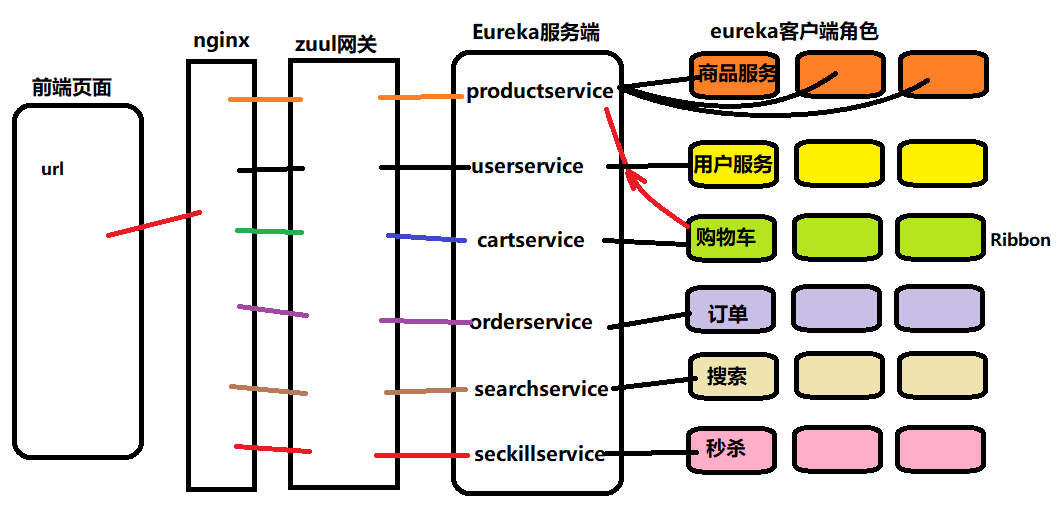
**搭建EasyMall项目**

# easymall项目结构

## easymall项目介绍

模拟的目前市面比较常见的电商系统,包括商品相关功能,用户相关功能,订单,购物车相关功能,搜索功能,秒杀功能;经过纵向拆分,利用springcloud框架搭建的一个高可用,高并发,高性能的系统;

## easymall项目结构



# 2.搭建easymall基础项目

## 2.1 parent工程

### 2.1.1 使用父级工程合理管理资源

图示

描述已自动生成

### 2.2.2搭建一个父级工程

#### 选择quickstart工程(可以删除libiray，src)

easymall-parent

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

删除src/main/java和src/test/java文件夹

图形用户界面, 应用程序

描述已自动生成

删除JRE SystemLibrary和Maven Dependencies

图形用户界面, 应用程序, Word

描述已自动生成

删除src文件夹

图形用户界面, 应用程序, Word

描述已自动生成

#### 2.修改工程类型，父级工程的Packaging必须是pom

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

#### 3.pom.xml文件维护子工程公用资源

1. **继承springboot-parent**

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.9.RELEASE</version>

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<java.version>1.8</java.version>

</properties>

1. 尽量在dependencies中引入所有工程都用到的公用依赖。

* 对java.lang包里的api做扩展（Math,String）

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-lang3</artifactId>

<version>3.3.2</version>

</dependency>

* 引入Springboot的简化依赖测试资源

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

* 引入eureka和ribbon

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-eureka</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-ribbon</artifactId>

</dependency>

1. 父级工程导入springcloud的资源

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>Edgware.RELEASE</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

## 2.2 自定义一个持久层简化依赖工程

继承parent，依赖jdbc,mysql,mybatis。

作用：提供持久层的相关依赖给其他功能子系统，完成持久层依赖的传递功能。

### 2.2.1 搭建步骤

#### 1.创建一个quickstart工程

easymall-common-repository

#### 2.pom.xml文件

* 继承parent(用一下spring-boot-starter-parent)

<parent>

<groupId>cn.edu.scnu</groupId>

<artifactId>easymall-parent</artifactId>

<version>0.0.1-SNAPSHOT</version>

</parent>

图形用户界面, 文本, 应用程序

描述已自动生成

* 依赖jdbc,mysql,mybatis

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-jdbc</artifactId>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<!-- MyBatis -->

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

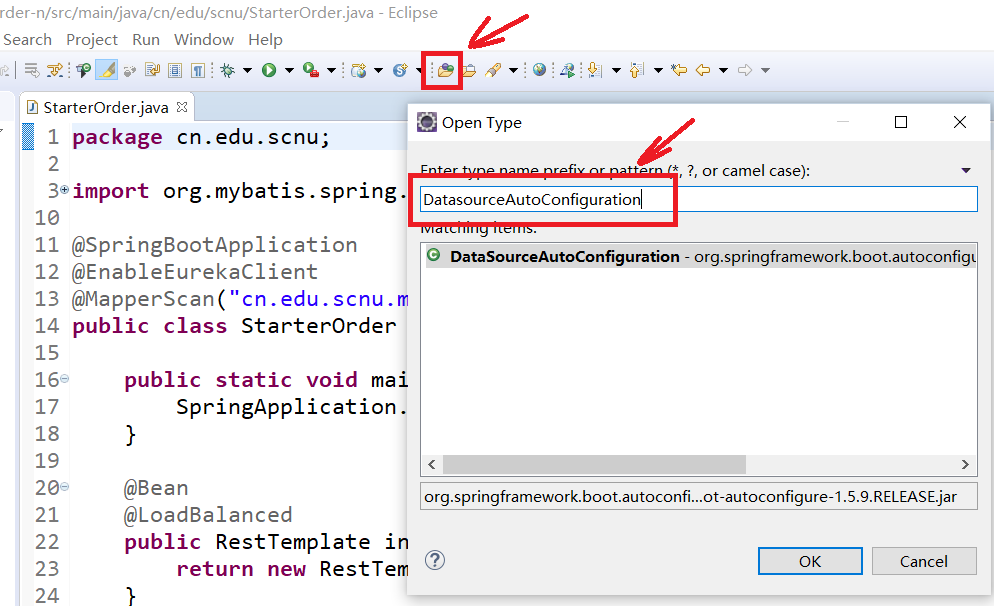
<version>1.3.0</version>

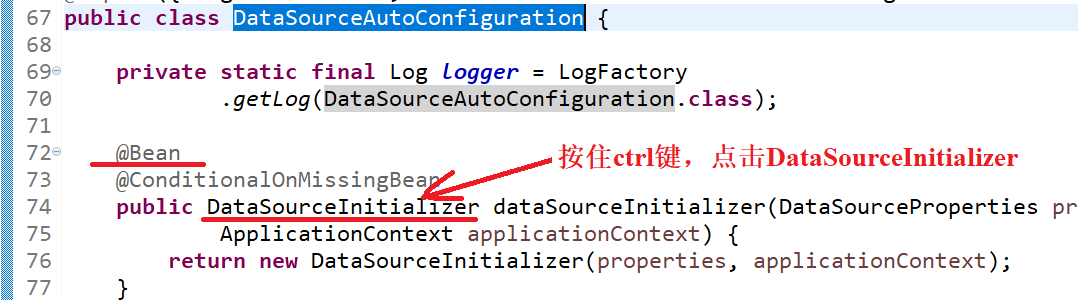
</dependency>

### 2.2.2 Druid连接池配置

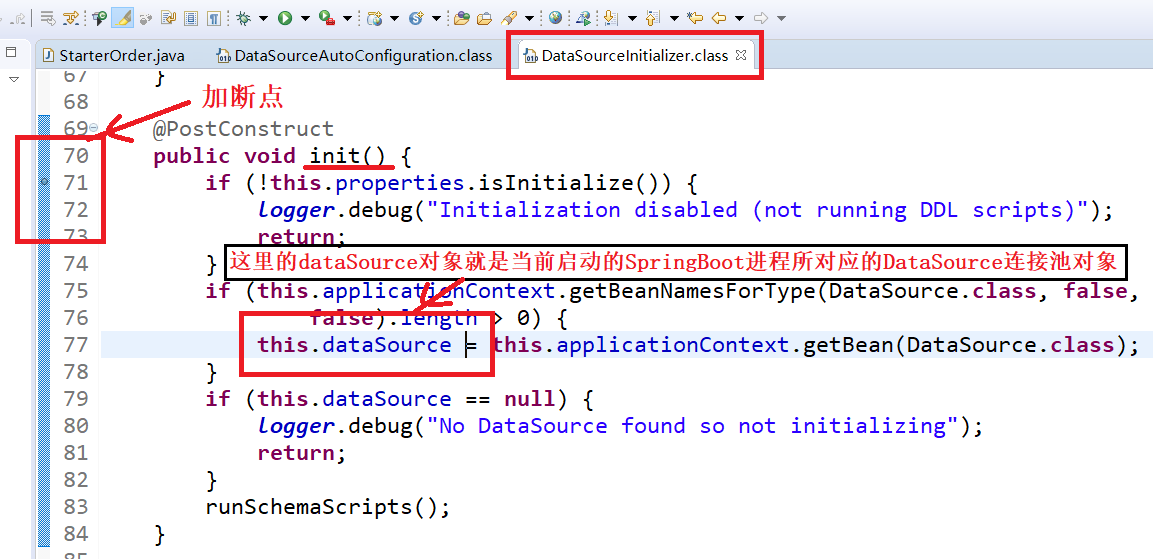
1. 查看默认连接池配置

自动配置启动类：



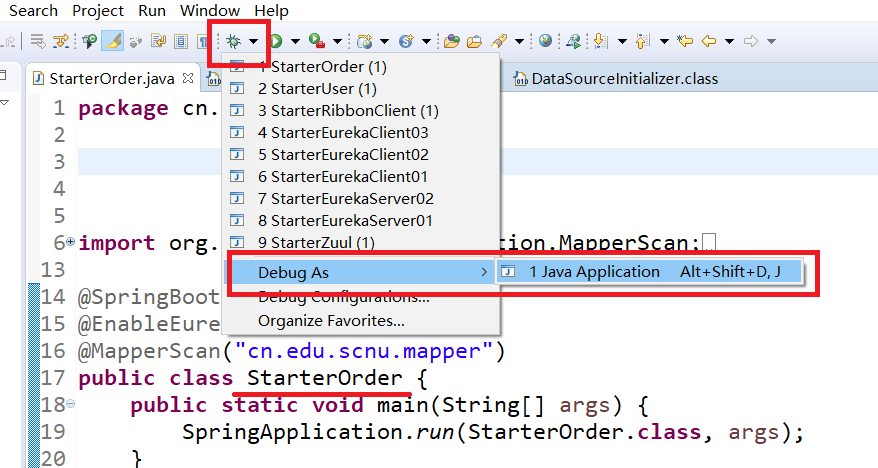


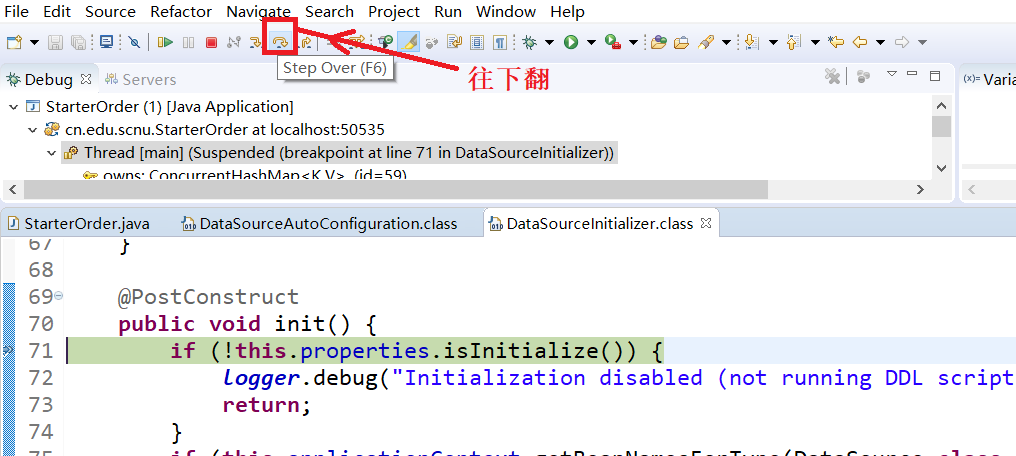
**按住ctrl键，点击DataSourceInitializer,打开DataSourceInitializer**



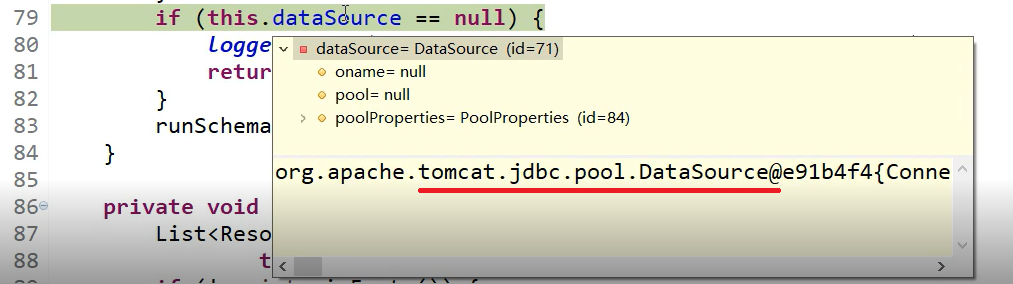
**这里的dataSource对象就是当前启动的SpringBoot进程所对应的DataSource连接池对象**。**我们通过加断点，来查看SpringBoot默认配置的的连接池。**

Debug As **StarterOrder**启动类进入断点，可以看到默认使用的是tomcat的连接池。

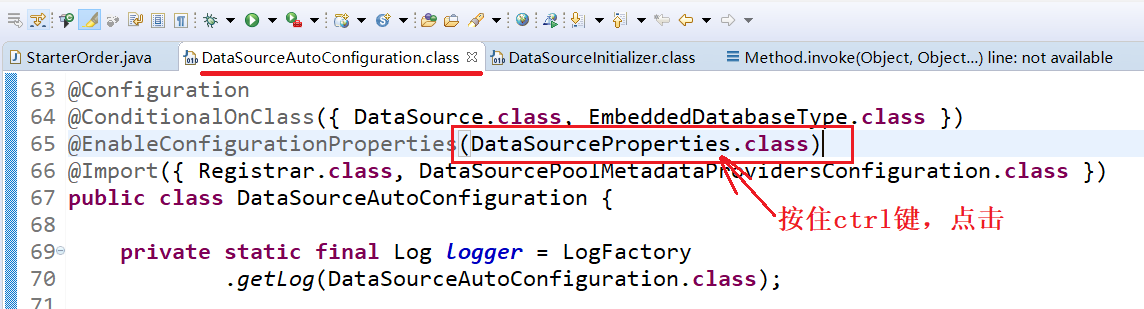




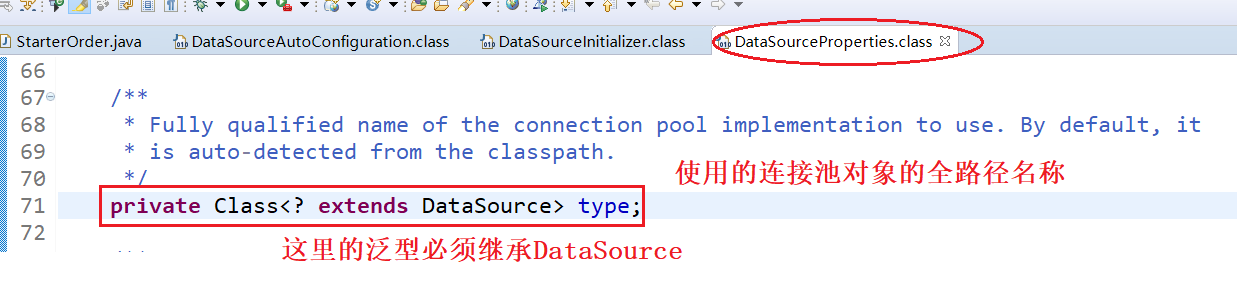
鼠标移到dataSource对象上面，可以看到springboot默认使用的是tomcat的连接池。



在DataSourceProperties中查看连接池的类型定义:



如果不指定type的值，则用额是默认的连接池。



我们可以在配置文件中通过指定type的值，来定义自定义连接池。

1. 在springboot-order中配置连接池对象

根据spring.datasource.type的连接池全路径名称，在springboot-order中配置新建连接池对象：

先在springboot-order的pom文件中加入Druid数据源依赖，再在配置文件中指定type的值。

<!-- datasource pool druid-->

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>druid</artifactId>

<version>1.0.14</version>

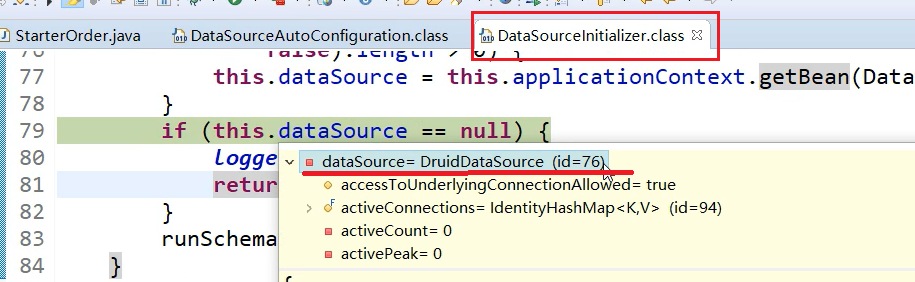
</dependency>

配置properties

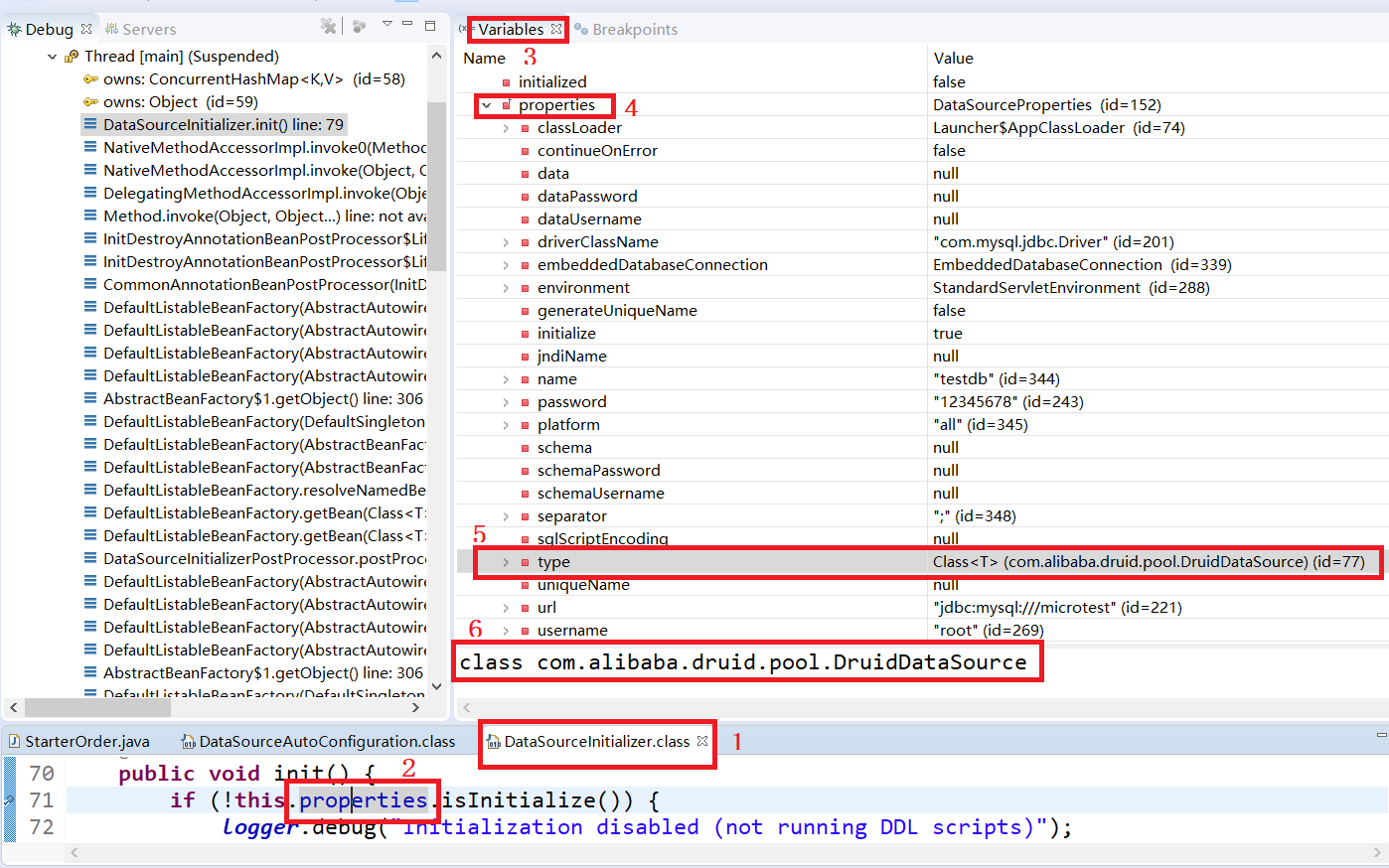
在springboot-order的配置文件中配置连接池对象的全路径名称

**spring.datasource.type=com.alibaba.druid.pool.DruidDataSource**

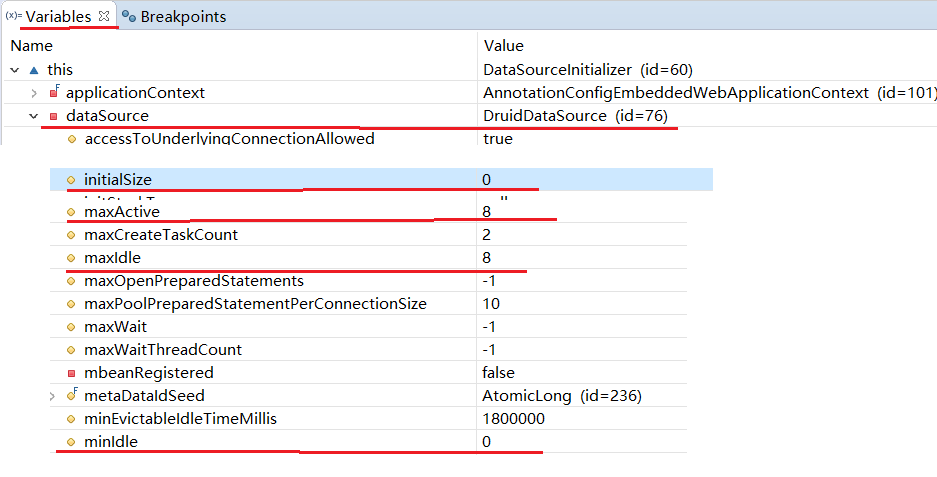
**再次** Debug As **StarterOrder**启动类进入断点，可以看到使用的是**Druid**。

****

**查看properties中的type,可以看到引入的是Druid连接池。**

****

虽然根据上面定义的全路径名称就可以创建了连接池对象,但对于连接池中的各种属性,包括maxActive最大连接数,initialSize初始化连接数量,minIdle最小空闲,maxIdle最大空闲数等属性，还是保持默认值。从下面dataSource可以查看到：



如果想对创建的DruidDatasource连接池对象配置属性值,需要自定义编写对象的创建过程;

1. 在easymall-common-repository中创建自定义初始化属性的配置类

创建之前，先在pom文件中加入Druid数据源依赖。

<!-- datasource pool druid-->

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>druid</artifactId>

<version>1.0.14</version>

</dependency>

为了对**创建的DruidDatasource连接池对象**配置属性值，我们在easymall-common-repository中创建自定义初始化属性的配置类

**package** cn.edu.scnu.repository.config;

**import** javax.sql.DataSource;

**import** org.springframework.boot.context.properties.ConfigurationProperties;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.context.annotation.Primary;

**import** com.alibaba.druid.pool.DruidDataSource;

@Configuration

@ConfigurationProperties("spring.datasource")

**public** **class** DataSourceInitConfig {

**private** String driverClassName;

**private** String url;

**private** String username;

**private** String password;

**private** Integer maxActive; //最大连接数

**private** Integer maxIdle; //最大空闲连接数

**private** Integer initialSize; //初始化连接数量

**private** Integer minIdle; //最大空闲数

//自定义创建的DruidDatasource对象返回给框架使用

@Bean

@Primary//当容器中存在多个同类对象时,以Primary所在的优先级最高

**public** DataSource initDruidDataSource(){

DruidDataSource datasource=**new** DruidDataSource();

datasource.setDriverClassName(driverClassName);

datasource.setUrl(url);

datasource.setUsername(username);

datasource.setPassword(password);

//连接池初始化参数

datasource.setInitialSize(initialSize);//5

datasource.setMaxActive(maxActive);//200

datasource.~~setMaxIdle~~(maxIdle);//8

datasource.setMinIdle(minIdle);//3

**return** datasource;

}

//setter&&getter

}

## 2.3 公用的工具工程

在系统开发过程中,多个系统可能用到同一个工具类,这种也叫作共享资源,可以单独提取一个维护**工具类,视图类,pojo类**的common-resource工程。

### 2.3.1使用工具工程

图示

描述已自动生成

### 2.3.2 搭建步骤

#### 1. 创建一个quickstart工程

easymall-common-resources

#### 2.pom文件

继承parent(用一下spring-boot-starter-parent)

图形用户界面, 文本

描述已自动生成

#### 3.拷贝资料common工具类工程代码

拷贝pojo包、utils包及vo包中的类。

#### 4.根据提示的编译报错调整其中的依赖内容

如果包名不一样，则会报错，根据错误提示调整。

图形用户界面

中度可信度描述已自动生成