**想了解下spring-boot**

**然后说动手就动手**

**工具是idea**

**框架是spring boot +mybatis + mysql + pagehepler分页插件**

**Easyui+layer弹窗**

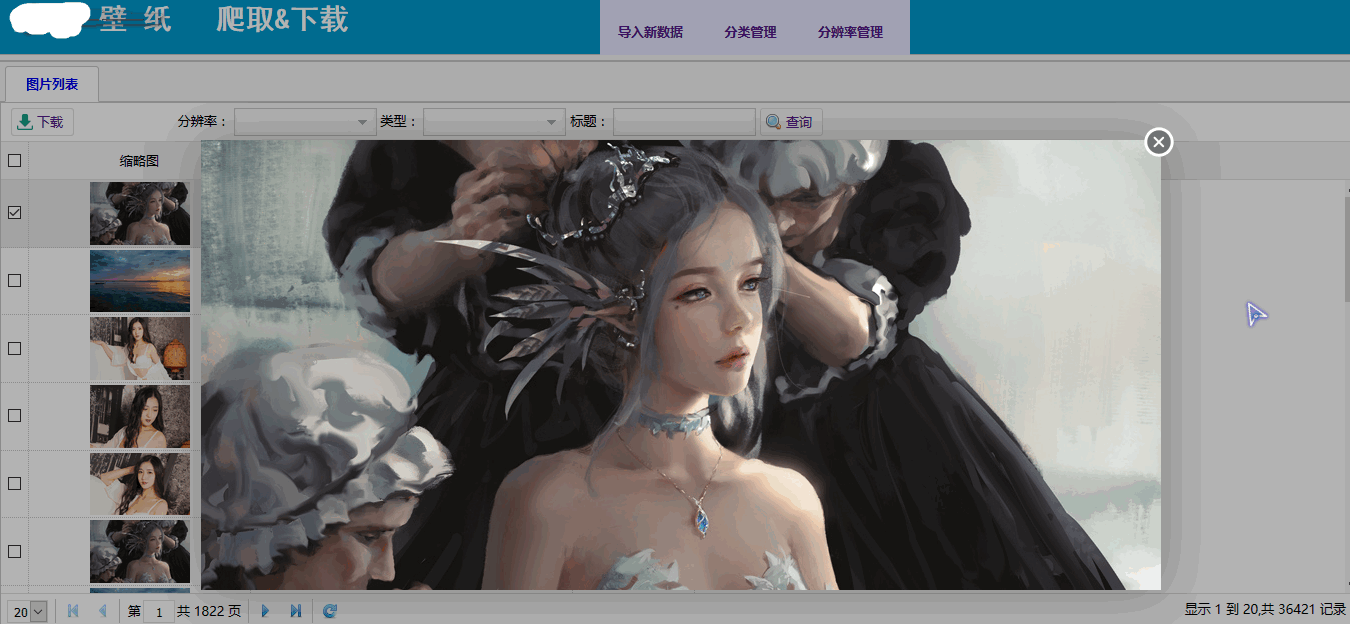
**页面用的是jsp**

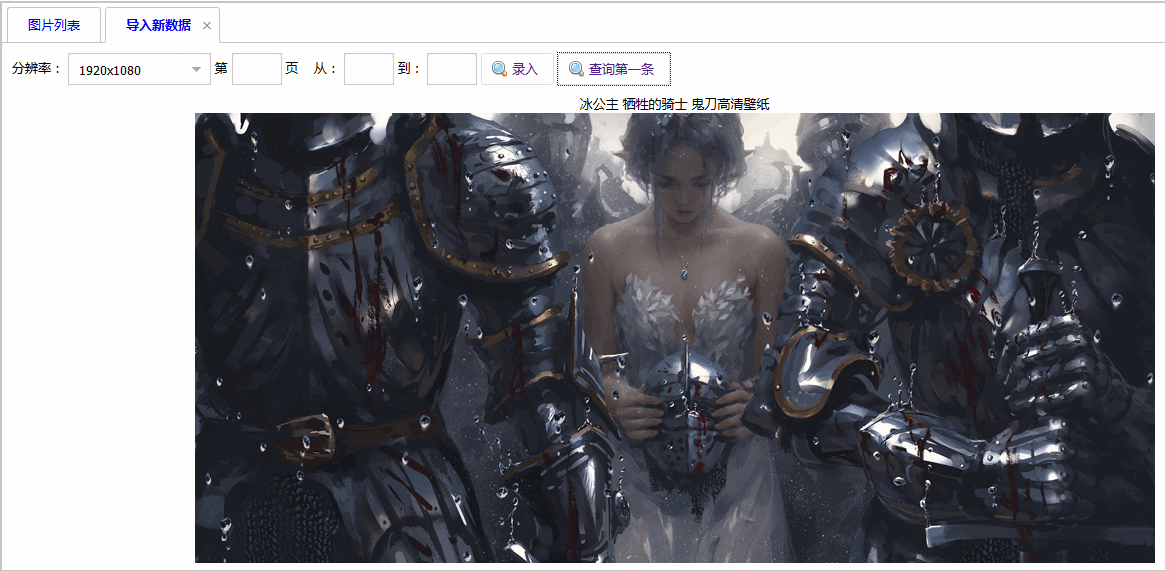
**不过官方不推荐使用，但是我还是想用，以后再改用其他的模板**

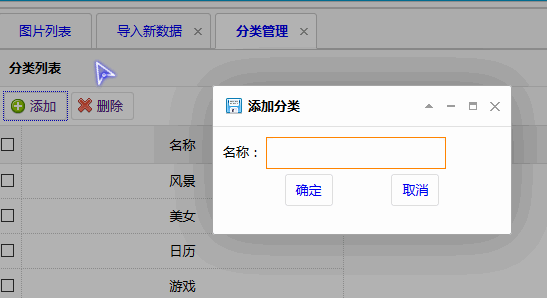
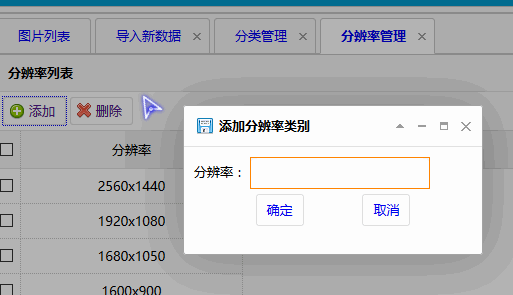
**页面比较简单：**

**本人对前端了解甚少，还是希望大家不要见笑吧，还是先看页面吧**

**我也没有那什么GIt账号啥的，不知道这个可以压缩上传不**

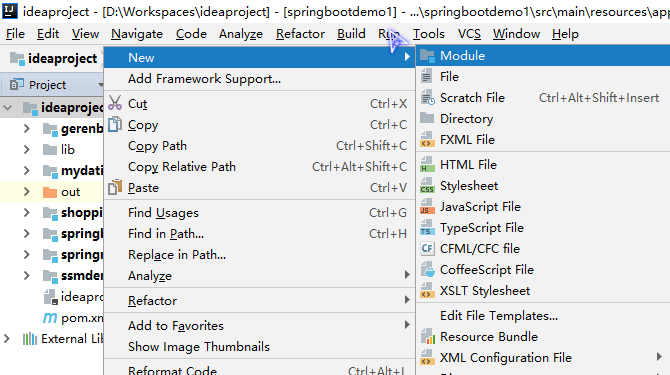


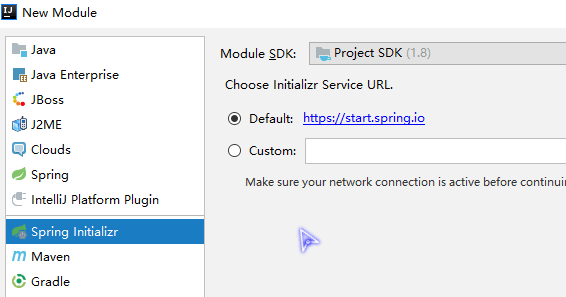


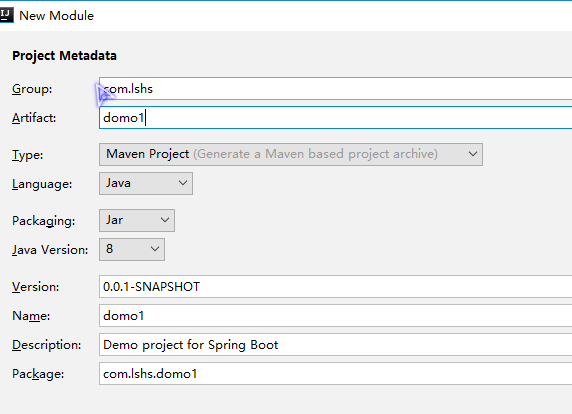
 

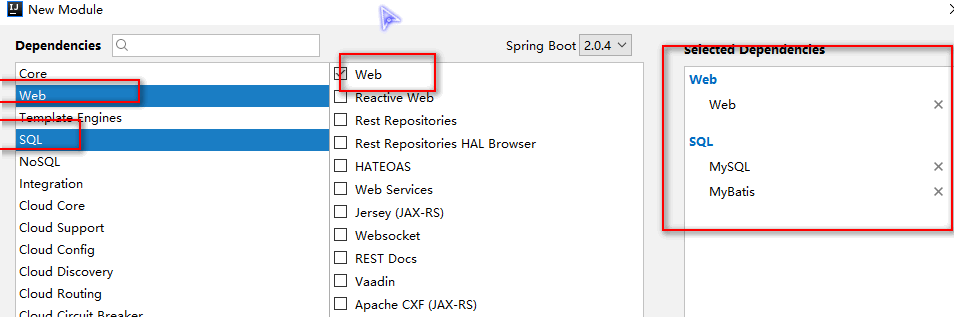


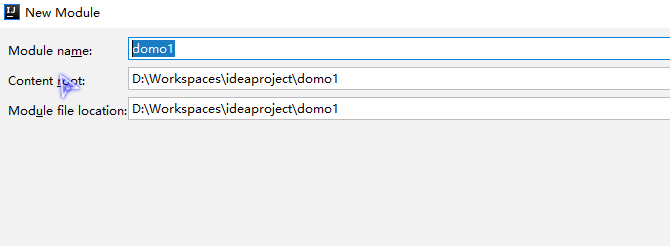
**下面就是开始新建项目了，开始就是真功夫了，**

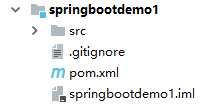




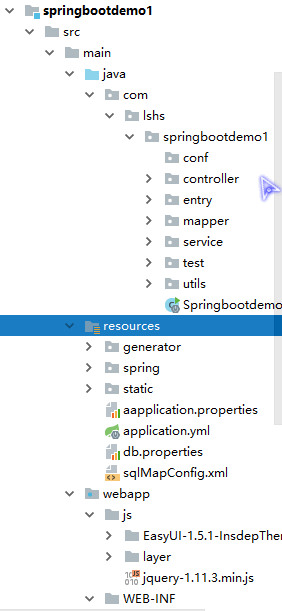








**刚刚建立完成就是这样子，不在这个图上面的文件，可以删除掉**



**那个application那个是主类**

**其他的包或者是什么要放在和这个类同一目录或者是主类的子目录**

**另外这个webapp是我自己建立的，然后设置了下根目录，**

**如果右键没有新建jsp的菜单，就新增一个web容器**

**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.lshs</**groupId**>  
 <**artifactId**>springbootdemo1</**artifactId**>  
 <**version**>0.0.1-SNAPSHOT</**version**>  
 <**packaging**>war</**packaging**>  
  
 <**name**>springbootdemo1</**name**>

**总感觉版本冲突比较厉害**

**如果你是想使用pageheloper mybatis的分页插件 就把版本改成1.5.多的就行，**

<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>1.5.8.RELEASE</**version**>  
 *<!--<version>1.4.7.RELEASE</version>-->  
 <!--<version>2.0.4.RELEASE</version>-->* <**relativePath**/> *<!-- lookup parent from repository -->* </**parent**>  
  
 <**properties**>  
 <**project.build.sourceEncoding**>UTF-8</**project.build.sourceEncoding**>  
 <**project.reporting.outputEncoding**>UTF-8</**project.reporting.outputEncoding**>  
 <**java.version**>1.8</**java.version**>  
 </**properties**>  
  
 <**dependencies**>  
  
 <**dependency**>  
 <**groupId**>org.mybatis.spring.boot</**groupId**>  
 <**artifactId**>mybatis-spring-boot-starter</**artifactId**>  
 <**version**>1.0.1</**version**>  
 *<!--<version>1.3.1</version>-->* </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
**jsp依赖的jar**  
 <**dependency**>  
 <**groupId**>org.apache.tomcat.embed</**groupId**>  
 <**artifactId**>tomcat-embed-jasper</**artifactId**>  
 *<!--<scope>runtime</scope>-->* </**dependency**>  
  
 *<!--<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-thymeleaf</artifactId>  
 </dependency>-->* <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-test</**artifactId**>  
 *<!--<scope>test</scope>-->* </**dependency**>  
  
 *<!---->* <**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**version**>5.1.45</**version**>  
 </**dependency**>  
  
 <**dependency**>  
 <**groupId**>com.github.pagehelper</**groupId**>  
 <**artifactId**>pagehelper</**artifactId**>  
 <**version**>5.1.0-beta2</**version**>  
 </**dependency**>  
  
 *<!-- springboot对mybatis的整合支持还是比较苛刻的，版本不兼容问题比较严重，pageHelper分页插件也是一样，不是最新的版本就没有问题，有时需要降低版本进行支持，这里的版本是调试通过可以用的版本，其他应该还有可以匹配的版本，但是没有去测试 -->  
 <!-- spring boot pagehelper autoconfigure依赖 版本有兼容性问题 第一种方式依赖的包-->* <**dependency**>  
 <**groupId**>com.github.pagehelper</**groupId**>  
 <**artifactId**>pagehelper-spring-boot-autoconfigure</**artifactId**>  
 <**version**>1.1.1</**version**>  
 </**dependency**>  
 *<!-- spring boot 整合pagehelper依赖 版本有兼容性问题 第一种方式依赖的包-->* <**dependency**>  
 <**groupId**>com.github.pagehelper</**groupId**>  
 <**artifactId**>pagehelper-spring-boot-starter</**artifactId**>  
 <**version**>1.2.3</**version**>  
 </**dependency**>  
  
 <**dependency**>  
 <**groupId**>org.apache.httpcomponents</**groupId**>  
 <**artifactId**>httpclient</**artifactId**>  
 <**version**>4.5.5</**version**>  
 </**dependency**>  
 *<!--js加载-->  
 <!--<dependency>  
 <groupId>net.sourceforge.htmlunit</groupId>  
 <artifactId>htmlunit</artifactId>  
 <version>2.13</version>  
 </dependency>-->* <**dependency**>  
 <**groupId**>org.jsoup</**groupId**>  
 <**artifactId**>jsoup</**artifactId**>  
 <**version**>1.7.2</**version**>  
 </**dependency**>

**阿里巴巴连接池**  
 <**dependency**>  
 <**groupId**>com.alibaba</**groupId**>  
 <**artifactId**>druid-spring-boot-starter</**artifactId**>  
 <**version**>1.1.0</**version**>  
 </**dependency**>  
  
 *<!--自动编译-->* <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-devtools</**artifactId**>  
 <**optional**>true</**optional**>  
 </**dependency**>  
  
   
 </**dependencies**>  
  
 <**build**>  
 <**plugins**>  
 <**plugin**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-maven-plugin</**artifactId**>  
 <**dependencies**>  
 *<!-- spring热部署 -->* <**dependency**>  
 <**groupId**>org.springframework</**groupId**>  
 <**artifactId**>springloaded</**artifactId**>  
 <**version**>1.2.6.RELEASE</**version**>  
 </**dependency**>  
 </**dependencies**>  
 *<!--<configuration>  
 <mainClass>com.lshs.springbootdemo1.Springbootdemo1Application</mainClass>  
 </configuration>-->*</**plugin**>

*<!-- mybatis generator 自动生成代码插件 -->  
 <!--<plugin>  
 <groupId>org.mybatis.generator</groupId>  
 <artifactId>mybatis-generator-maven-plugin</artifactId>  
 <version>1.3.2</version>  
 <configuration>  
 <configurationFile>${basedir}/src/main/resources/generator/generatorConfig.xml</configurationFile>  
 <overwrite>true</overwrite>  
 <verbose>true</verbose>  
 </configuration>  
 </plugin>-->* </**plugins**>  
 *<!--编译的时候xml没有编译进去-->* <**resources**>  
 <**resource**>  
 <**directory**>src/main/java</**directory**>  
 <**includes**>  
 <**include**>\*\*/\*.xml</**include**>  
 </**includes**>  
 <**filtering**>true</**filtering**>  
 </**resource**>  
 <**resource**>  
 <**directory**>src/main/resources</**directory**>  
 <**includes**>  
 <**include**>\*\*/\*.xml</**include**>  
 <**include**>\*\*/\*.properties</**include**>  
 <**include**>\*\*/\*.yml</**include**>  
 </**includes**>  
 <**filtering**>true</**filtering**>  
 </**resource**>  
 </**resources**>  
 </**build**>  
  
  
</**project**>

generatorConfig.xml

*<?***xml version="1.0" encoding="UTF-8"***?>***<!DOCTYPE generatorConfiguration  
 PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"  
 "http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd"*>***<**generatorConfiguration**>  
 *<!-- 数据库驱动:选择你的本地硬盘上面的数据库驱动包-->* <**classPathEntry location="D:\Workspaces\dependfile\mysql-connector-java-5.1.8-bin.jar"**/>  
 <**context id="DB2Tables" targetRuntime="MyBatis3"**>  
 <**commentGenerator**>  
 <**property name="suppressDate" value="true"**/>  
 *<!-- 是否去除自动生成的注释 true：是 ： false:否 -->* <**property name="suppressAllComments" value="true"**/>  
 </**commentGenerator**>  
 *<!--数据库链接URL，用户名、密码 -->* <**jdbcConnection driverClass="com.mysql.jdbc.Driver" connectionURL="jdbc:mysql://127.0.0.1/bian" userId="root" password="ls0611"**>  
 </**jdbcConnection**>  
 <**javaTypeResolver**>  
 <**property name="forceBigDecimals" value="false"**/>  
 </**javaTypeResolver**>  
 *<!-- 生成模型的包名和位置-->* <**javaModelGenerator targetPackage="com.lshs.springbootdemo1.entry" targetProject="src/main/java"**>  
 <**property name="enableSubPackages" value="true"**/>  
 <**property name="trimStrings" value="true"**/>  
 </**javaModelGenerator**>  
 *<!-- 生成映射文件的包名和位置-->* <**sqlMapGenerator targetPackage="com.lshs.springbootdemo1.mapper" targetProject="src/main/java"**>  
 <**property name="enableSubPackages" value="true"**/>  
 </**sqlMapGenerator**>  
 *<!-- 生成DAO的包名和位置-->* <**javaClientGenerator type="XMLMAPPER" targetPackage="com.lshs.springbootdemo1.mapper" targetProject="src/main/java"**>  
 <**property name="enableSubPackages" value="true"**/>  
 </**javaClientGenerator**>  
 *<!-- 要生成的表 tableName是数据库中的表名或视图名 domainObjectName是实体类名-->* <**table tableName="bian" domainObjectName="BiAn" enableCountByExample="false" enableUpdateByExample="false" enableDeleteByExample="false" enableSelectByExample="false" selectByExampleQueryId="false"**></**table**>  
 <**table tableName="category" domainObjectName="Category" enableCountByExample="false" enableUpdateByExample="false" enableDeleteByExample="false" enableSelectByExample="false" selectByExampleQueryId="false"**></**table**>  
 <**table tableName="screens" domainObjectName="Screens" enableCountByExample="false" enableUpdateByExample="false" enableDeleteByExample="false" enableSelectByExample="false" selectByExampleQueryId="false"**></**table**>  
 </**context**>  
</**generatorConfiguration**>

application.yml

**server:  
 port:** 80  
 **context-path:** /  
  
**spring:  
 datasource:  
 driver-class-name:** com.mysql.jdbc.Driver  
 **url:** jdbc:mysql://localhost:3306/bian  
 **username:** root  
 **password:** ls0611  
 *###################以下为druid增加的配置###########################* **type:** com.alibaba.druid.pool.DruidDataSource  
 *# 下面为连接池的补充设置，应用到上面所有数据源中  
 # 初始化大小，最小，最大* **initialSize:** 5  
 **minIdle:** 5  
 **maxActive:** 20  
 *# 配置获取连接等待超时的时间* **maxWait:** 60000  
 *# 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒* **timeBetweenEvictionRunsMillis:** 60000  
 *# 配置一个连接在池中最小生存的时间，单位是毫秒* **minEvictableIdleTimeMillis:** 300000  
 **validationQuery:** SELECT 1 FROM DUAL  
 **testWhileIdle:** true  
 **testOnBorrow:** false  
 **testOnReturn:** false  
 *# 打开PSCache，并且指定每个连接上PSCache的大小* **poolPreparedStatements:** true  
 **maxPoolPreparedStatementPerConnectionSize:** 20  
 *# 配置监控统计拦截的filters，去掉后监控界面sql无法统计，'wall'用于防火墙，此处是filter修改的地方* **filters:  
 commons-log.connection-logger-name:** stat,wall,log4j  
 *# 通过connectProperties属性来打开mergeSql功能；慢SQL记录* **connectionProperties:** druid.stat.mergeSql=true;druid.stat.slowSqlMillis=5000  
 *# 合并多个DruidDataSource的监控数据* **useGlobalDataSourceStat:** true  
 **mvc:  
 static-path-pattern:** /js/\*\*  
*# static-path-pattern: /static/js/\*\** **view:  
 prefix:** /WEB-INF/jsps/  
 **suffix:** .jsp  
 **resources:  
 static-locations:** /js/  
  
**mybatis:  
 mapper-locations:** classpath\*:com/lshs/springbootdemo1/mapper/\*Mapper.xml  
*#pagehelper分页插件***pagehelper:  
 helperDialect:** mysql  
 **reasonable:** true  
 **support-methods-arguments:** true  
 **params:** count=countSql  
*# dialect: mysql  
# type-aliases-package: com.lshs.springbootdemo1.entry*

Springbootdemo1Application.java

**package** com.lshs.springbootdemo1;  
  
**import** org.mybatis.spring.annotation.MapperScan;  
**import** org.springframework.boot.SpringApplication;  
**import** org.springframework.boot.autoconfigure.SpringBootApplication;  
**import** org.springframework.boot.builder.SpringApplicationBuilder;  
**import** org.springframework.boot.context.properties.ConfigurationProperties;  
**import** org.springframework.boot.web.support.SpringBootServletInitializer;  
**import** org.springframework.context.annotation.Configuration;  
  
  
*//@EnableAutoConfiguration*

**这个注解就是扫描Mapper.java文件的**@MapperScan(**"com.lshs.springbootdemo1.mapper"**)  
@SpringBootApplication  
**public class** Springbootdemo1Application {  
  
 **public static void** main(String[] args) {  
 SpringApplication.*run*(Springbootdemo1Application.**class**, args);  
 }  
}

JsoupUtil.java

**package** com.lshs.springbootdemo1.utils;  
  
**import** com.lshs.springbootdemo1.entry.BiAn;  
**import** org.apache.http.HttpResponse;  
**import** org.apache.http.client.methods.HttpGet;  
**import** org.apache.http.impl.client.DefaultHttpClient;  
**import** org.jsoup.Jsoup;  
**import** org.jsoup.nodes.Document;  
**import** org.jsoup.nodes.Element;  
**import** org.jsoup.select.Elements;  
  
**import** java.io.IOException;  
**import** java.net.URI;  
**import** java.net.URISyntaxException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
*/\*\*  
 \** ***@Description:*** *爬取网页工具  
 \** ***@author:*** *LuShao  
 \** ***@create:*** *2018-08-26 8:30  
 \*\*/***public class** JsoupUtil {  
  
 */\*\*  
 \* 得到url响应状态  
 \** ***@param url*** *\** ***@return*** *\** ***@throws*** *URISyntaxException  
 \** ***@throws*** *IOException  
 \*/* **public static int** getStatus(String url) **throws** URISyntaxException, IOException {  
 HttpGet get=**new** HttpGet(**new** URI(url));  
 HttpResponse response=**new** DefaultHttpClient().execute(get);  
 **int** statusCode = response.getStatusLine().getStatusCode();  
 **return** statusCode;  
 }  
  
 */\*\*  
 \* 得到第一个页面的list内容  
 \** ***@param url1*** *\** ***@param biAn*** *\** ***@param start*** *从这个页面的第几个开始  
 \** ***@param end*** *到第几个结束 1-18  
 \** ***@return*** *\** ***@throws*** *IOException  
 \*/* **public static** List<BiAn> getUrl1List(String url1, BiAn biAn,Integer start,Integer end) **throws** IOException {  
 List<BiAn> url1s=**new** ArrayList<>();  
 Document document = Jsoup.*connect*(url1).timeout(300000).get();  
 Element list = document.getElementsByClass(**"list"**).get(0);  
 Elements lis = list.getElementsByTag(**"li"**);  
  
 **int** num=0;  
 **int** size;  
 **if**(start!=**null**) num=start-1;  
 **if** (end!=**null**) size=end;  
 **else** size=lis.size();  
  
 **for** (**int** i=num;i<size;i++){  
 Element li = lis.get(i);  
 Element a = li.getElementsByTag(**"a"**).get(0);  
 String href1 = a.attr(**"href"**);  
 **if** (!href1.equals(**"http://pic.netbian.com/"**)){  
 BiAn biAn1=**new** BiAn();  
 biAn1.setPage(biAn.getPage());  
 biAn1.setScreen(biAn.getScreen());  
 biAn1.setFromurl(biAn.getFromurl());  
 String href = **"http://www.netbian.com"** + href1;  
 biAn1.setUrl1(href);  
 String title = a.attr(**"title"**);  
 biAn1.setTitle(title);  
 Element img = a.getElementsByTag(**"img"**).get(0);  
 String src = img.attr(**"src"**);  
 biAn1.setImg1(src);  
 Element p = li.getElementsByTag(**"p"**).get(0);  
*// 更新时间：2017-01-07* String replace = p.text().replace(**"更新时间："**, **""**);  
 biAn1.setUpdatetime(replace);  
 url1s.add(biAn1);  
 }  
 }  
 **return** url1s;  
 }  
  
 */\*\*  
 \* 根据传入的url得到第二个页面内容  
 \** ***@param url1*** *\** ***@param biAn*** *\** ***@return*** *\** ***@throws*** *IOException  
 \*/* **public static** BiAn getUrl2(String url1,BiAn biAn) **throws** IOException {  
 Document document = Jsoup.*connect*(url1).timeout(300000).get();  
 Element action = document.getElementsByClass(**"action"**).get(0);  
*// 风景壁纸* String s = action.getElementsByTag(**"a"**).get(1).text().replaceAll(**"壁纸"**, **""**);  
 biAn.setCategory(s);  
 Element pic = document.getElementsByClass(**"pic"**).get(0);  
 Element a = pic.getElementsByTag(**"a"**).get(0);  
 String href = a.attr(**"href"**);  
 href=**"http://www.netbian.com"**+href;  
 biAn.setUrl2(href);  
 Element img = a.getElementsByTag(**"img"**).get(0);  
 String src = img.attr(**"src"**);  
 biAn.setImg2(src);  
 **return** biAn;  
 }  
  
 */\*\*  
 \* 得到最后一页的内容  
 \** ***@param url2*** *\** ***@param biAn*** *\** ***@return*** *\** ***@throws*** *IOException  
 \*/* **public static** BiAn getUrl3(String url2,BiAn biAn) **throws** IOException {  
 Document document = Jsoup.*connect*(url2).timeout(300000).get();  
 String attr = document.getElementById(**"endimg"**).getElementsByTag(**"a"**).get(0).attr(**"href"**);  
 biAn.setUrl3(attr);  
 **return** biAn;  
 }  
  
  
}

**Controller.java**

**这里要说一下**@RestController  
**这个注解就是把@Controller和@ResponseBody合在一起了**

**也就是说如果这个类都是要返回json数据的话 只要在类上面注解这个就行**

**package** com.lshs.springbootdemo1.controller;  
  
**import** com.lshs.springbootdemo1.entry.BiAn;  
**import** com.lshs.springbootdemo1.entry.Contains;  
**import** com.lshs.springbootdemo1.entry.EUDataGridResult;  
**import** com.lshs.springbootdemo1.entry.QueryEntry;  
**import** com.lshs.springbootdemo1.service.BianService;  
**import** com.lshs.springbootdemo1.utils.JsoupUtil;  
**import** org.jsoup.Jsoup;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.bind.annotation.RestController;  
  
**import** java.io.\*;  
**import** java.net.URISyntaxException;  
**import** java.net.URL;  
**import** java.net.URLConnection;  
**import** java.util.Date;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
  
*/\*\*  
 \** ***@Description:*** *彼岸图  
 \** ***@author:*** *LuShao  
 \** ***@create:*** *2018-08-23 17:09  
 \*\*/*@RestController  
@RequestMapping(**"/main/"**)  
**public class** FigureCtr {  
  
 @Autowired  
 **private** BianService **bianService**;  
  
 @RequestMapping(**"all"**)  
 **public** EUDataGridResult getCategoryQuery(QueryEntry queryEntry){  
 EUDataGridResult imgs = **bianService**.findImgs(queryEntry);  
 **return** imgs;  
 }  
  
 @RequestMapping(**"last"**)  
 **public** BiAn lastData(String spx){  
 BiAn biAn = **bianService**.lastData(spx);  
 **return** biAn;  
 }  
  
 @RequestMapping(**"first"**)  
 **public** BiAn firstData(String spx){  
 BiAn biAn = **bianService**.firstData(spx);  
 **return** biAn;  
 }  
  
 @RequestMapping(**"download"**)  
 **public** Map<String,Object> download(String[] ids){  
 Map<String,Object> map=**new** HashMap<>();  
 **try** {  
 String format = Contains.***SDF1***.format(**new** Date());  
 File file=**new** File(Contains.***DIR***+format);  
 **if**(!file.exists()) file.mkdirs();  
 **for** (String img : ids) {  
 URLConnection conn=**new** URL(img).openConnection();  
 InputStream is = conn.getInputStream();  
 String imgname = img.substring(img.lastIndexOf(**"/"**) + 1);  
 BufferedOutputStream bos=**new** BufferedOutputStream(**new** FileOutputStream(file+File.***separator***+imgname));  
 **byte**[] bytes=**new byte**[1024\*1024];  
 **int** size;  
 **while** (-1!=(size=is.read(bytes))){  
 bos.write(bytes,0,size);  
 }  
 bos.close();  
 is.close();  
 }  
 map.put(**"status"**,200);  
 }**catch** (Exception e){  
 map.put(**"status"**,500);  
 e.printStackTrace();  
 }  
 **return** map;  
 }  
  
 @RequestMapping(**"add"**)  
 **public** Map<String,Object> add(QueryEntry queryEntry){  
 Map<String,Object> map=**new** HashMap<>();  
 String spx=queryEntry.getScreen();  
 Integer page = queryEntry.getPage();  
 String url=**"http://www.netbian.com/"**+spx+**"/"**+**"index\_"** + page +**".htm"**;  
 **try** {  
 **int** status = JsoupUtil.*getStatus*(url);  
 **if**(status==200){  
 BiAn biAn=**new** BiAn();  
 biAn.setPage(page);  
 biAn.setFromurl(url);  
 biAn.setScreen(spx);  
 List<BiAn> url1s = JsoupUtil.*getUrl1List*(url, biAn,queryEntry.getStart(),queryEntry.getEnd());  
 **int** size=0;  
 **for** (BiAn biAn1 : url1s) {  
 String url1 = biAn1.getUrl1();  
 **if** (url1.contains(**".htm"**)){  
 **int** status1 = JsoupUtil.*getStatus*(url1);  
 **if**(status1==200){  
 BiAn biAn2 = JsoupUtil.*getUrl2*(url1, biAn1);  
 String url2 = biAn2.getUrl2();  
 **if** (url2.contains(**".htm"**)){  
 **int** status2 = JsoupUtil.*getStatus*(url2);  
 **if** (status2==200){  
 BiAn biAn3 = JsoupUtil.*getUrl3*(url2, biAn2);  
 **int** i1 = **bianService**.saveBian(biAn3);  
 size+=i1;  
 }  
 }  
 }  
 }  
 }  
 **if** (size>0) map.put(**"status"**,200);  
 }**else** map.put(**"status"**,404);  
 } **catch** (Exception e) {  
 map.put(**"status"**,500);  
 e.printStackTrace();  
 }  
 **return** map;  
 }  
  
}

**Mapper.java**

**package** com.lshs.springbootdemo1.mapper;  
  
**import** com.lshs.springbootdemo1.entry.BiAn;  
**import** com.lshs.springbootdemo1.entry.QueryEntry;  
  
**import** java.util.List;  
  
**public interface** BiAnMapper {  
 **int** deleteByPrimaryKey(Integer id);  
  
 **int** insert(BiAn record);  
  
 **int** insertSelective(BiAn record);  
  
 BiAn selectByPrimaryKey(Integer id);  
  
 **int** updateByPrimaryKeySelective(BiAn record);  
  
 **int** updateByPrimaryKey(BiAn record);  
  
 List<BiAn> findByCase(QueryEntry queryEntry);  
  
 **int** deleteBian();  
  
 BiAn lastData(String spx);  
 BiAn firstData(String spx);  
}

**Mpaaer.xml**

*<?***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.lshs.springbootdemo1.mapper.BiAnMapper"**>  
   
自动生成的代码

<**delete id="deleteBian"**>  
 DELETE from bian  
 </**delete**>  
  
 <**select id="lastData" resultMap="BaseResultMap" parameterType="string"**>  
 SELECT *\** FROM bian where id=  
 (select *max*(id) from bian where screen=#{spx});  
 </**select**>  
  
 <**select id="firstData" resultMap="BaseResultMap" parameterType="string"**>  
 SELECT *\** FROM bian where id=  
 (select *max*(id) from bian where screen=#{spx} AND page=1);  
 </**select**>  
  
 <**select id="findByCase" resultMap="BaseResultMap" parameterType="com.lshs.springbootdemo1.entry.QueryEntry"**>  
 SELECT *\** FROM bian  
 <**where**>  
 <**if test="screen !='' and screen!=null"**>  
 and screen=#{screen}  
 </**if**>  
 <**if test="category !='' and category !=null"**>  
 and category=#{category}  
 </**if**>  
 <**if test="title !='' and title !=null"**>  
 and title LIKE CONCAT('%',#{title},'%')  
 </**if**>  
 </**where**>  
 </**select**>  
</**mapper**>

**Entry.java**

**package** com.lshs.springbootdemo1.entry;  
  
**public class** BiAn {  
 **private** Integer **id**;  
  
 **private** String **title**;  
  
 **private** String **url1**;  
  
 **private** String **url2**;  
  
 **private** String **url3**;  
  
 **private** String **screen**;  
  
 **private** String **category**;  
  
 **private** String **img1**;  
  
 **private** String **img2**;  
  
 **private** String **updatetime**;  
  
 **private** Integer **page**;  
  
 **private** String **fromurl**;

**getter setter**

**service.java**

**package** com.lshs.springbootdemo1.service;  
  
**import** com.lshs.springbootdemo1.entry.BiAn;  
**import** com.lshs.springbootdemo1.entry.EUDataGridResult;  
**import** com.lshs.springbootdemo1.entry.QueryEntry;  
  
*/\*\*  
 \** ***@Description:*** *定义接口  
 \** ***@author:*** *LuShao  
 \** ***@create:*** *2018-08-24 10:22  
 \*\*/***public interface** BianService {  
  
 **int** saveBian(BiAn biAn);  
  
 **int** deleteBian();  
  
 BiAn lastData(String spx);  
 BiAn firstData(String spx);  
  
 EUDataGridResult findImgs(QueryEntry queryEntry);  
  
}

**serviceImpl.java**

**package** com.lshs.springbootdemo1.service;  
  
**import** com.github.pagehelper.PageHelper;  
**import** com.github.pagehelper.PageInfo;  
**import** com.lshs.springbootdemo1.entry.BiAn;  
**import** com.lshs.springbootdemo1.entry.EUDataGridResult;  
**import** com.lshs.springbootdemo1.entry.QueryEntry;  
**import** com.lshs.springbootdemo1.mapper.BiAnMapper;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Service;  
  
**import** java.util.List;  
  
*/\*\*  
 \** ***@Description:*** *\** ***@author:*** *LuShao  
 \** ***@create:*** *2018-08-24 10:23  
 \*\*/*@Service  
**public class** BianServiceI **implements** BianService {  
  
 @Autowired  
 **private** BiAnMapper **biAnMapper**;  
  
 @Override  
 **public int** saveBian(BiAn biAn) {  
 **int** i = **biAnMapper**.insertSelective(biAn);  
 **return** i;  
 }  
  
 @Override  
 **public int** deleteBian() {  
 **int** i = **biAnMapper**.deleteBian();  
 **return** i;  
 }  
  
 @Override  
 **public** BiAn lastData(String spx) {  
 **return biAnMapper**.lastData(spx);  
 }  
  
 @Override  
 **public** BiAn firstData(String spx) {  
 **return biAnMapper**.firstData(spx);  
 }  
  
 */\*\*  
 \*  
 \** ***@param page*** *\** ***@param rows*** *\** ***@param queryEntry*** *\** ***@return*** *\*/* @Override  
 **public** EUDataGridResult findImgs(QueryEntry queryEntry){  
 PageHelper.*startPage*(queryEntry.getPage(), queryEntry.getRows());  
 List<BiAn> list = **biAnMapper**.findByCase(queryEntry);  
 PageInfo<BiAn> info=**new** PageInfo<>(list);  
 EUDataGridResult result=**new** EUDataGridResult();  
 result.setRows(list);  
 result.setTotal(info.getTotal());  
 **return** result;  
 }  
  
  
}