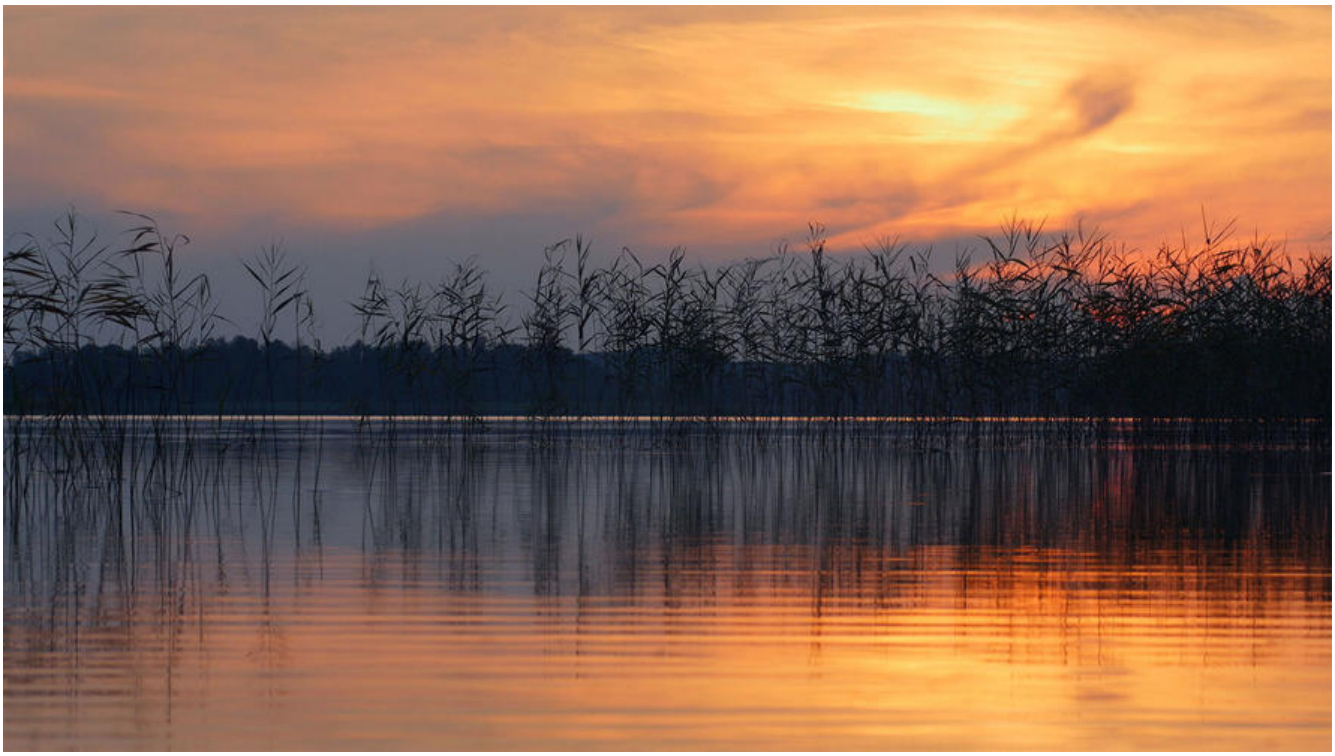
链滴

Springboot(二)、子模块层级依赖

作者：TOJing

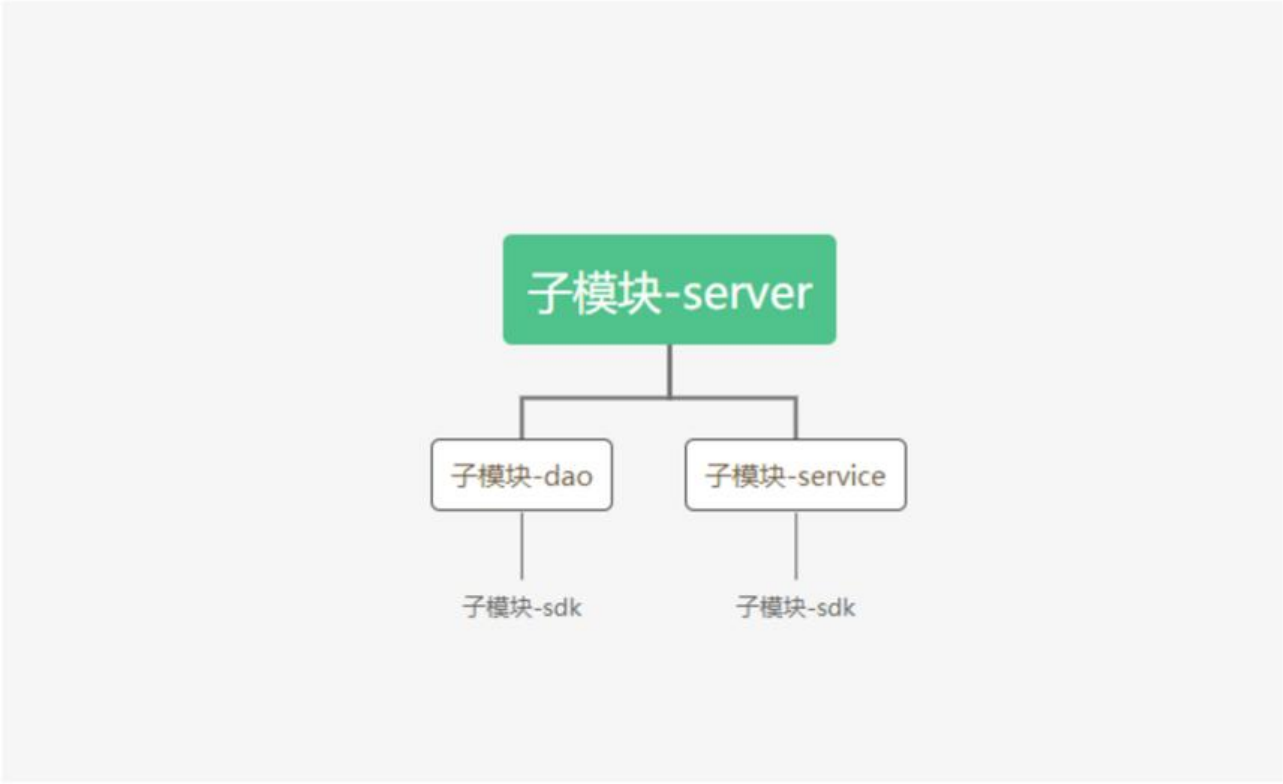
原文链接：https://ld246.com/article/1604914851989

来源网站：链滴



Springboot(二)、子模块层级依赖

在上一章节中创建了基于Maven管理的多模块项目，这一节将介绍如何实现模块之间的依赖。



从图中可以看出sdk实体层会被dao层和service依赖，因为代码里会引用实体，同理dao层和service又会被server依赖。serive层主要是定义一些接口，并不进行具体的实现，具体的实现是在server模，用过dubbo的都知道服务消费端和提供端会引用一个公共的service接口。而在某些项目中也有人接将公共service接口定义在了sdk中，本项目为了更清晰点将其拆开了。

技术选用

- idea2020
- mybatis
- mysql
- navicat
- maven
- springboot

子模块层级依赖

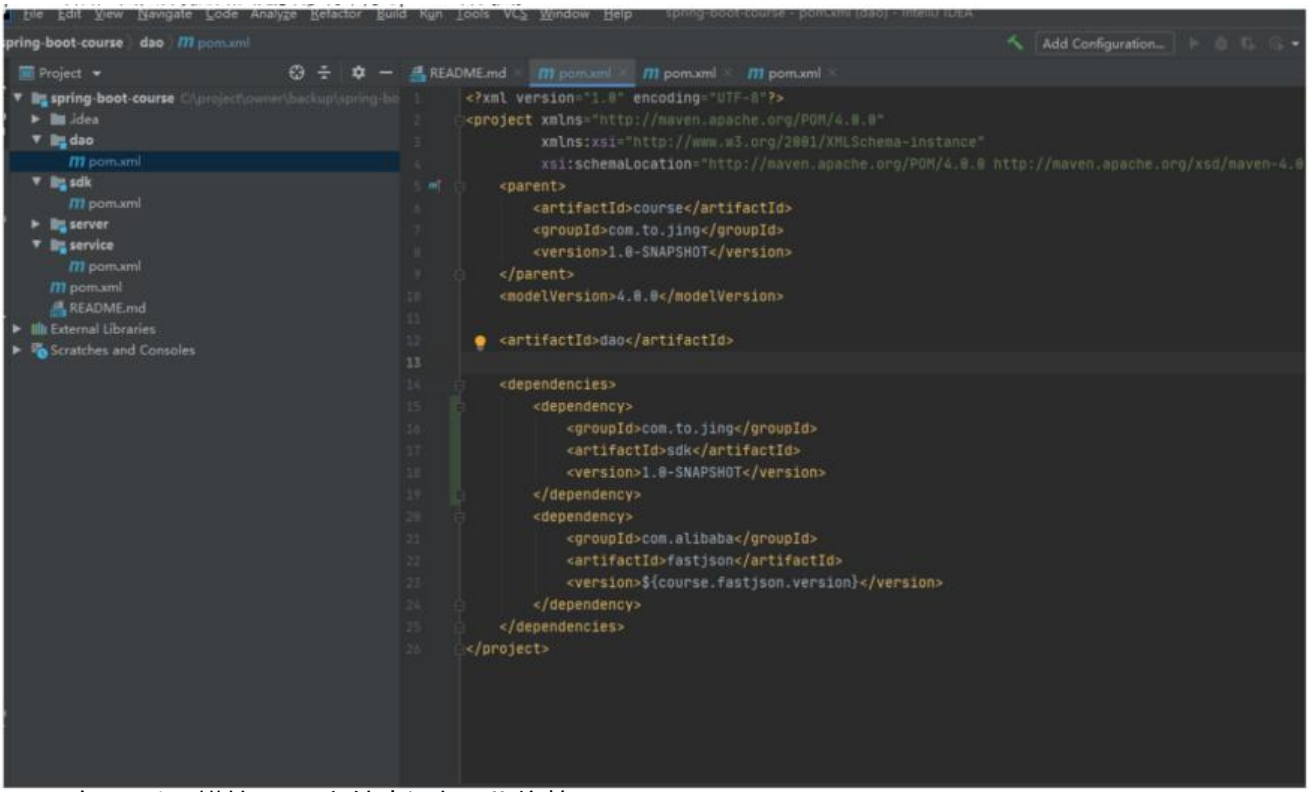
首先我们将根据上图来实现子模块之间的层级依赖。

（1）在dao模块pom文件中添加sdk依赖

在<dependencies>标签下添加代码：

```
<dependency>
    <groupId>com.to.jing</groupId>
    <artifactId>sdk</artifactId>
    <version>1.0-SNAPSHOT</version>
</dependency>
```

pom文件如下图，后续随着依赖的增多将不再对pom文件截图。



（2）在service模块pom文件中添加sdk依赖

在<dependencies>标签下添加代码：

```
<dependency>
```

```
    <groupId>com.to.jing</groupId>
    <artifactId>sdk</artifactId>
    <version>1.0-SNAPSHOT</version>
</dependency>
```

（3）在server模块pom文件中添加service和sdk依赖

在<dependencies>标签下添加代码：

```
<dependency>
    <groupId>com.to.jing</groupId>
    <artifactId>service</artifactId>
    <version>1.0-SNAPSHOT</version>
</dependency>
<dependency>
    <groupId>com.to.jing</groupId>
    <artifactId>dao</artifactId>
    <version>1.0-SNAPSHOT</version>
</dependency>
```

整合mybatis

添加完子模块层级依赖后，我们写个完整的项目Helloword来测试下，同时也对各个模块的功能划分代码的编写有个清晰的认识。。

- 添加库表结构

先创建数据库course,在添加用户表user，插入一条数据。代码如下：

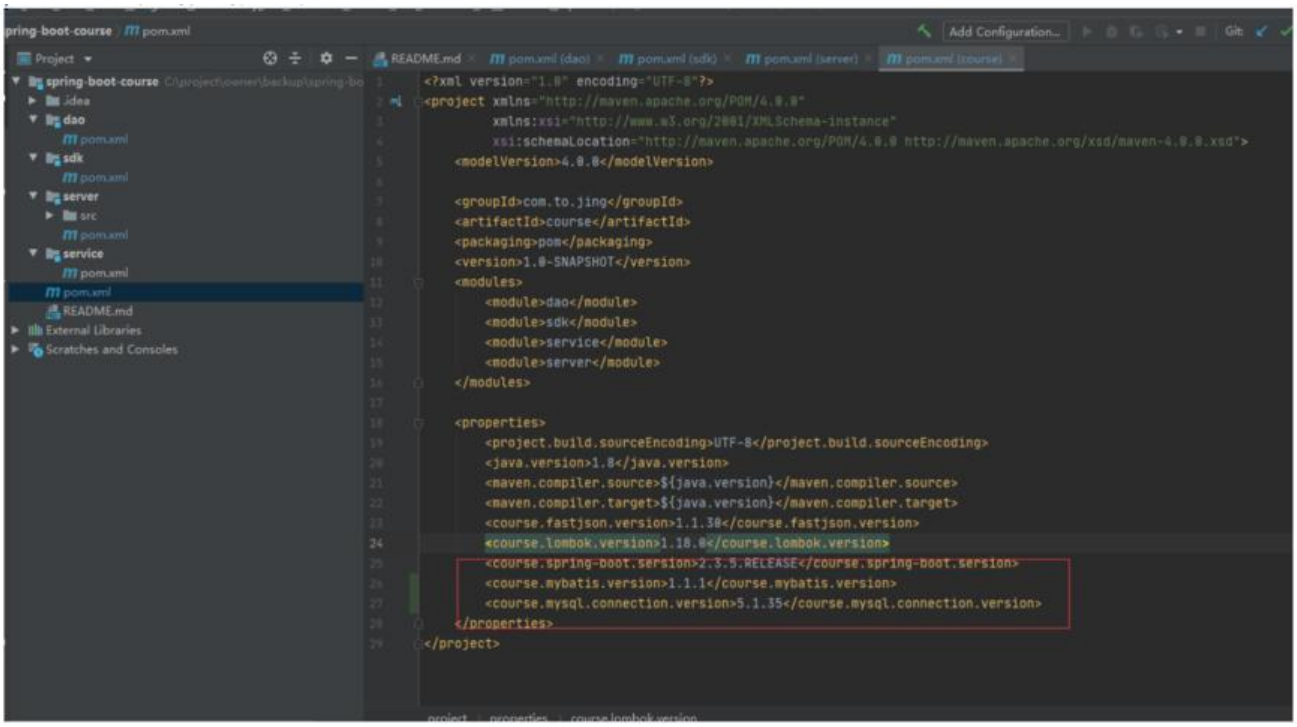
```
DROP TABLE IF EXISTS `user`;
CREATE TABLE `user`  (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT 'id',
  `username` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NULL DEF
ULT NULL COMMENT '姓名',
  `password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NULL DEFA
LT NULL COMMENT '密码',
  `age` int(11) NULL DEFAULT NULL COMMENT '年龄',
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 2 CHARACTER SET = utf8mb4 COLLATE = utf8mb4
general_ci ROW_FORMAT = Dynamic;

-- ----------------------------
-- Records of user
-- ----------------------------
INSERT INTO `user` VALUES (1, 'asd', 'asd', 12);
```

- dao层引入mybatis依赖

在父模块中pom文件中设置mybatis和mysql的版本，在properties标签下添加如下代码

```
<course.mybatis.version>1.1.1</course.mybatis.version>
<course.mysql.connection.version>5.1.35</course.mysql.connection.version>
```

pom文件如下图，后续将不再进行截图。



然后在dao层引入mybatis和mysql依赖。

```
<!-- mybatis依赖 -->
        <dependency>
            <groupId>org.mybatis.spring.boot</groupId>
            <artifactId>mybatis-spring-boot-starter</artifactId>
            <version>${course.mybatis.version}</version>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>${course.mysql.connection.version}</version>
        </dependency>
```

- sdk模块中创建user实体

sdk模块中主要存放实体，新建包com.to.jing.course.sdk.domain，在包下新建User.java实体。

```
package com.to.jing.course.sdk.domain;

import lombok.Data;

@Data
public class User {
    private Integer id;
    private String username;
    private String password;
    private Integer age;
}
```

- dao模块创建mapper

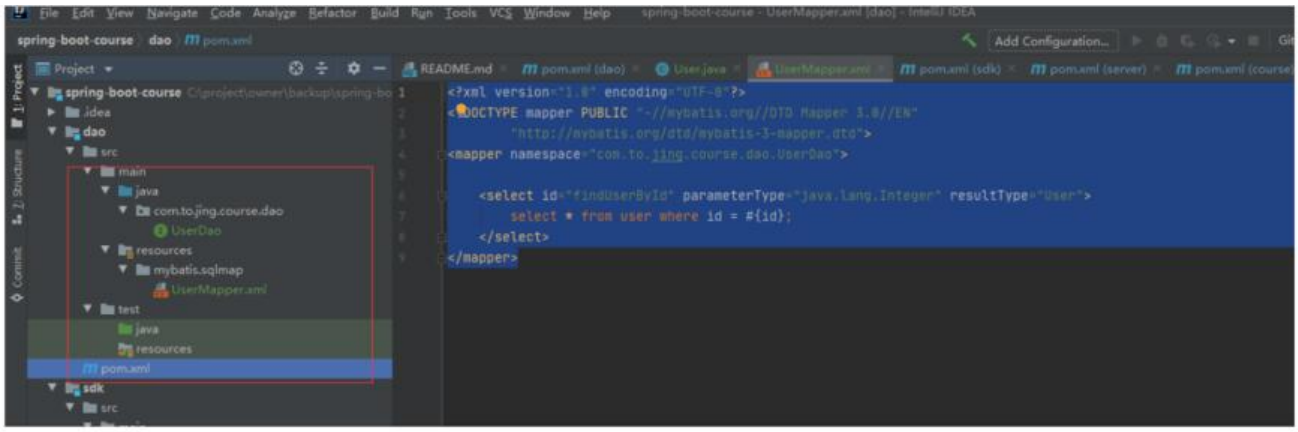
在dao模块编写sql.xml文件和数据库映射方法接口。创建包com.to.jing.course.dao，在包下新建接UserDao。注意接口前要加@Mapper注解。

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

import com.to.jing.course.sdk.domian.User;
import org.apache.ibatis.annotations.Mapper;

@Mapper
public interface UserDao {
    User findUserById(Integer id);
}
```

然后在resources下新建目录mybatis/sqlmap，在目录下新建UserMapper.xml文件。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.to.jing.course.dao.UserDao">

    <select id="findUserById" parameterType="java.lang.Integer" resultType="User">
        select * from user where id = #{id};
    </select>
</mapper>
```

完整项目结构如下：



● service模块创建服务接口

service下主要定义一些公共服务接口，用于服务的发布。新建包com.to.jing.course.service,在包下建接口UserService。

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

import com.to.jing.course.sdk.domian.User;

public interface UserService {
    User findUserById(Integer id);
}
```

● server模块实现业务逻辑

先添加配置文件，在resources中新建application.yaml文件，配置mybatis，原来有xml文件的可以掉，或者自行配置xml内容。

```
spring:
```

```
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    url: jdbc:mysql://192.168.119.110:3306/course?characterEncoding=UTF-8
    username: root
    password: 123456
mybatis:
  mapper-locations: classpath:mybatis/sqlmap/**/*.xml
  type-aliases-package: com.to.jing.course.sdk.domain
```

新建包com.to.jing.course.server.service.impl,在包下新建UserServiceImpl.java文件,实现UserServie接口。

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

import com.to.jing.course.dao.UserDao;
import com.to.jing.course.sdk.domian.User;
import com.to.jing.course.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class UserServiceImpl implements UserService {
    @Autowired
    private UserDao userDao;
    @Override
    public User findUserById(Integer id) {
        return userDao.findUserById(id);
    }
}
```

注意，这里@Autowired会在idea中显示一个波浪线，因为springboot后面不推荐使用这种注入方式暂时先不管。

修改上一节HelloController中的代码为：

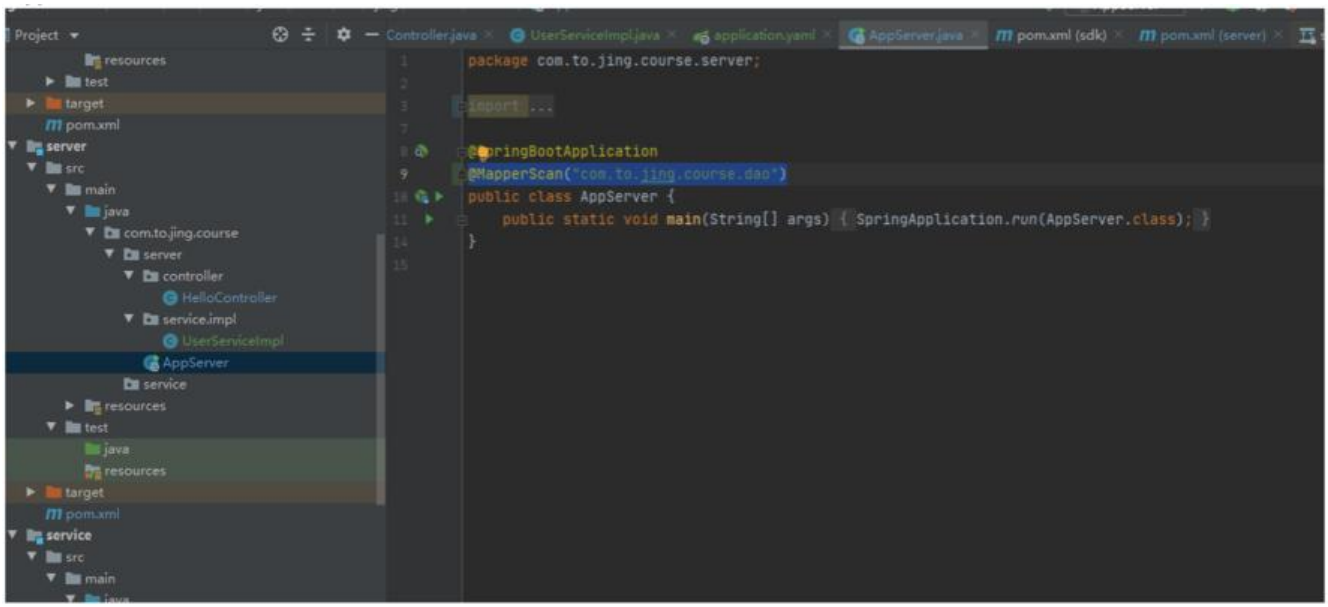
```
package com.to.jing.course.server.controller;

import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.serializer.SerializerFeature;
import com.to.jing.course.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HelloController {
    @Autowired
    private UserService userService;
    @RequestMapping("/")
    @ResponseBody
    public String hello(){
        return JSON.toJSONString(userService.findUserById(1), SerializerFeature.BrowserCompat
ble);
    }
```

}

在AppServer上添加@MapperScan注解

@MapperScan("com.to.jing.course.dao")

项目结构如下图

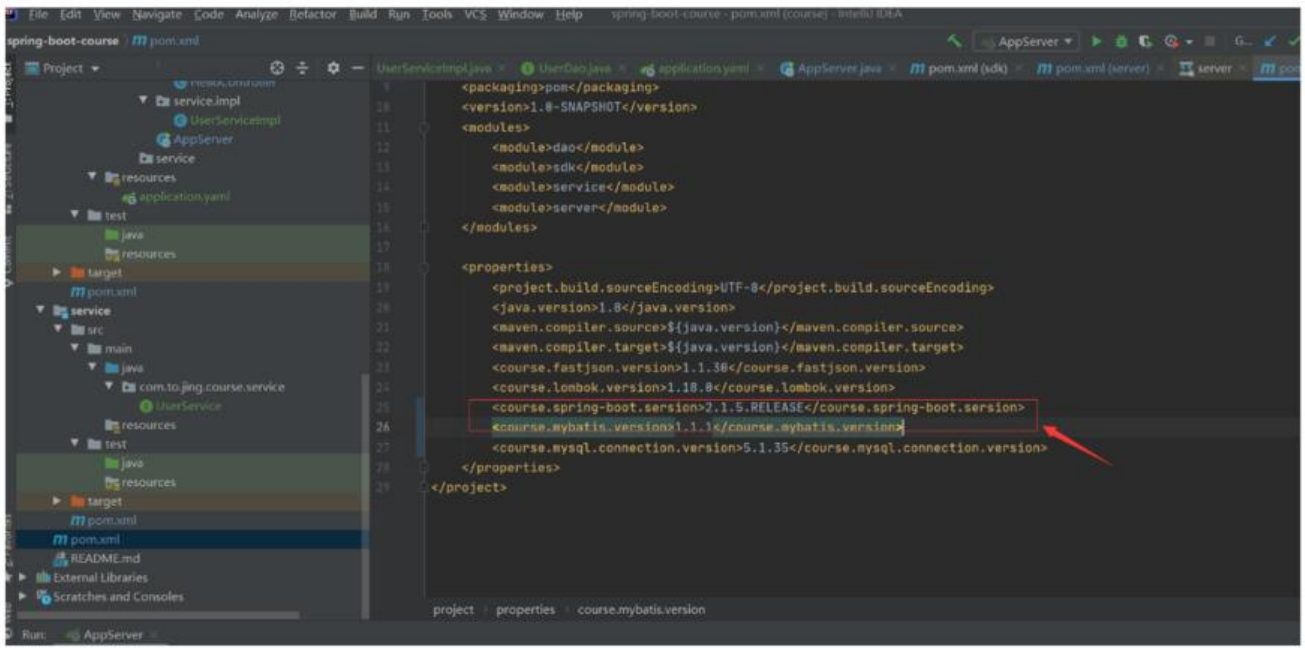


此时运行AppServer显示如下报错：

原因为jar版本冲突了，我们可以去到server模块pom文件下通过快捷键ctrl+alt+shift+u，查看如下。

红色就证明存在jar冲突，所以我们这里在父模块pom文件中替换springboot的版本从2.3.5.RELEASE 2.1.5.RELEASE。

<course.spring-boot.sersion>2.1.5.RELEASE</course.spring-boot.sersion>



再次启动AppServer，在浏览器中查看http://localhost:8080/，结果如下图。

源码地址

https://github.com/ToJing/spring-boot-course tagV1.1

博客地址

http://m.loveplaycat.club/articles/2020/11/09/1604914780246.html