软件构造第7次实验

实验报告

基于JavaWeb的创业网站

**班级**：软件工程14-3

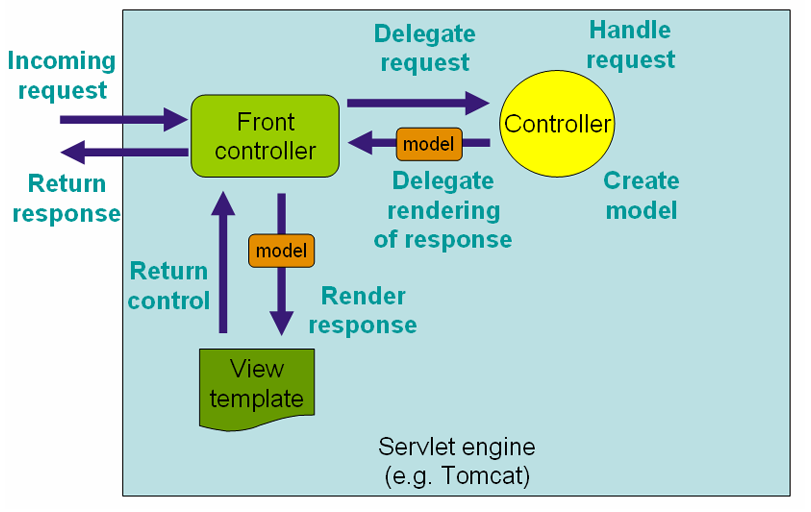
**学号**：201401061038

**姓名**：张正锟

2016年5月19日

# 设计

## 网站整体MVC架构图（Java-Spring Web MVC）





spring Web MVC是一种基于Java的实现了Web MVC设计模式的请求驱动类型的轻量级Web框架，即使用了MVC架构模式的思想，将web层进行职责解耦，基于请求驱动指的就是使用请求-响应模型，框架的目的就是帮助我们简化开发。

具体执行步骤如下：

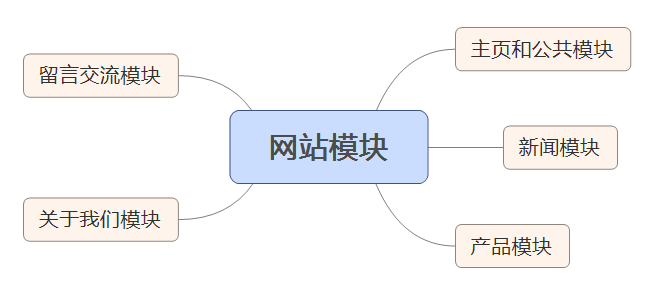
1、 首先用户发送请求————>前端控制器，前端控制器根据请求信息（如URL）来决定选择哪一个页面控制器进行处理并把请求委托给它，即以前的控制器的控制逻辑部分；

2、 页面控制器接收到请求后，进行功能处理，首先需要收集和绑定请求参数到一个对象，这个对象在Spring Web MVC中叫命令对象，并进行验证，然后将命令对象委托给业务对象进行处理；处理完毕后返回一个ModelAndView（模型数据和逻辑视图名）；

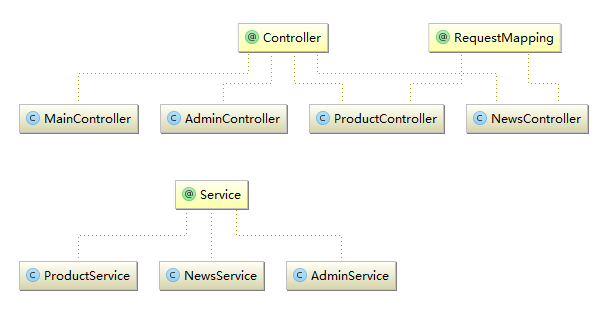
3、 前端控制器收回控制权，然后根据返回的逻辑视图名，选择相应的视图进行渲染，并把模型数据传入以便视图渲染；

4、 前端控制器再次收回控制权，将响应返回给用户。

## 2. 网站五大模块



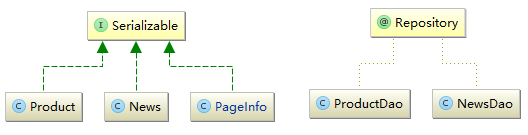
## 控制层（Controller+Service）



在程序中，Controller层负责页面映射和转向，处理网页请求

Service层负责处理业务逻辑

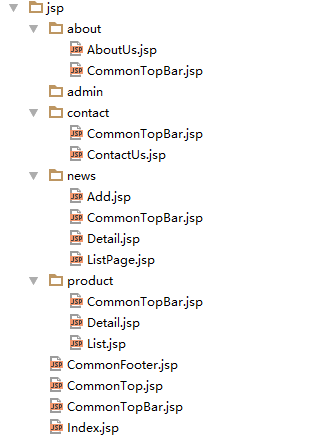
## 模型层（DAO+Model）



产品、新闻和分页信息类作为模型，定义他们的数据结构

DAO（Repository）层负责操作数据库，对数据库进行增删改查

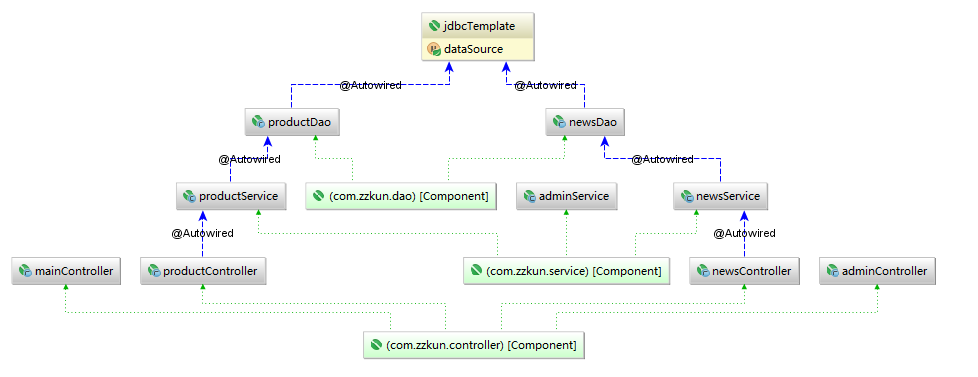
## 视图层（View、Jsp）



视图层全部使用Jsp页面，展示页面信息

JSP技术使用Java编程语言编写类XML的tags和scriptlets，来封装产生动态网页的处理逻辑。网页还能通过tags和scriptlets访问存在于服务端的资源的应用逻辑。JSP将网页逻辑与网页设计的显示分离，支持可重用的基于组件的设计，使基于Web的应用程序的开发变得迅速和容易。 JSP(JavaServer Pages)是一种动态页面技术，它的主要目的是将表示逻辑从Servlet中分离出来。

## 整体MVC结构图



以上所有部分（bean）均采用Spring框架保证单例

使用了面向切面（AOP）、依赖注入（IOC）和控制反转

默认所有的bean设置为单例模式，即对所有具有相同id的bean的请求，都返回一个共享bean实例，可以降低java创建和销毁时的系统开销。

# 实现

## Github源代码

实验报告中的代码可能不全，或者无法查看目录结构，详情可通过github查看：

<https://github.com/kun368/MyECompany>

## Maven包依赖情况pom.xml

*<?***xml version="1.0" encoding="UTF-8"***?>*<**project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"**>  
 <**modelVersion**>4.0.0</**modelVersion**>  
  
 <**groupId**>com.zzkun</**groupId**>  
 <**artifactId**>MyECompany</**artifactId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
  
 <**properties**>  
 <**maven.compiler.source**>1.8</**maven.compiler.source**>  
 <**maven.compiler.target**>1.8</**maven.compiler.target**>  
 </**properties**>  
  
  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**version**>5.1.38</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>junit</**groupId**>  
 <**artifactId**>junit</**artifactId**>  
 <**version**>4.12</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>com.mchange</**groupId**>  
 <**artifactId**>c3p0</**artifactId**>  
 <**version**>0.9.5.2</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-context</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-core</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-webmvc</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-jdbc</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.tomcat</**groupId**>  
 <**artifactId**>tomcat-catalina</**artifactId**>  
 <**version**>8.0.30</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>jstl</**groupId**>  
 <**artifactId**>jstl</**artifactId**>  
 <**version**>1.2</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.aspectj</**groupId**>  
 <**artifactId**>aspectjweaver</**artifactId**>  
 <**version**>1.8.9</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-test</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.security</**groupId**>  
 <**artifactId**>spring-security-core</**artifactId**>  
 <**version**>4.1.0.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.security</**groupId**>  
 <**artifactId**>spring-security-web</**artifactId**>  
 <**version**>4.1.0.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>spring-aspects</**artifactId**>  
 <**version**>4.2.5.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.security</**groupId**>  
 <**artifactId**>spring-security-config</**artifactId**>  
 <**version**>4.1.0.RELEASE</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>javax.validation</**groupId**>  
 <**artifactId**>validation-api</**artifactId**>  
 <**version**>1.1.0.Final</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.hibernate</**groupId**>  
 <**artifactId**>hibernate-validator</**artifactId**>  
 <**version**>5.2.4.Final</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>commons-logging</**groupId**>  
 <**artifactId**>commons-logging</**artifactId**>  
 <**version**>1.2</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>log4j</**groupId**>  
 <**artifactId**>log4j</**artifactId**>  
 <**version**>1.2.17</**version**>  
 </**dependency**>  
 </**dependencies**>  
</**project**>

## WEB项目配置主文件web.xml

*<?***xml version="1.0" encoding="UTF-8"***?>*<**web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"  
 version="3.1"**>  
  
 <**servlet**>  
 <**servlet-name**>springmvc</**servlet-name**>  
 <**servlet-class**>org.springframework.web.servlet.DispatcherServlet</**servlet-class**>  
 <**init-param**>  
 <**param-name**>contextConfigLocation</**param-name**>  
 <**param-value**>classpath:springmvc-servlet.xml</**param-value**>  
 </**init-param**>  
 <**load-on-startup**>1</**load-on-startup**>  
 </**servlet**>  
 <**servlet-mapping**>  
 <**servlet-name**>springmvc</**servlet-name**>  
 <**url-pattern**>/</**url-pattern**>  
 </**servlet-mapping**>  
  
 <**filter**>  
 <**filter-name**>CharacterEncodingFilter</**filter-name**>  
 <**filter-class**>org.springframework.web.filter.CharacterEncodingFilter</**filter-class**>  
 <**init-param**>  
 <**param-name**>encoding</**param-name**>  
 <**param-value**>utf-8</**param-value**>  
 </**init-param**>  
 </**filter**>  
 <**filter-mapping**>  
 <**filter-name**>CharacterEncodingFilter</**filter-name**>  
 <**url-pattern**>/\*</**url-pattern**>  
 </**filter-mapping**>  
  
</**web-app**>

## SpringMVC配置文件

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
 xmlns:aop="http://www.springframework.org/schema/aop"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:p="http://www.springframework.org/schema/p"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd  
 http://www.springframework.org/schema/mvc  
 http://www.springframework.org/schema/mvc/spring-mvc.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd"**>  
  
 <**context:property-placeholder location="classpath:jdbc.properties"**/>  
 <**context:component-scan base-package="com.zzkun.controller"**/>  
 <**context:component-scan base-package="com.zzkun.service"**/>  
 <**context:component-scan base-package="com.zzkun.dao"**/>  
  
 <**mvc:annotation-driven**/>  
 <**mvc:resources mapping="/assets/\*\*" location="/assets/"**/>  
  
 <**bean class="com.zzkun.config.SecurityConfig"**/>  
 <**bean class="com.zzkun.config.SecurityWebInitializer"**/>  
  
 <**bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver"**>  
 <**property name="prefix" value="/WEB-INF/jsp/"**/>  
 <**property name="suffix" value=".jsp"**/>  
 </**bean**>  
  
 <**bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"  
 destroy-method="close"**>  
 <**property name="driverClass" value="${jdbc.driverClassName}"**/>  
 <**property name="jdbcUrl" value="${jdbc.url}"**/>  
 <**property name="user" value="${jdbc.username}"**/>  
 <**property name="password" value="${jdbc.password}"**/>  
 *<!--连接池中保留的最小连接数。 -->* <**property name="minPoolSize" value="5"**/>  
 *<!--连接池中保留的最大连接数。Default: 15 -->* <**property name="maxPoolSize" value="30"**/>  
 *<!--初始化时获取的连接数，取值应在minPoolSize与maxPoolSize之间。Default: 3 -->* <**property name="initialPoolSize" value="10"**/>  
 *<!--最大空闲时间,60秒内未使用则连接被丢弃。若为0则永不丢弃。Default: 0 -->* <**property name="maxIdleTime" value="60"**/>  
 *<!--当连接池中的连接耗尽的时候c3p0一次同时获取的连接数。Default: 3 -->* <**property name="acquireIncrement" value="5"**/>  
 *<!--JDBC的标准参数，用以控制数据源内加载的PreparedStatements数量。但由于预缓存的statements 属于单个connection而不是整个连接池。所以设置这个参数需要考虑到多方面的因素。  
 如果maxStatements与maxStatementsPerConnection均为0，则缓存被关闭。Default: 0 -->* <**property name="maxStatements" value="0"**/>  
 *<!--每60秒检查所有连接池中的空闲连接。Default: 0 -->* <**property name="idleConnectionTestPeriod" value="60"**/>  
 *<!--定义在从数据库获取新连接失败后重复尝试的次数。Default: 30 -->* <**property name="acquireRetryAttempts" value="30"**/>  
 *<!--获取连接失败将会引起所有等待连接池来获取连接的线程抛出异常。但是数据源仍有效 保留，并在下次调用getConnection()的时候继续尝试获取连接。如果设为true，那么在尝试  
 获取连接失败后该数据源将申明已断开并永久关闭。Default: false -->* <**property name="breakAfterAcquireFailure" value="true"**/>  
 *<!--因性能消耗大请只在需要的时候使用它。如果设为true那么在每个connection提交的 时候都将校验其有效性。建议使用idleConnectionTestPeriod或automaticTestTable  
 等方法来提升连接测试的性能。Default: false -->* <**property name="testConnectionOnCheckout" value="false"**/>  
 </**bean**>  
  
 *<!-- 配置Jdbc模板 -->* <**bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 *<!-- 配置事务管理器 -->* <**bean id="transactionManager"  
 class="org.springframework.jdbc.datasource.DataSourceTransactionManager"  
 p:dataSource-ref="dataSource"** />  
  
 *<!-- 通过AOP配置提供事务增强，让service包下所有Bean的所有方法拥有事务 -->* <**aop:config proxy-target-class="true"**>  
 <**aop:pointcut id="serviceMethod"  
 expression=" execution(\* com.zzkun.service..\*(..))"** />  
 <**aop:advisor pointcut-ref="serviceMethod" advice-ref="txAdvice"** />  
 </**aop:config**>  
 <**tx:advice id="txAdvice" transaction-manager="transactionManager"**>  
 <**tx:attributes**>  
 <**tx:method name="\*"** />  
 </**tx:attributes**>  
 </**tx:advice**>  
</**beans**>

## News（新闻类）

**package** com.zzkun.model;  
  
**import** javax.validation.constraints.Size;  
**import** java.io.Serializable;  
**import** java.util.Date;  
  
*/\*\*  
 \* 新闻类  
 \* Created by kun on 2016/5/10.  
 \*/***public class** News **implements** Serializable {  
  
 **private int id**;  
  
 @Size(min = 3, max = 50)  
 **private** String **title**;  
  
 @Size(min = 3)  
 **private** String **content**;  
 **private** Date **addtime**;  
  
 **public** News() {  
 }  
  
 **public int** getId() {  
 **return id**;  
 }  
  
 **public void** setId(**int** id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getTitle() {  
 **return title**;  
 }  
  
 **public void** setTitle(String title) {  
 **this**.**title** = title;  
 }  
  
 **public** String getContent() {  
 **return content**;  
 }  
  
 **public void** setContent(String content) {  
 **this**.**content** = content;  
 }  
  
 **public** Date getAddtime() {  
 **return addtime**;  
 }  
  
 **public void** setAddtime(Date addtime) {  
 **this**.**addtime** = addtime;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "News{"** +  
 **"id="** + **id** +  
 **", title='"** + **title** + **'\''** +  
 **", content='"** + **content** + **'\''** +  
 **", addtime="** + **addtime** +  
 **'}'**;  
 }  
}

## Product（产品类）

**package** com.zzkun.model;  
  
**import** java.io.Serializable;  
**import** java.util.ArrayList;  
**import** java.util.Date;  
**import** java.util.List;  
  
*/\*\*  
 \* 产品类  
 \* Created by kun on 2016/5/11.  
 \*/***public class** Product **implements** Serializable {  
  
 **private int id**;  
 **private** String **name**;  
 **private double price**;  
 **private** String **intro**;  
 **private** Date **addtime**;  
 **private** String **address**;  
 **private** String **activity**;  
 **private** List<String> **imageurl**; *//产品介绍中的图片url* **private** List<String> **imagemain**; *//产品主图片* **public** Product() {  
 **imageurl** = **new** ArrayList<>();  
 **imagemain** = **new** ArrayList<>();  
 }  
  
 **public int** getId() {  
 **return id**;  
 }  
  
 **public void** setId(**int** id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 **public double** getPrice() {  
 **return price**;  
 }  
  
 **public void** setPrice(**double** price) {  
 **this**.**price** = price;  
 }  
  
 **public** String getIntro() {  
 **return intro**;  
 }  
  
 **public void** setIntro(String intro) {  
 **this**.**intro** = intro;  
 }  
  
 **public** Date getAddtime() {  
 **return addtime**;  
 }  
  
 **public void** setAddtime(Date addtime) {  
 **this**.**addtime** = addtime;  
 }  
  
 **public** String getAddress() {  
 **return address**;  
 }  
  
 **public void** setAddress(String address) {  
 **this**.**address** = address;  
 }  
  
 **public** String getActivity() {  
 **return activity**;  
 }  
  
 **public void** setActivity(String activity) {  
 **this**.**activity** = activity;  
 }  
  
 **public void** setImageurl(List<String> imageurl) {  
 **this**.**imageurl** = imageurl;  
 }  
  
 **public** List<String> getImageurl() {  
 **return imageurl**;  
 }  
  
 **public void** setImagemain(List<String> imagemain) {  
 **this**.**imagemain** = imagemain;  
 }  
  
 **public** List<String> getImagemain() {  
 **return imagemain**;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "Product{"** +  
 **"id="** + **id** +  
 **", name='"** + **name** + **'\''** +  
 **", price="** + **price** +  
 **", intro='"** + **intro** + **'\''** +  
 **", addtime="** + **addtime** +  
 **", address='"** + **address** + **'\''** +  
 **", activity='"** + **activity** + **'\''** +  
 **", imageurl="** + **imageurl** +  
 **", imagemain="** + **imagemain** +  
 **'}'**;  
 }  
}

## PageInfo（分页信息类）

**package** com.zzkun.dao;  
  
**import** java.io.Serializable;  
**import** java.util.List;  
  
*/\*\*  
 \* 页面信息，定义第0页为第一个  
 \* Created by kun on 2016/5/17.  
 \*/***public class** PageInfo<T> **implements** Serializable {  
  
 **private** List<T> **list**;  
 **private boolean isFirstPage**;  
 **private boolean isLastPage**;  
 **private int pageSum**;  
 **private int pageIndex**;  
 **private int onePageCnt**;  
  
 **public** PageInfo(List<T> list, **int** onePageCnt, **int** pageIndex, **int** pageSum) {  
 **this**.**list** = list;  
 **this**.**onePageCnt** = onePageCnt;  
 **this**.**pageIndex** = pageIndex;  
 **this**.**pageSum** = pageSum;  
 **this**.**isFirstPage** = pageIndex == 0;  
 **this**.**isLastPage** = pageIndex == pageSum-1;  
 }  
  
 **public** List<T> getList() {  
 **return list**;  
 }  
  
 **public boolean** isFirstPage() {  
 **return isFirstPage**;  
 }  
  
 **public boolean** isLastPage() {  
 **return isLastPage**;  
 }  
  
 **public int** getPageSum() {  
 **return pageSum**;  
 }  
  
 **public int** getPageIndex() {  
 **return pageIndex**;  
 }  
  
 **public int** getOnePageCnt() {  
 **return onePageCnt**;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "PageInfo{"** +  
 **"list="** + **list**.size() +  
 **", isFirstPage="** + **isFirstPage** +  
 **", isLastPage="** + **isLastPage** +  
 **", pageSum="** + **pageSum** +  
 **", pageIndex="** + **pageIndex** +  
 **", onePageCnt="** + **onePageCnt** +  
 **'}'**;  
 }  
}

## NewsDAO（新闻数据库操作）

**package** com.zzkun.dao;  
  
**import** com.zzkun.model.News;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.jdbc.core.RowMapper;  
**import** org.springframework.stereotype.Repository;  
  
**import** java.sql.ResultSet;  
**import** java.sql.SQLException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
*/\*\*  
 \* Created by kun on 2016/5/10.  
 \*/*@Repository  
**public class** NewsDao {  
  
 @Autowired  
 **private** JdbcTemplate **jdbc**;  
  
 */\*\*  
 \* 将数据中的每一行数据封装成定义的News类.  
 \*/* **private class** NewsRowMapper **implements** RowMapper<News> {  
 @Override  
 **public** News mapRow(ResultSet resultSet, **int** i) **throws** SQLException {  
 News news = **new** News();  
 news.setId(resultSet.getInt(**"id"**));  
 news.setTitle(resultSet.getString(**"title"**));  
 news.setContent(resultSet.getString(**"content"**));  
 news.setAddtime(resultSet.getTimestamp(**"addtime"**));  
 **return** news;  
 }  
 }  
  
 **public** List<News> getAllNews() {  
 String sql = **"SELECT** *\** **FROM `news` ORDER BY id DESC"**;  
 **return jdbc**.query(sql, **new** NewsRowMapper());  
 }  
  
 **public** News getNewsById(**int** id) {  
 String sql = **"SELECT** *\** **FROM `news` where id = ?"**;  
 List<News> res = **jdbc**.query(sql, **new** Object[]{id}, **new** NewsRowMapper());  
 **if**(res == **null** || res.isEmpty()) **return null**;  
 **return** res.get(0);  
 }  
  
 **public** News getLatestNews() {  
 String sql = **"SELECT** *\** **FROM `news` ORDER BY id DESC LIMIT 0, 1"**;  
 **return jdbc**.queryForObject(sql, **new** NewsRowMapper());  
 }  
  
 **public void** add(News news) {  
 String sql = **"INSERT INTO `news` (`title`, `content`, `addtime`) VALUES (?, ?, ?)"**;  
 **jdbc**.update(sql, news.getTitle(), news.getContent(), news.getAddtime());  
 }  
}

## ProductDAO（产品数据库操作）

**package** com.zzkun.dao;  
  
**import** com.zzkun.model.Product;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.jdbc.core.RowCallbackHandler;  
**import** org.springframework.jdbc.core.RowMapper;  
**import** org.springframework.stereotype.Repository;  
  
**import** java.sql.ResultSet;  
**import** java.sql.SQLException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
*/\*\*  
 \* Created by kun on 2016/5/11.  
 \*/*@Repository  
**public class** ProductDao {  
  
 @Autowired  
 **private** JdbcTemplate **jdbc**;  
  
 **private class** ProductRowMapper **implements** RowMapper<Product> {  
 @Override  
 **public** Product mapRow(ResultSet resultSet, **int** i) **throws** SQLException {  
 Product product = **new** Product();  
 product.setId(resultSet.getInt(**"id"**));  
 product.setName(resultSet.getString(**"name"**));  
 product.setIntro(resultSet.getString(**"intro"**));  
 product.setAddtime(resultSet.getTimestamp(**"addtime"**));  
 product.setPrice(resultSet.getDouble(**"price"**));  
 product.setAddress(resultSet.getString(**"address"**));  
 product.setActivity(resultSet.getString(**"activity"**));  
 **return** product;  
 }  
 }  
  
 **public void** setImageUrlList(Product product) {  
 product.getImageurl().clear();  
 product.getImagemain().clear();  
 String sql = **"SELECT** *\** **FROM product\_img WHERE product\_img.productid = ?"**;  
 **jdbc**.query(sql, **new** Object[]{product.getId()}, **new** RowCallbackHandler() {  
 @Override  
 **public void** processRow(ResultSet resultSet) **throws** SQLException {  
 **int** type = resultSet.getInt(**"type"**);  
 **if**(type == 1) product.getImageurl().add(resultSet.getString(**"imageurl"**));  
 **else if**(type == 2) product.getImagemain().add(resultSet.getString(**"imageurl"**));  
 }  
 });  
 }  
  
 **public** List<Product> getAllProduct() {  
 String sql = **"SELECT** *\** **FROM product ORDER BY product.addtime DESC"**;  
 List<Product> list = **jdbc**.query(sql, **new** ProductRowMapper());  
 **if**(list == **null** || list.isEmpty()) **return null**;  
 **for**(Product product : list)  
 setImageUrlList(product);  
 **return** list;  
 }  
  
 **public** Product getProductById(**int** id) {  
 String sql = **"SELECT** *\** **FROM product WHERE product.id = ?"**;  
 Product product = **jdbc**.queryForObject(sql, **new** Object[]{id}, **new** ProductRowMapper());  
 setImageUrlList(product);  
 **return** product;  
 }  
}

## NewsService（新闻服务类）

**package** com.zzkun.service;  
  
**import** com.zzkun.dao.NewsDao;  
**import** com.zzkun.dao.PageInfo;  
**import** com.zzkun.model.News;  
**import** com.zzkun.util.MyPaging;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Service;  
  
**import** java.util.Date;  
**import** java.util.List;  
  
*/\*\*  
 \* Created by kun on 2016/5/10.  
 \*/*@Service  
**public class** NewsService {  
  
 @Autowired  
 **private** NewsDao **newsDao**;  
  
 **private** MyPaging<News> **myPaging** = **null**;  
 **private boolean newsChanged** = **false**;  
  
 **public** List<News> getAllNews() {  
 **return newsDao**.getAllNews();  
 }  
  
 **public void** saveNews(News news) {  
 news.setAddtime(**new** Date());  
 **newsDao**.add(news);  
 **newsChanged** = **true**;  
 }  
  
 **public int** getLatestNewsId() {  
 News news = **newsDao**.getLatestNews();  
 **if**(news == **null**) **return** 0;  
 **return** news.getId();  
 }  
  
 **public** News getNews(**int** id) {  
 **return newsDao**.getNewsById(id);  
 }  
  
 **public** PageInfo<News> getPage(**int** id) {  
 **if**(**myPaging** == **null** || **newsChanged**) {  
 **myPaging** = **new** MyPaging<>(getAllNews(), 10);  
 **newsChanged** = **false**;  
 }  
 **return myPaging**.getPage(id);  
 }  
}

## ProductService（产品服务类）

**package** com.zzkun.service;  
  
**import** com.zzkun.dao.ProductDao;  
**import** com.zzkun.model.Product;  
**import** org.apache.commons.logging.Log;  
**import** org.apache.commons.logging.LogFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Service;  
  
**import** java.util.List;  
  
*/\*\*  
 \* Created by kun on 2016/5/14.  
 \*/*@Service  
**public class** ProductService {  
  
 @Autowired  
 **private** ProductDao **productDao**;  
  
 **private static** Log *logger* = LogFactory.*getLog*(ProductService.**class**);  
  
 **public** List<Product> getAllProduct() {  
 List<Product> products = **productDao**.getAllProduct();  
 *logger*.info(**"geekun"** + products);  
 **return** products;  
 }  
  
 **public** Product getProductById(**int** id) {  
 Product product = **productDao**.getProductById(id);  
 *logger*.info(**"geekun"** + product);  
 **return** product;  
 }  
}

## MainController（主控制器）

**package** com.zzkun.controller;  
  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.servlet.ModelAndView;  
  
*/\*\*  
 \* Created by kun on 2016/5/8.  
 \*/*@Controller  
**public class** MainController {  
  
 @RequestMapping(value = {**"/"**, **"index"**})  
 **public** ModelAndView index() {  
 **return new** ModelAndView(**"Index"**);  
 }  
  
 @RequestMapping(**"/about"**)  
 **public** String about() {  
 **return "about/AboutUs"**;  
 }  
  
 @RequestMapping(**"contactus"**)  
 **public** String contactus() {  
 **return "contact/ContactUs"**;  
 }  
}

## NewsController（新闻模块控制器）

**package** com.zzkun.controller;  
  
**import** com.zzkun.dao.PageInfo;  
**import** com.zzkun.model.News;  
**import** com.zzkun.service.NewsService;  
**import** com.zzkun.util.MyPaging;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.validation.BindingResult;  
**import** org.springframework.validation.FieldError;  
**import** org.springframework.web.bind.annotation.ModelAttribute;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.servlet.ModelAndView;  
  
**import** javax.validation.Valid;  
  
*/\*\*  
 \* Created by kun on 2016/5/10.  
 \*/*@Controller  
@RequestMapping(**"/news"**)  
**public class** NewsController {  
  
 @Autowired  
 **private** NewsService **newsService**;  
  
 @RequestMapping(**"/list/page/{pageid}"**)  
 **public** ModelAndView listPage(@PathVariable **int** pageid) {  
 **return new** ModelAndView(**"news/ListPage"**, **"curpage"**, **newsService**.getPage(pageid-1));  
 }  
  
 @RequestMapping(**"/add"**)  
 **public** ModelAndView add() {  
 **return new** ModelAndView(**"news/Add"**, **"news"**, **new** News());  
 }  
  
 @RequestMapping(**"/save"**)  
 **public** ModelAndView save(@Valid @ModelAttribute News news, BindingResult bindingResult) {  
 **if**(bindingResult.hasErrors()) {  
 **return new** ModelAndView(**"news/Add"**);  
 }  
 **newsService**.saveNews(news);  
 **return new** ModelAndView(**"redirect:/news/detail/"** + **newsService**.getLatestNewsId());  
 }  
  
 @RequestMapping(**"/detail/{id}"**)  
 **public** ModelAndView detail(@PathVariable **int** id) {  
 **return new** ModelAndView(**"news/Detail"**, **"news"**, **newsService**.getNews(id));  
 }  
}

## ProductController（产品模块控制器类）

**package** com.zzkun.controller;  
  
**import** com.zzkun.service.ProductService;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.servlet.ModelAndView;  
  
*/\*\*  
 \* Created by kun on 2016/5/14.  
 \*/*@Controller  
@RequestMapping(**"/product"**)  
**public class** ProductController {  
  
 @Autowired  
 **private** ProductService **service**;  
  
 @RequestMapping(**"/list"**)  
 **public** ModelAndView list() {  
 **return new** ModelAndView(**"product/List"**, **"products"**, **service**.getAllProduct());  
 }  
  
 @RequestMapping(**"/detail/{id}"**)  
 **public** ModelAndView detail(@PathVariable **int** id) {  
 **return new** ModelAndView(**"product/Detail"**, **"product"**, **service**.getProductById(id));  
 }  
}

## MyPaging（分页工具类）

**package** com.zzkun.util;  
  
**import** com.zzkun.dao.PageInfo;  
**import** org.apache.commons.logging.Log;  
**import** org.apache.commons.logging.LogFactory;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
*/\*\*  
 \* 分页工具  
 \* Created by kun on 2016/5/17.  
 \*/***public class** MyPaging<T> {  
  
 **private static** Log *logger* = LogFactory.*getLog*(MyPaging.**class**);  
  
 **private** List<T> **all**;  
 **private int onePageCnt**;  
  
 **public** MyPaging(List<T> all, **int** onePageCnt) {  
 **this**.**all** = all;  
 **this**.**onePageCnt** = onePageCnt;  
 }  
  
 */\*\*  
 \* 得到第i页的内容  
 \** ***@param index*** *页面编号，从0开始  
 \** ***@return*** *第i页的内容  
 \*/* **public** PageInfo<T> getPage(**int** index) {  
 **int** pageSum = ((**all**.size()-1)/**onePageCnt**) + 1;  
 **if**(index < 0 || index >= pageSum)  
 **return null**;  
 *logger*.info(**"pagesum:"** + pageSum + **", index:"** + index);  
 List<T> list = **new** ArrayList<>();  
 **for**(**int** i = index \* **onePageCnt**; i < **all**.size() && i < (index + 1) \* **onePageCnt**; ++i)  
 list.add(**all**.get(i));  
 PageInfo<T> result = **new** PageInfo<>(list, **onePageCnt**, index, pageSum);  
 *logger*.info(**"getPage:"** + result);  
 **return** result;  
 }  
}

# 简单测试