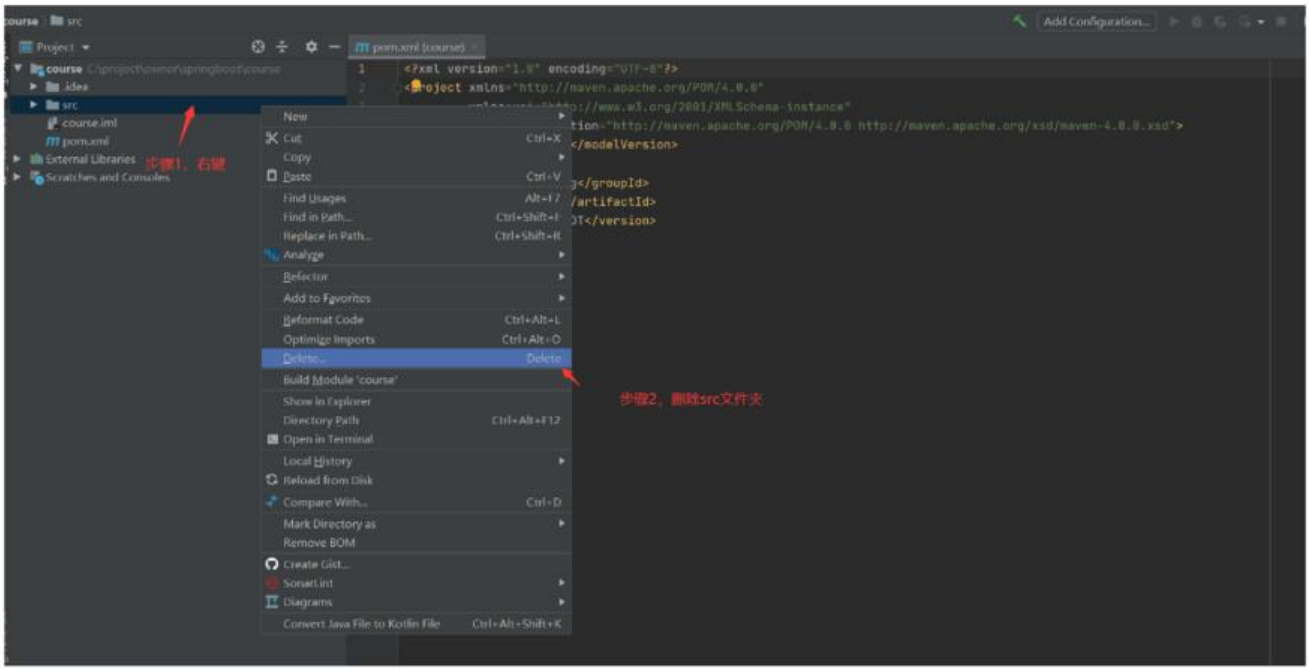
（1）打开开发工具Idea,然后选择菜单栏中“File”的“New”命令，进入“New Project”，即创新项目的界面



（2）单击“Next”按钮，进入Maven多模块项目的命名界面，在这里建议Maven项目的命名尽量简、规范，单击“Finish"完成

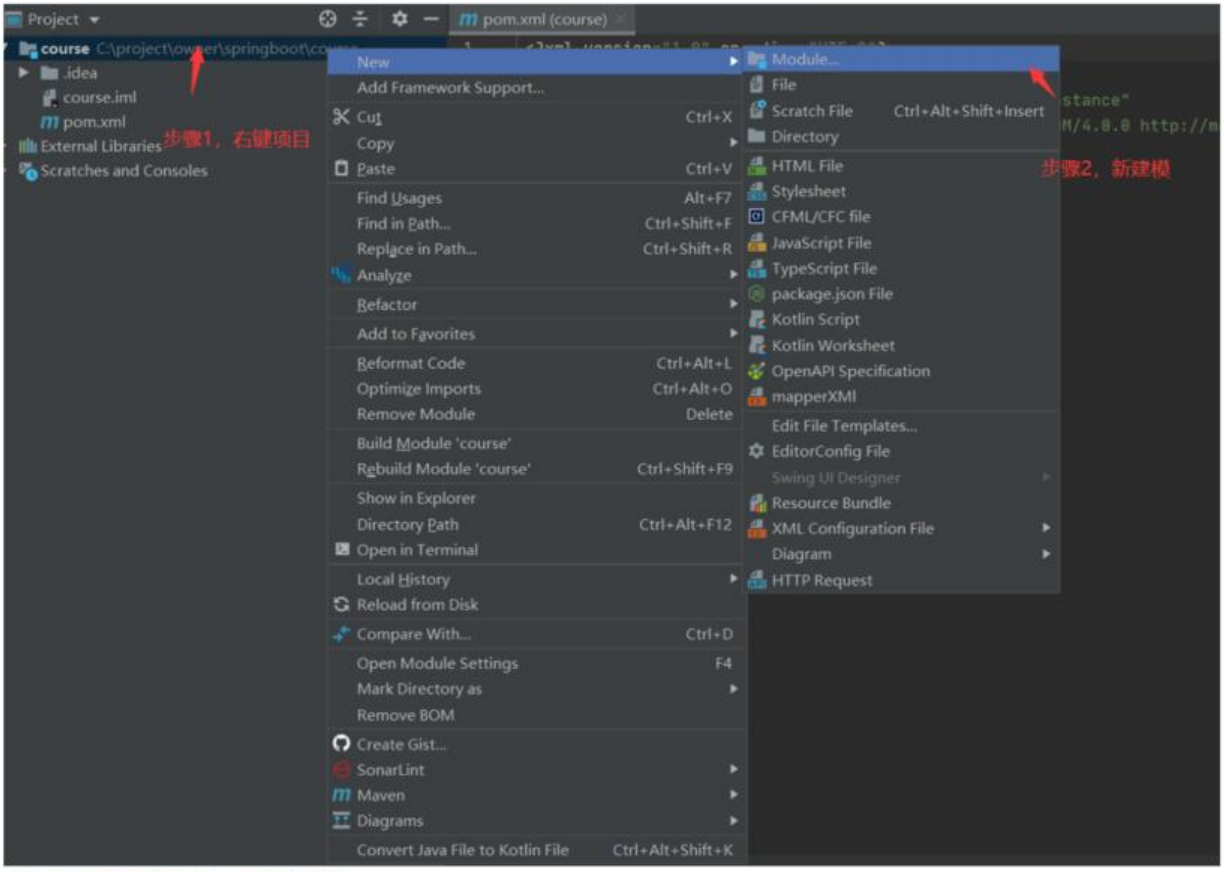


（3）选中父项目的src文件夹，右键点击删除。

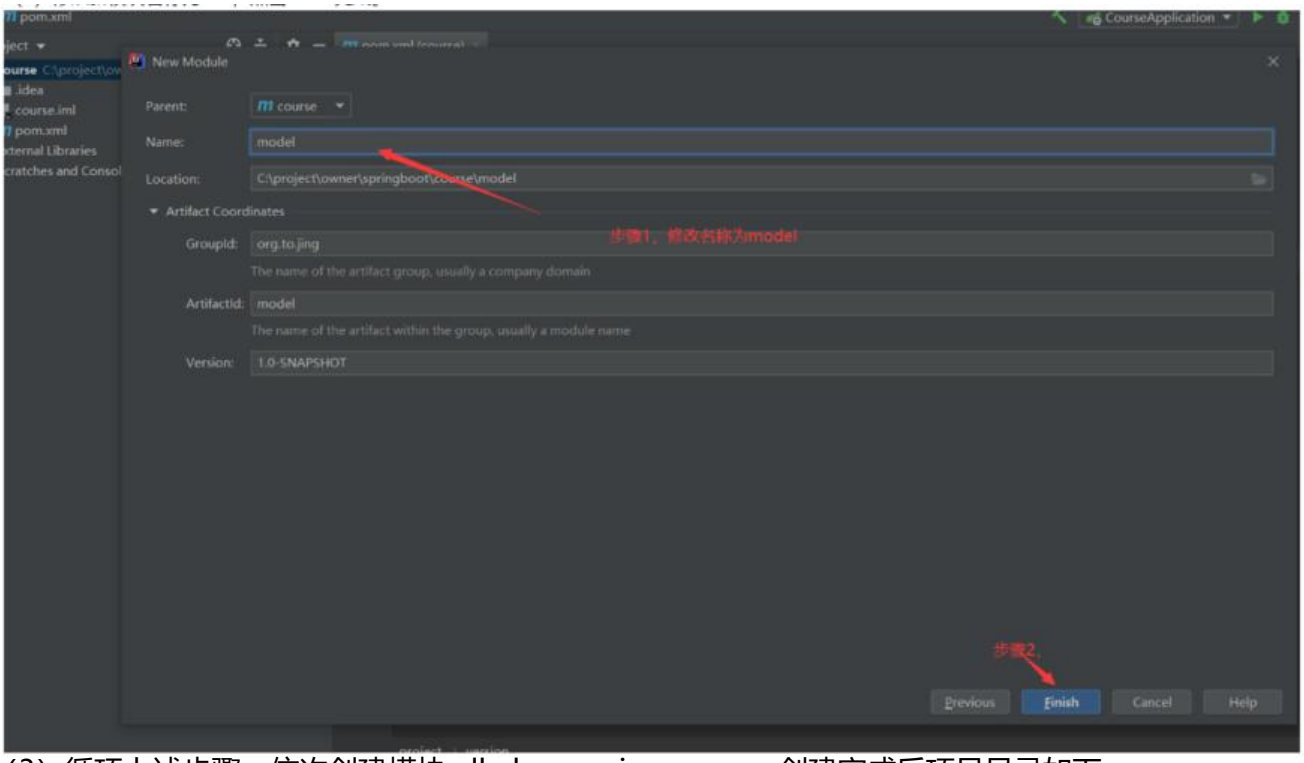


使用idea构建子模块

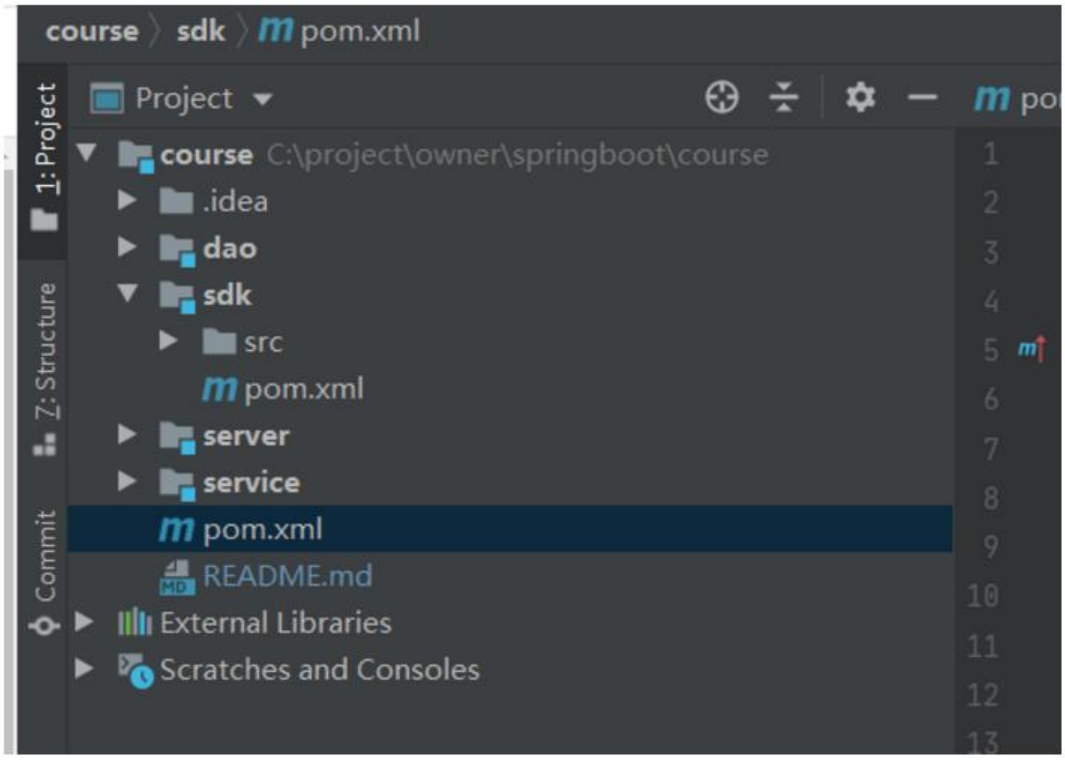
（1）右键父项目course,新建模块sdk。



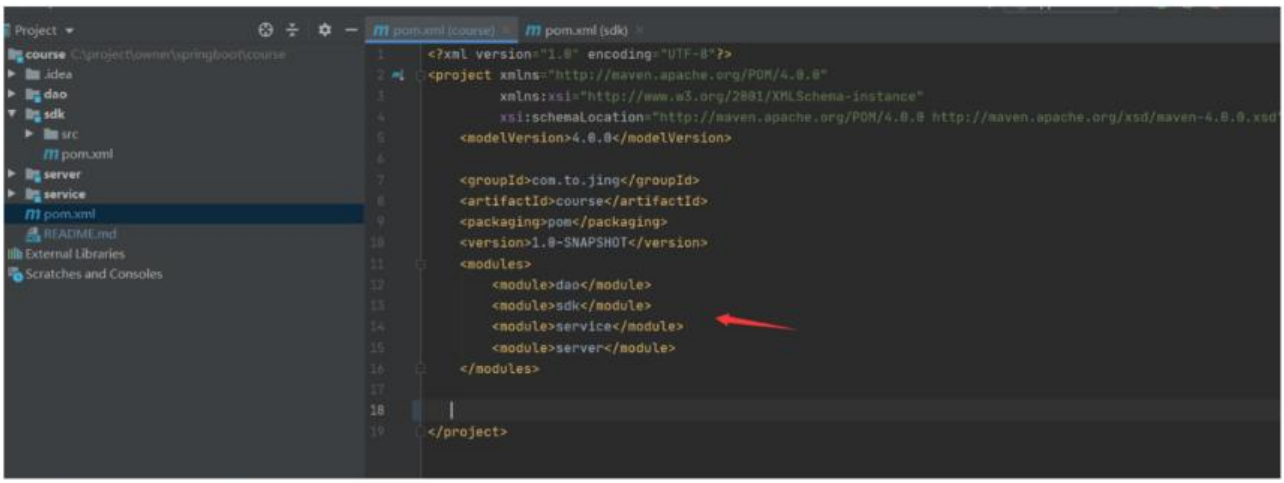
（2）修改新模块名称为sdk，点击finish完成。



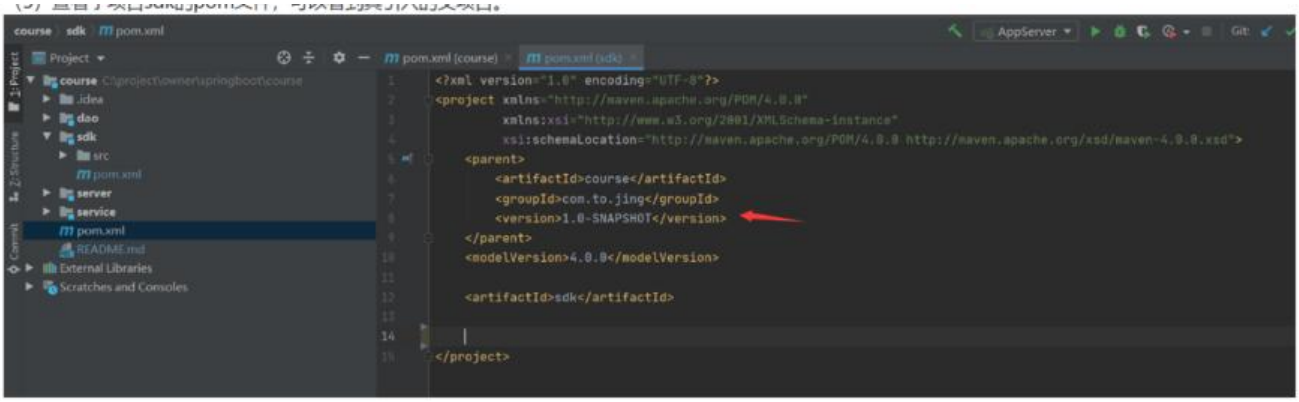
（3）循环上述步骤，依次创建模块sdk,dao,service,server。创建完成后项目目录如下：



（4）查看父项目的pom文件，可以看到其引入的四个模块。



（5）查看子项目sdk的pom文件，可以看到其引入的父项目。

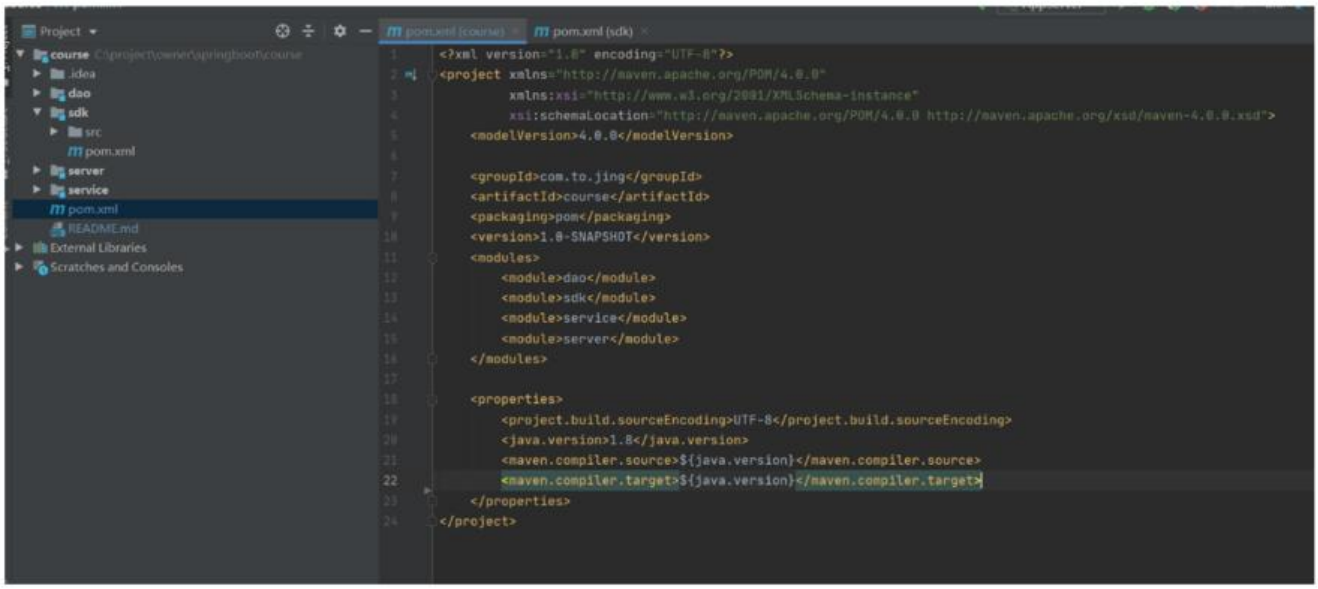


父项目设置统一资源和版本控制

在父项目course的pom文件下添加如下代码：

```
<!-- 定义项目整体资源的编码为UTF-8以及JDK的版本为1.8 -->
<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <java.version>1.8</java.version>
    <maven.compiler.source>${java.version}</maven.compiler.source>
    <maven.compiler.target>${java.version}</maven.compiler.target>
</properties>
```

添加后pom文件内容如下：



为各个模块引入相关依赖

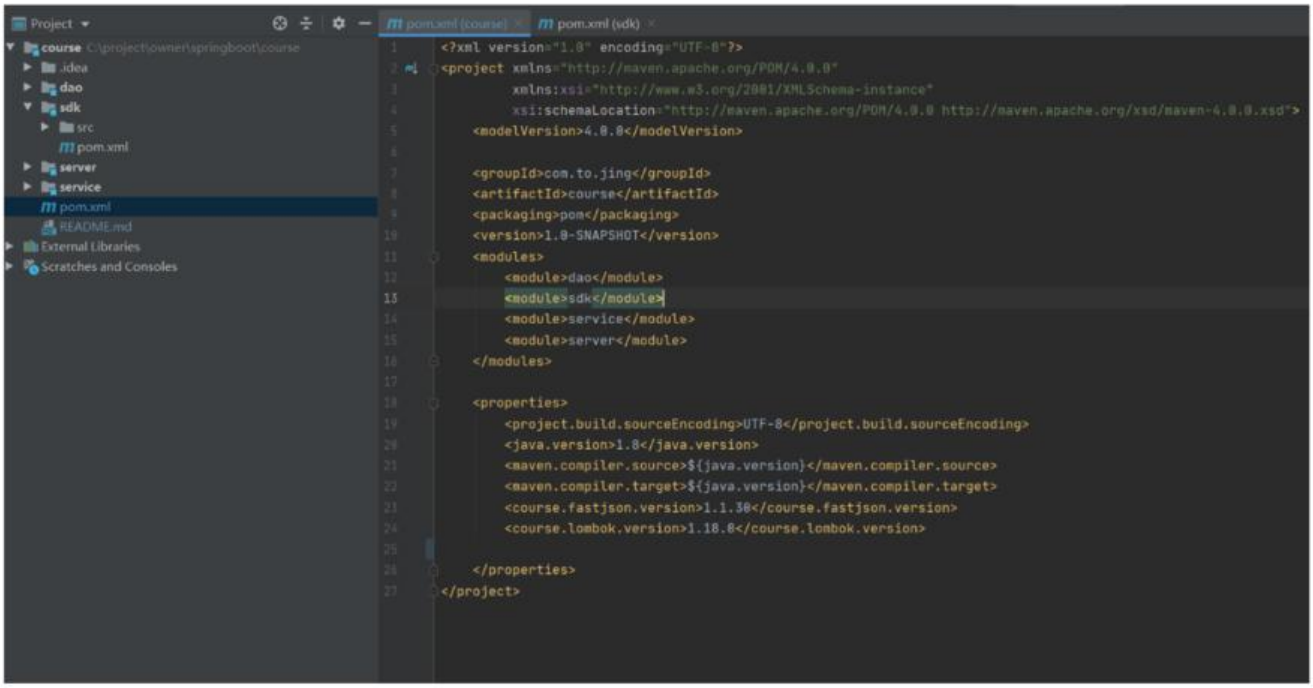
本项目将在各个子模块下引入fastjson和lombok，通过父项目来控制统一的版本管理。

（1）在父项目中pom引入如下代码定义版本。

```
<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <java.version>1.8</java.version>
    <maven.compiler.source>${java.version}</maven.compiler.source>
    <maven.compiler.target>${java.version}</maven.compiler.target>
    <course.fastjson.version>1.1.30</course.fastjson.version>
    <course.lombok.version>1.18.0</course.lombok.version>

</properties>
```

在propertises中增加了两行fastjson.version和lombok.version。完整pom文件如下图：



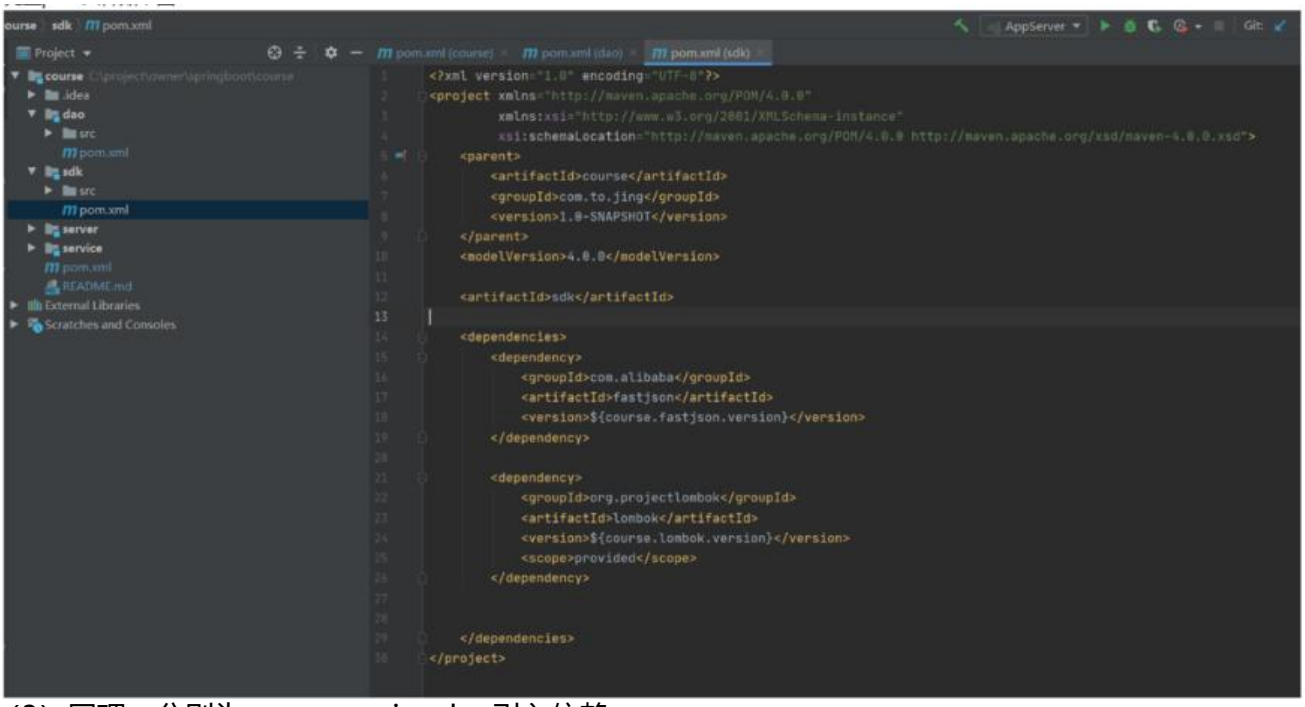
（2）在子项目sdk中引入依赖。添加如下代码：

```
<dependencies>
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>fastjson</artifactId>
        <version>${course.fastjson.version}</version>
    </dependency>

    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>${course.lombok.version}</version>
        <scope>provided</scope>
    </dependency>


  </dependencies>
```

完整pom文件如下图：



（3）同理，分别为server,service,dao引入依赖。

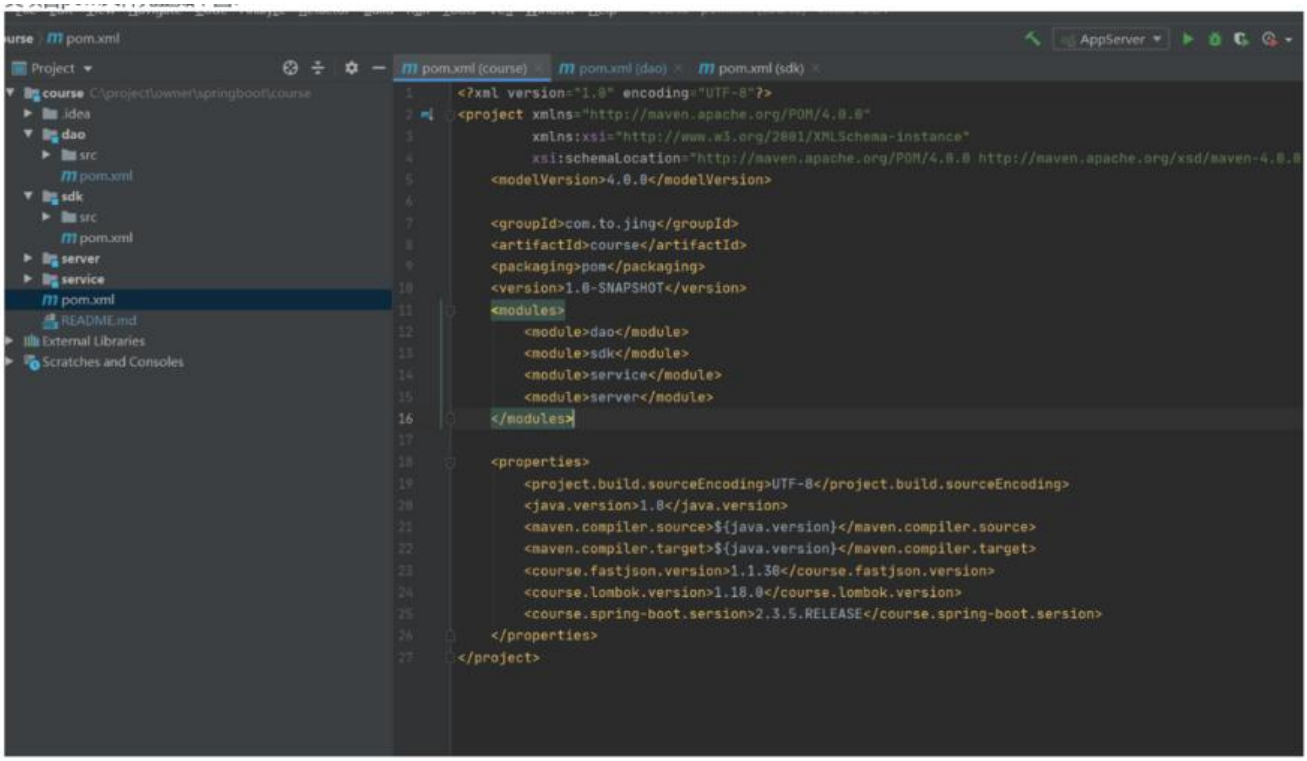
为server子模块引入springboot依赖

server模块其实是一个springboot项目，所以需要引入springboot依赖。

（1）在父项目course中添加springboot版本，在properties中添加如下代码：

```
<course.spring-boot.sersion>2.3.5.RELEASE</course.spring-boot.sersion>
```

父项目pom文件完整如下图：



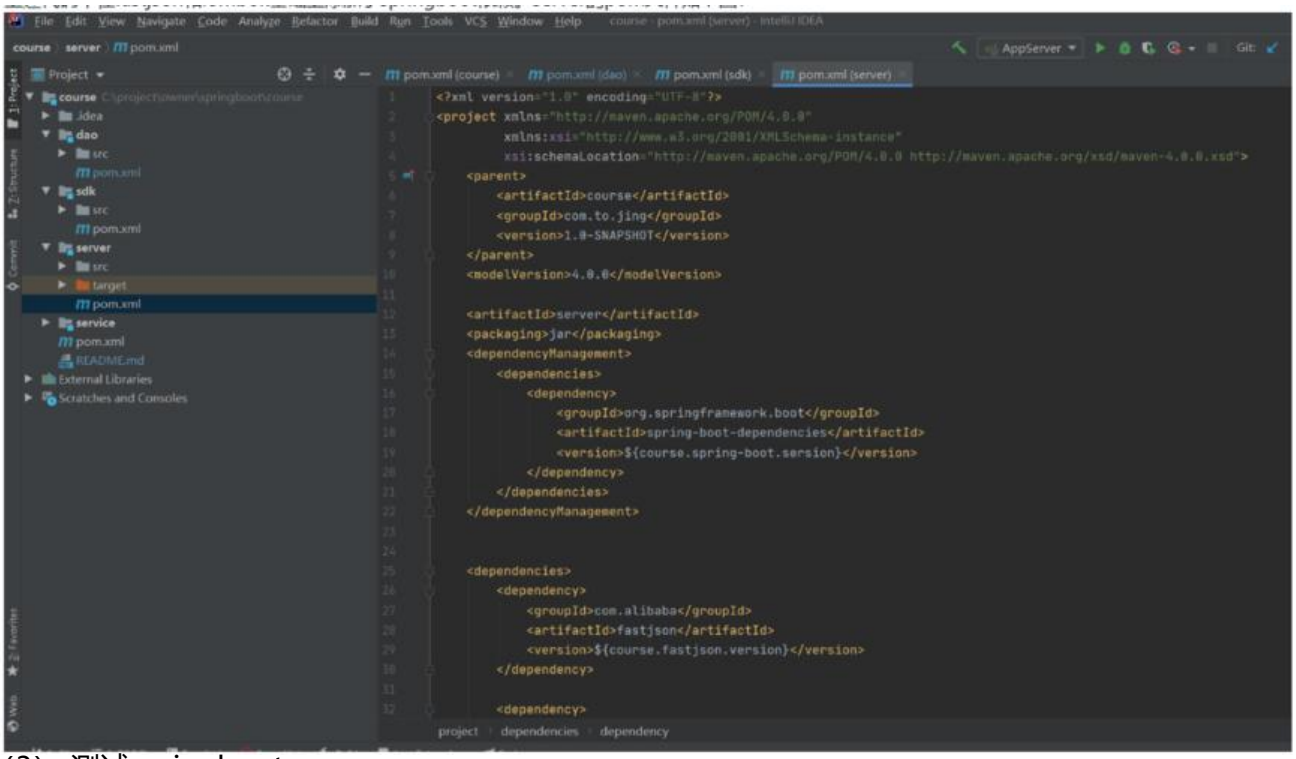
（2）在server模块的pom文件添加springboot依赖，代码如下：

```
<dependencyManagement>
      <dependencies>
        <dependency>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-dependencies</artifactId>
          <version>${course.spring-boot.sersion}</version>
        </dependency>
      </dependencies>
  </dependencyManagement>
 <dependencies>
      <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>fastjson</artifactId>
        <version>${course.fastjson.version}</version>
      </dependency>

      <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>${course.lombok.version}</version>
        <scope>provided</scope>
      </dependency>
      <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
        <version>${course.spring-boot.sersion}</version>
      </dependency>
      <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-test</artifactId>
        <version>${course.spring-boot.sersion}</version>
      </dependency>
  </dependencies>
```

关于dependencyManagement与 dependencies标签的异同读者有兴趣的可以自行了解。

上述代码中在fastjson和lombok基础上添加了springboot依赖。server的pom文件如下图：



（3） 测试springboot

在server模块新建包com.to.jing.course.server,包下新建Appserver.java。代码如下：

```
package com.to.jing.course.server;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class AppServer {
    public static void main(String[] args) {
        SpringApplication.run(AppServer.class);
    }
}
```

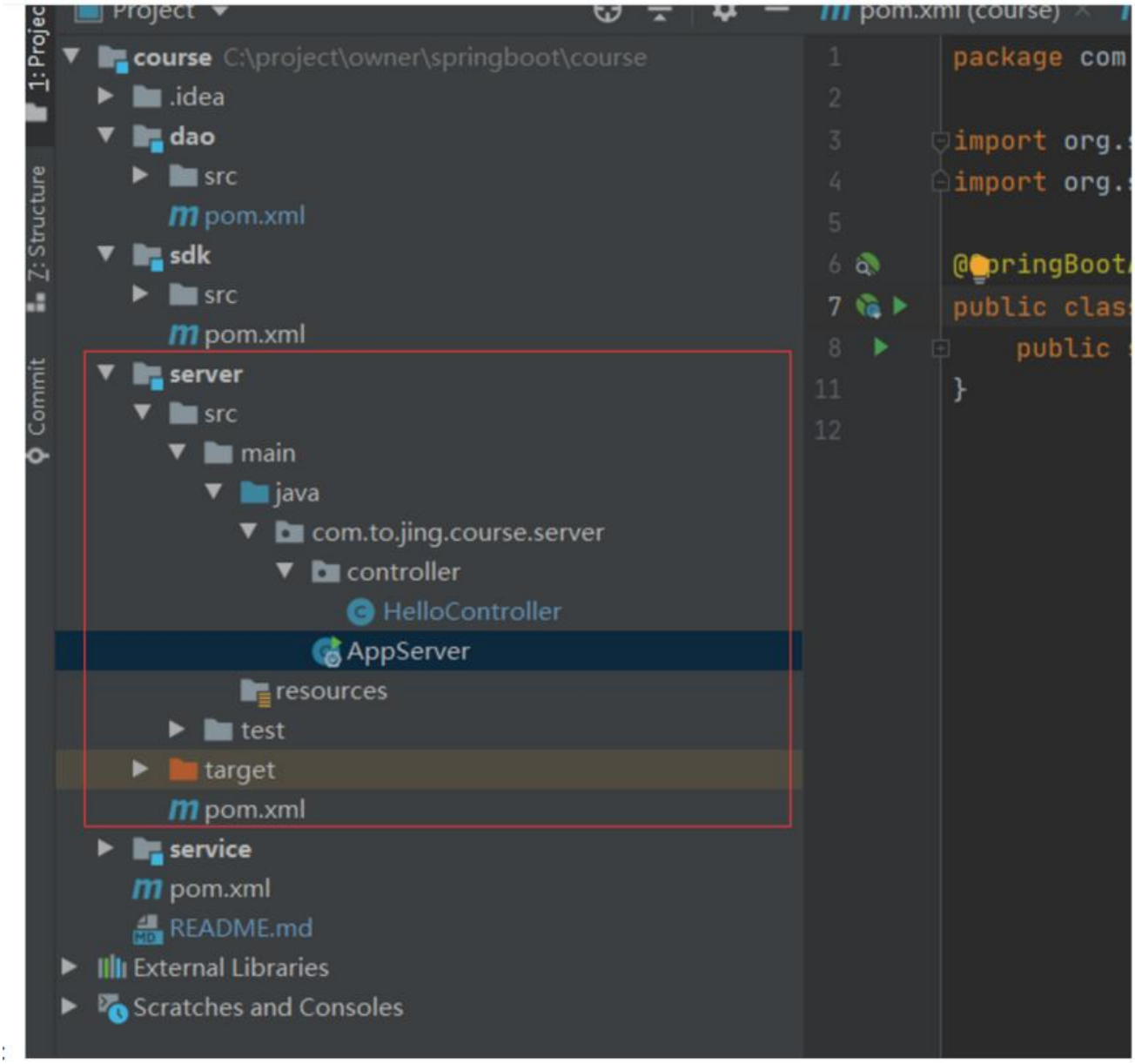
包下com.to.jing.course.server再建一个包controller,新建HelloController.java,代码如下：

```
package com.to.jing.course.server.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HelloController {
    @RequestMapping("/")
    @ResponseBody
    public String hello(){
        return "hello,world";
    }
}
```

完整项目结构图如下：



点击运行AppServer,再浏览器中打开地址http://localhost:8080/,结果如下图：

引入springboot依赖成功。

源码地址

https://github.com/ToJing/spring-boot-course tagV1.0

结语

该章节讲解的比较详细，后续会视情况进行简略。关于Maven构建多项目其中的pom文件的编写，面许多标签的含义大家可以深入了解下。

参考

- 书籍《基于Springboot实现Java分布式中间件开发入门与实战》