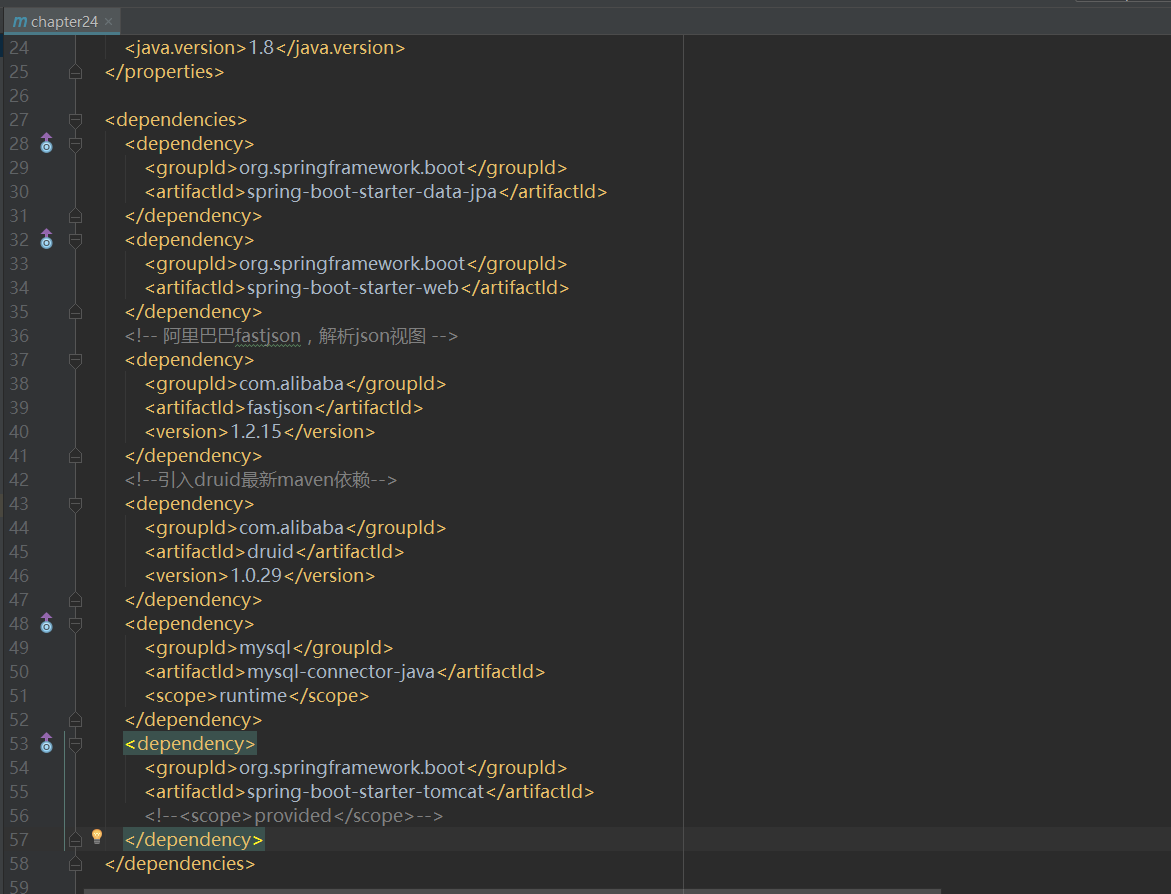
**SpringBoot项目整合JPA多数据源配置**

基于SpringBoot项目整合SpringDataJpa完成多数据源根据包名自动切换。

# 构建项目

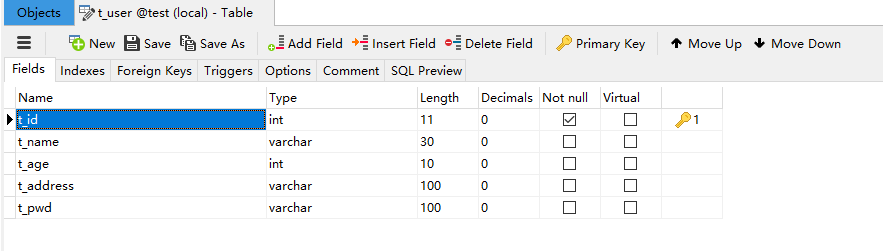
我们使用IDEA工具创建一个SpringBoot项目，并添加JPA、Web、FastJson、Druid等依赖，pom.xml如下图1所示：



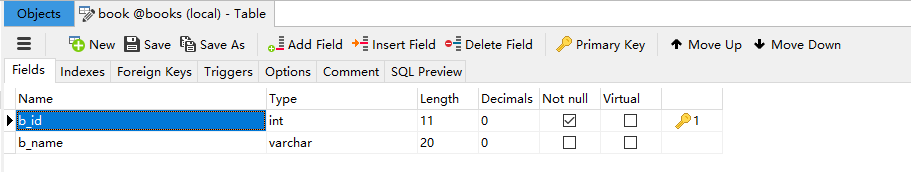
## 创建数据库

在正式编写项目之前，我们首先创建两个数据库（test、books）test数据库内包含了一张数据表t\_user，books数据库内包含了一张数据库book《表结构在项目源码resources目录下》

t\_user表结构如下图2所示：



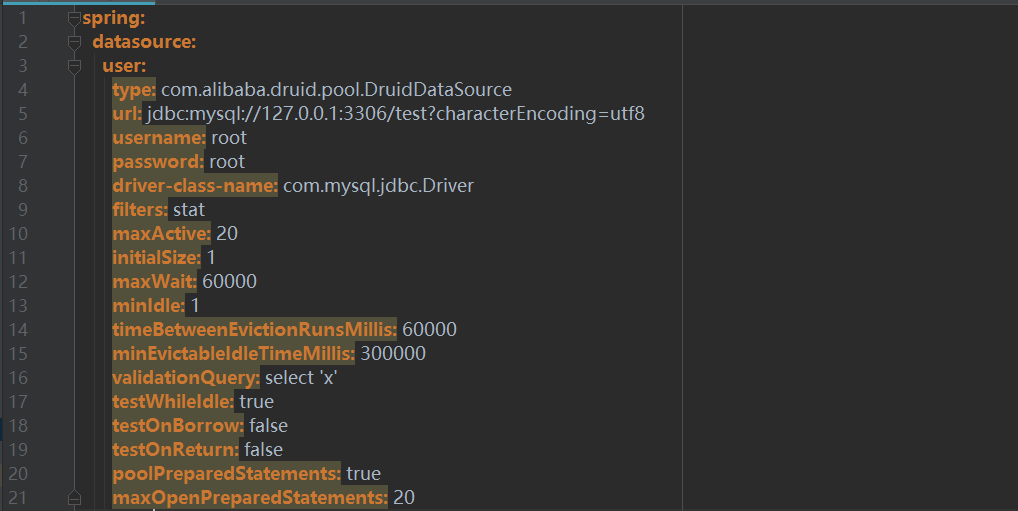
book表结构如下图3所示：



## 编写application.yml配置文件

在我们之前单数据源单数据库的示例中我们的配置都是比较简单的，下面我们来看下我们多数据源的配置

用户数据库数据源配置如下图4所示：



书籍数据库数据源配置如下图5所示：

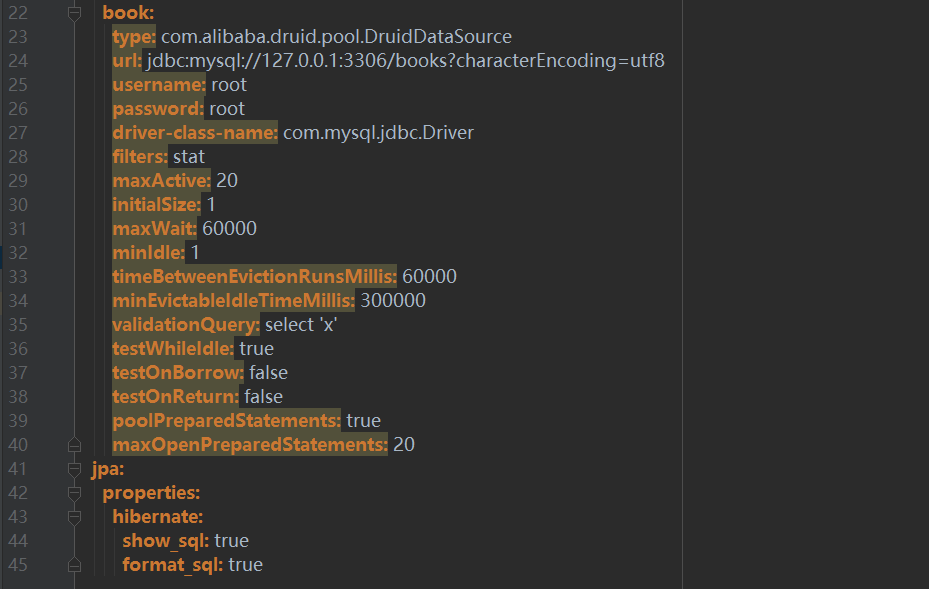


图5

可以看到上面两个数据源命名时都有前缀，分别是user、book。这个是我们配置时必要的属性，下面我们就来编写数据库源的映射配置类。

## DataSourceConfigurer

我们自定义一个数据源配置类DataSourceConfigurer，在类内对应声明两个数据源的Bean对象，以及使用application.yml配置文件内的前缀属性配置，如下图6所示：

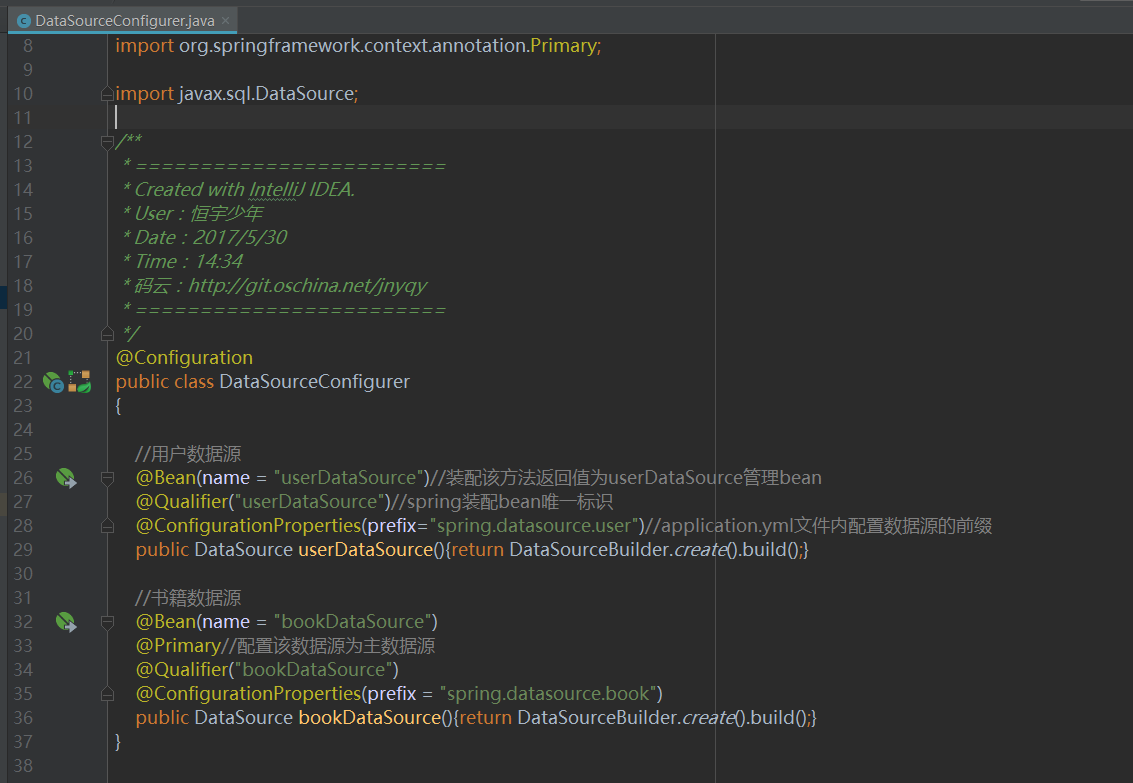


图6

    上图6内我们对应两个数据库分别定义了两个数据源，userDataSource、bookDataSource。  
    使用@Qualifier注解并且设置内容，是为了防止注入时冲突问题。  
    userDataSource数据源配置使用了application.yml文件中spring.datasource.user前缀的配置，而bookDataSource则是配置使用了spring.datasource.book前缀的配置。

#### @Primary

@Primary配置了数据源为主数据源，当没有配置自动切换的package时默认使用该数据源进行数据处理操作。

数据源的声明我们已经创建完成，下面我们来详细的实现对应的数据源。

## BookDataSourceConfigurer

bookDataSourceConfigurer配置文件类是详细实现了DataSourceConfigurer类中声明的bookDataSource，实现类中大致分为了如下及部分：

#### 配置EntityManager

配置让SpringDataJpa管理实体，根据EntityManagerFactory工厂的参数构建出EntityManager，如下图7所示：

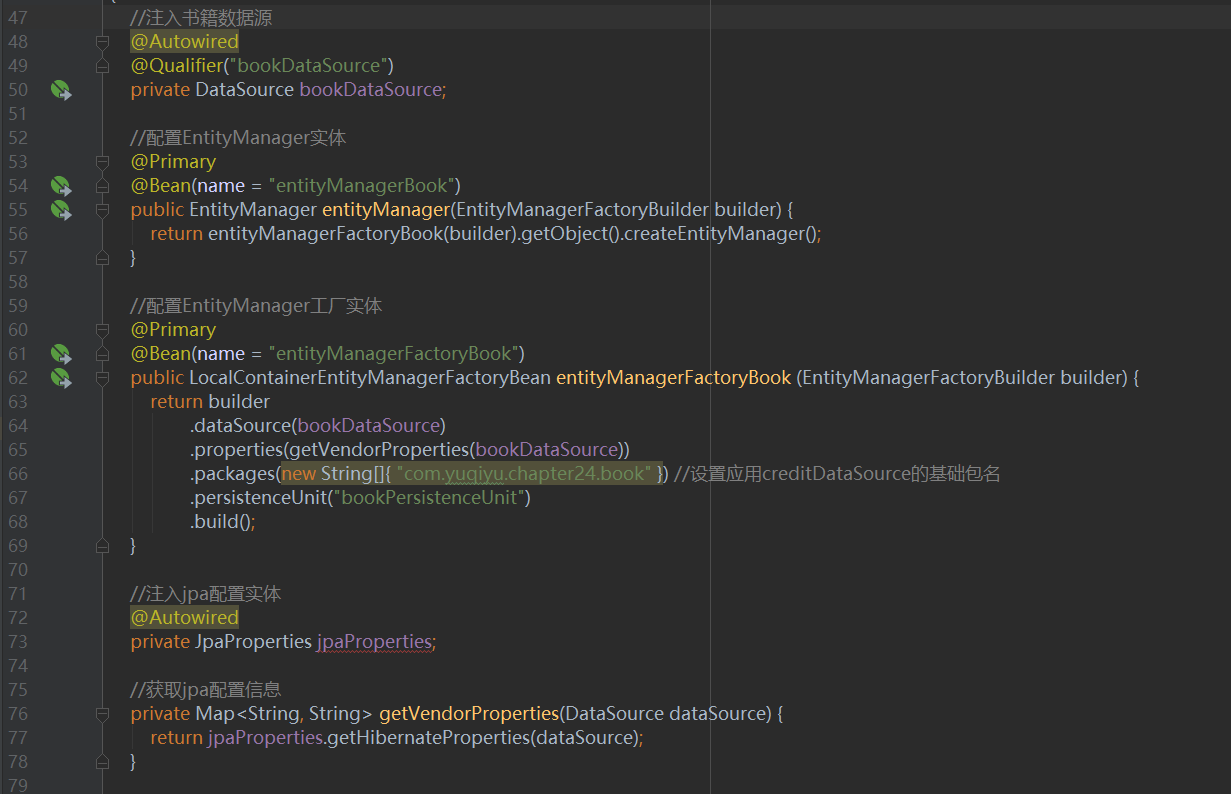


图7

#### 配置TransactionManager

根据EntityManagerFactory创建事务Manager，让事务应用到实体工厂内，如下图8所示：

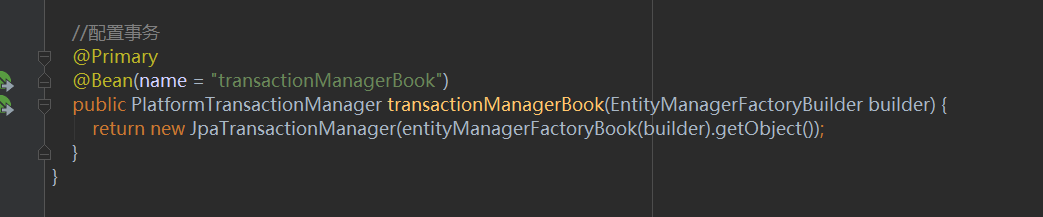


图8

因为DataSource的声明跟实体管理器、事务管理器是分离的所以在上图7、图8内仍然需要使用@Primary注解来配置主实体管理器、主事务管理器。

类内的配置已经添加完成，现在我们需要指定BookDataSourceConfiguer来使用对应的配置，如下图9所示：

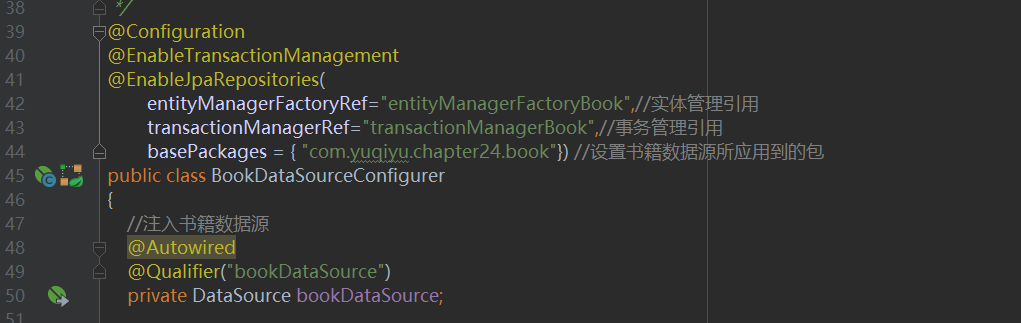
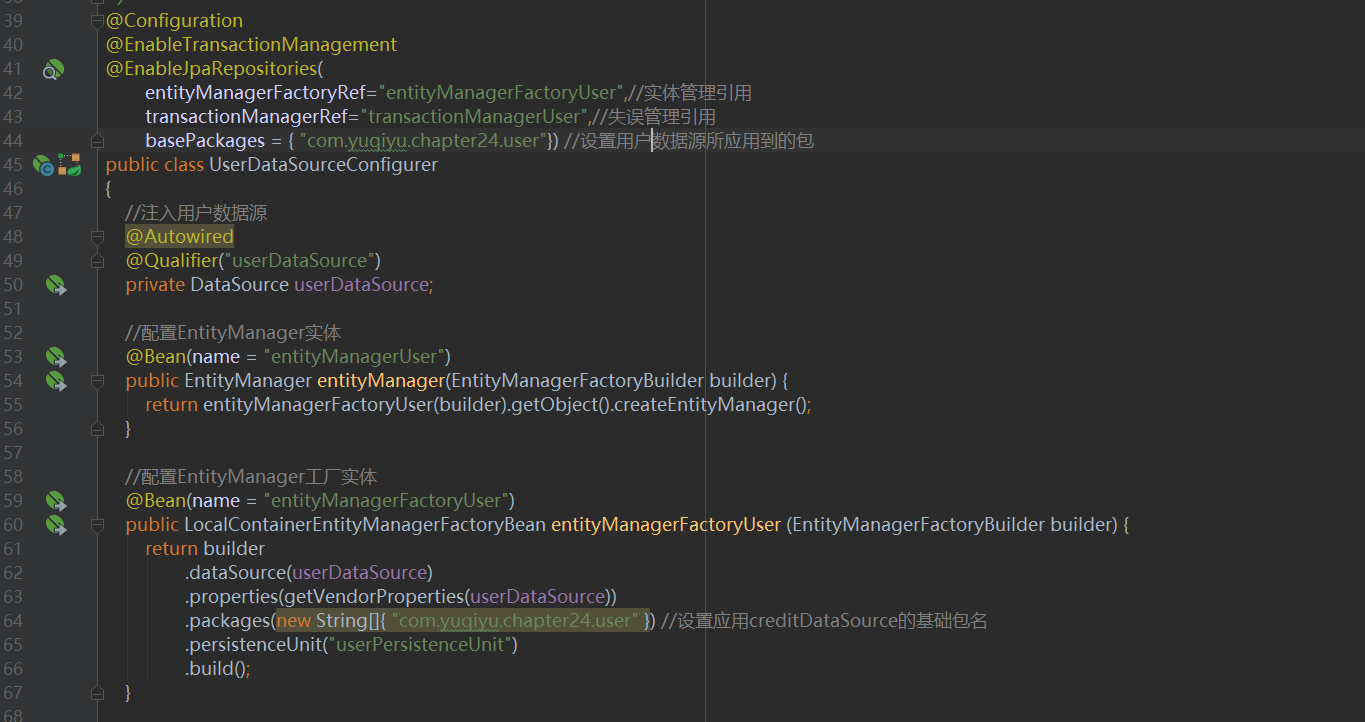


图9

我们在类上添加开启了事务配置以及启用了SpringDataJpa的配置参数，设置事务、实体管理器、数据源所应用到的包，当使用该包下的实体以及数据接口时就会自动调用bookDataSource来处理数据。userDataSource的实现跟bookDataSource的实现几乎一样，只是少了@Primart注解以及修改对应的参数等，如下图10所示：

图10

userDataSource事务配置如下图11所示：

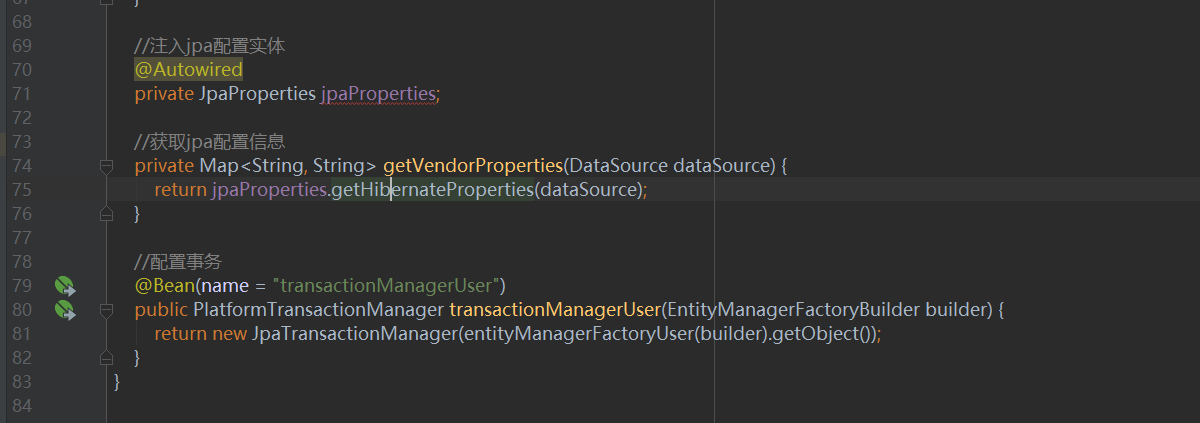


图11

## 查询书籍列表接口

我们来编写查询书籍列表的接口，创建对应的package，com.yuqiyu.chapter24.book，创建controller.BookController控制器如下图12所示：

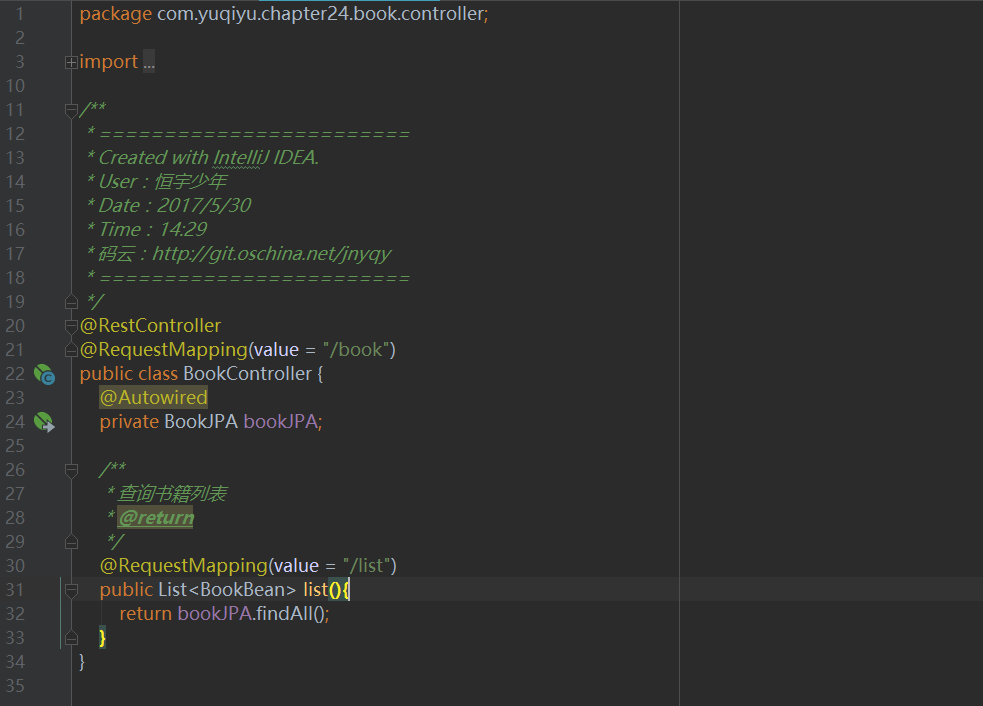
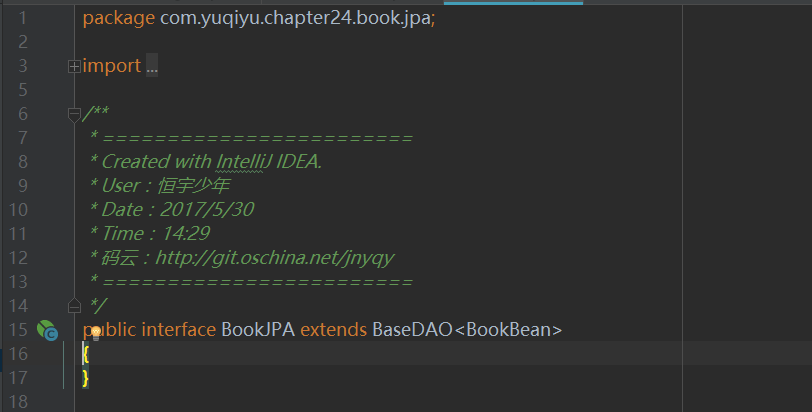


图12

上图12中注入了BookJPA，调用了jpa内部的findAll方法来读取全部数据列表，并通过@RestController注解作用返回Json字符串。

创建jpa.BookJPA数据接口，如下图13所示：

图13

可以看到BookJPA继承了BaseDAO父泛型接口，BaseDAO接口内统一继承了JpaRepository、JpaSpecificationExecutor等公用接口，子接口只需要继承BaseDAO就可以拥有SpringDataJPA内的内置方法，BaseDAO源码如下图14所示：

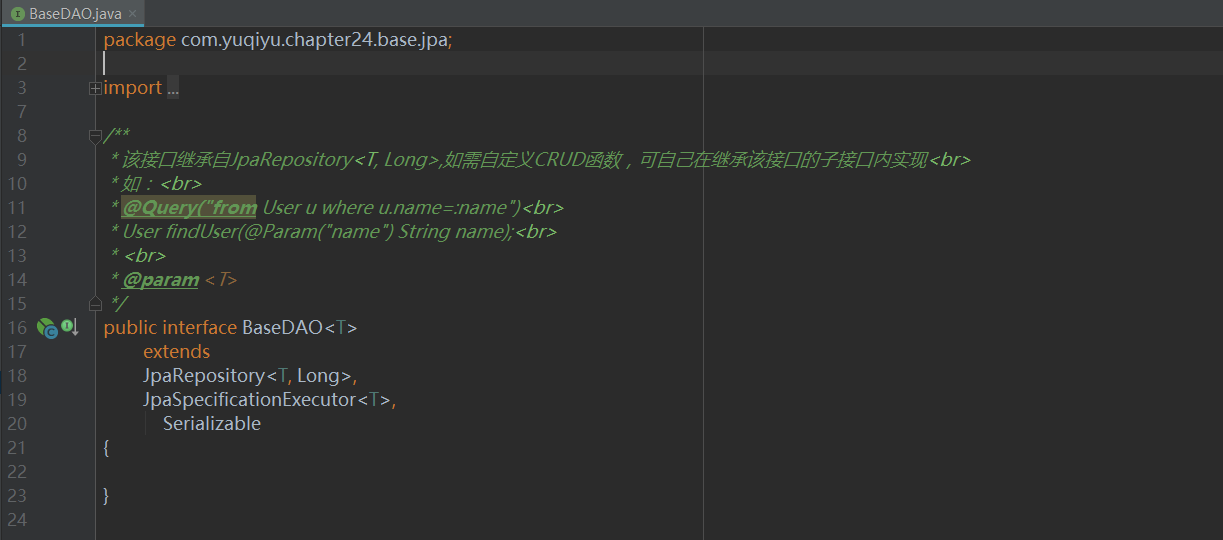


图14

**查询用户列表接口**

跟上面书籍列表接口一样，我们先来创建对应的package,com.yuqiyu.chapter24.user，创建用户控制器controller.UserController，如下图15所示：

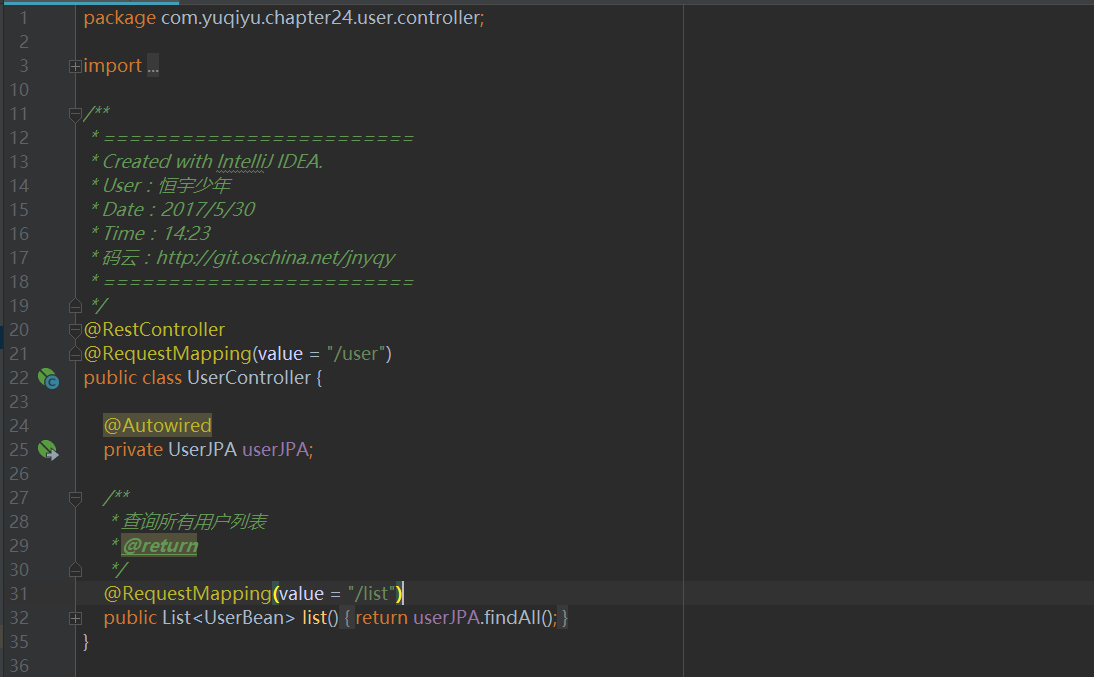


图15

上图15内引用的userJPA也是跟bookJPA一样，继承了BaseDAO泛型接口，这里就不做出讲解了。

**运行测试**

上面步骤我们编码已经完成，下面我们开启项目来测试下数据是否是对应不用的数据库获取的，运行项目控制台输出内容如下图16所示：

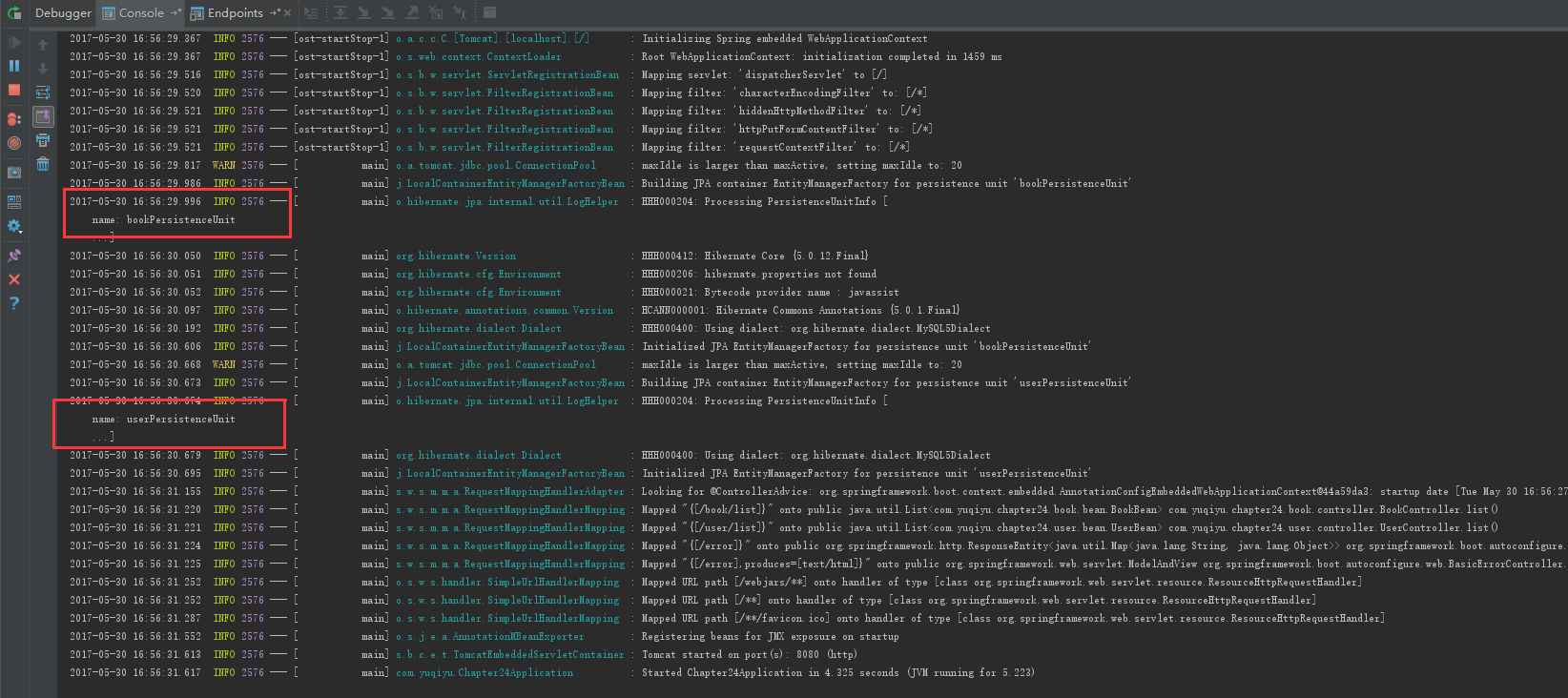
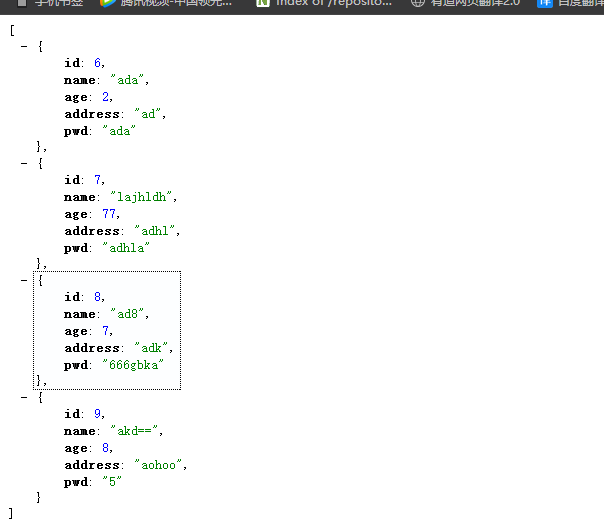


图16

上图16内可以看到项目启动时加载了我们配置的两个数据源，这个name的配置对应数据源实现类内。下面我们来访问下用户列表：[127.0.0.1:8080/user/list](http://127.0.0.1:8080/user/list" \t "_blank)，界面输出内容如下图17所示：



可以看到我们成功的读取到了test数据库内的t\_user表内的数据。我们下面再来访问下书籍列表：[127.0.0.1:8080/book/list](http://127.0.0.1:8080/book/list)，界面输出内容如下图18所示：

