武汉纺织大学

Web应用开发课程设计

**生源地人数统计图**

**学 院： 数学与计算机学院**

**班 级： 软件11802**

**姓 名： 胡徐安**

**学 号： 1804240425**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月15日**

目 录

[1 需求分析 1](#_Toc59100499)

[1.1 实现地图显示 1](#_Toc59100500)

[1.2 以地图为坐标系进行数据显示 1](#_Toc59100501)

[2 系统设计 1](#_Toc59100502)

[2.1 ER图 1](#_Toc59100503)

[2.2 UML类图（Class Diagram） 1](#_Toc59100504)

[2.3 UML时序图（Sequence Diagram） 2](#_Toc59100505)

[2.4 UML活动图（Activity Diagram） 3](#_Toc59100506)

[2.4.1 系统登录 3](#_Toc59100507)

[2.4.2 \*\*\*\*\*\*模块 3](#_Toc59100508)

[2.4.3 \*\*\*\*\*\*模块 3](#_Toc59100509)

[2.5 UML活动图（Activity Diagram） 3](#_Toc59100510)

[2.5.1 登陆 3](#_Toc59100511)

[2.5.2 收银 4](#_Toc59100512)

[2.5.3 \*\*\*模块 5](#_Toc59100513)

[2.5.4 \*\*\*\*模块 5](#_Toc59100514)

[3 系统实现 6](#_Toc59100515)

[3.1 项目结构 6](#_Toc59100516)

[3.2 配置文件 6](#_Toc59100517)

[3.2.1 jdbc.properties文件 6](#_Toc59100518)

[3.2.2 log4j.properties文件 6](#_Toc59100519)

[3.2.3 mybatis-config.xml文件 7](#_Toc59100520)

[3.3 VO类User.java 7](#_Toc59100521)

[3.4 DAO接口类IUserDAO.java 8](#_Toc59100522)

[3.5 接口映射文件UserMapper.xml 8](#_Toc59100523)

[3.6 工具包Util 8](#_Toc59100524)

[3.6.1 MybatisUtils.java 8](#_Toc59100525)

[3.6.2 MD5Util.java 9](#_Toc59100526)

[3.7 服务层UserService.java 10](#_Toc59100527)

[3.8 用户界面Driver.java 11](#_Toc59100528)

[3.9 \*\*\*\*\*\*.java 11](#_Toc59100529)

[4 系统测试 11](#_Toc59100530)

[5 系统总结 11](#_Toc59100531)

# 1 需求分析

现在数据难以直观表示，利用百度地图和echarts主键实现数据的可视化操作你，将学生的生源地人数作为数据在地图上直观的显示出来。其功能如下：

## 1.1 实现地图显示

当程序运行时，能通过百度地图的API调用百度地图显示。

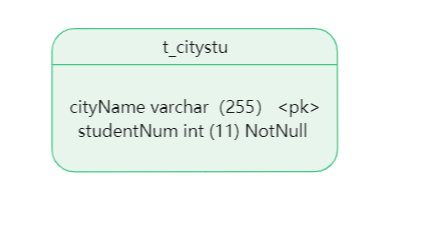
## 1.2 以地图为坐标系进行数据显示

将数据库的数据读出来后与地图的地理坐标相绑定，并利用ecahrts组件用散点图实现数据的可视化，散点图以地图为地理坐标系，并将数据排序，前10个数据利用带涟漪的散点图进行突出显示。

# 2 系统设计

## 2.1 ER图

数据库设计了一张表，城市学生人数表，对应的ER图如图2-1所示。



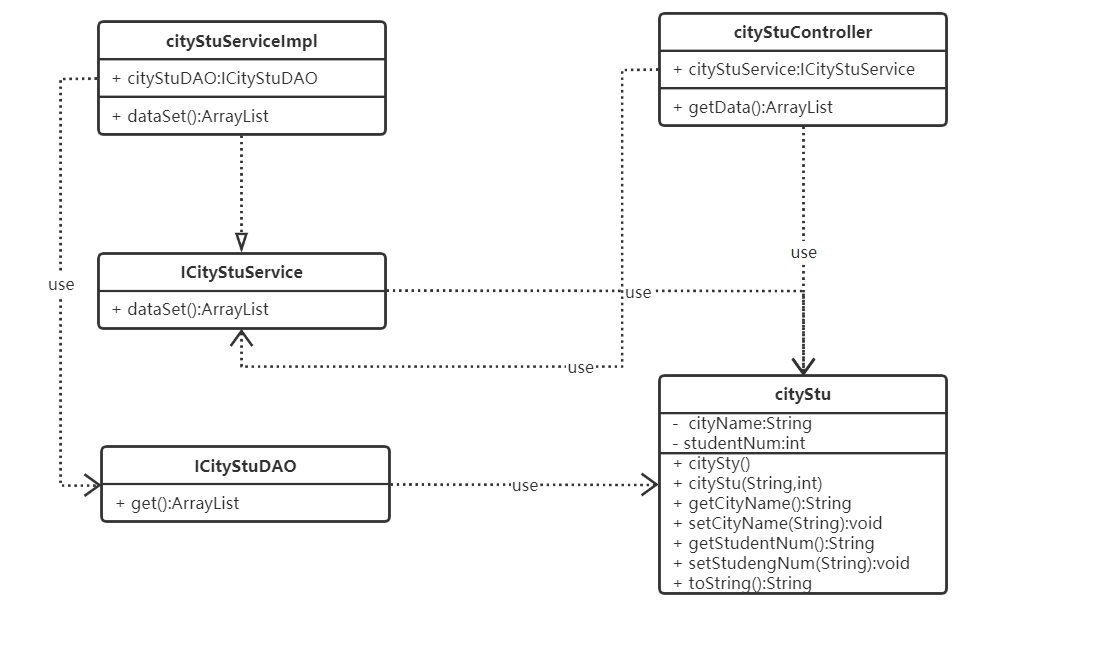
2-1 城市学生人数

## 2.2 UML类图（Class Diagram）

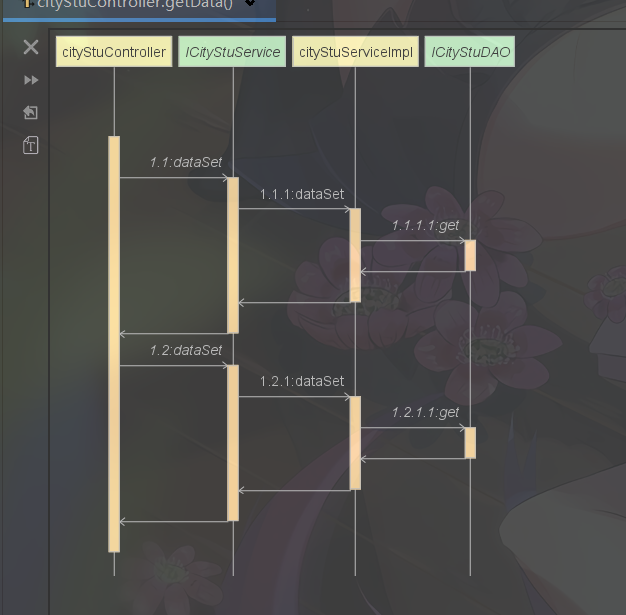
对于后端数据获取部分，共设计如下5个类。

* VO实体类 cityStu:与数据库进行映射的类。主要由属性，setter, getter方法组成，VO类中的属性与表中的字段相对应，每一个VO类的对象都表示表中的每一条记录
* DAO接口 ICityStuDAO：主要定义操作的接口，定义一系列数据库的原子性操作，例如增删改查（通常称为CRUD）等。
* Service服务接口类 ICityStuService:主要对数据库原子型操作进行整合，进行对事物逻辑的控制。
* Service服务实现类 cityStuServiceImpl:实现对应的接口类。
* Controller servlet控制器类：cityStuController类：主要用于接收前端发送的请求，作出相关的响应操作，返回前端所需要的数据。
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

各类的结构及类之间的关系如图所示：

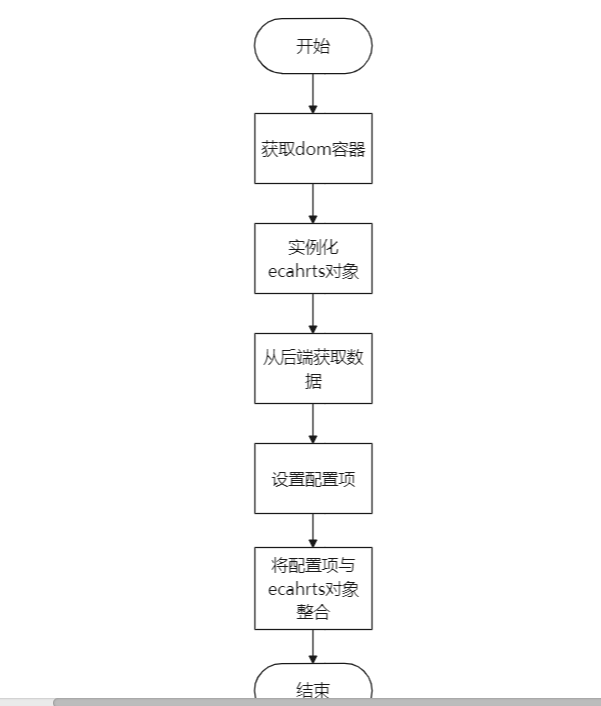


## 2.3 UML时序图（Sequence Diagram）



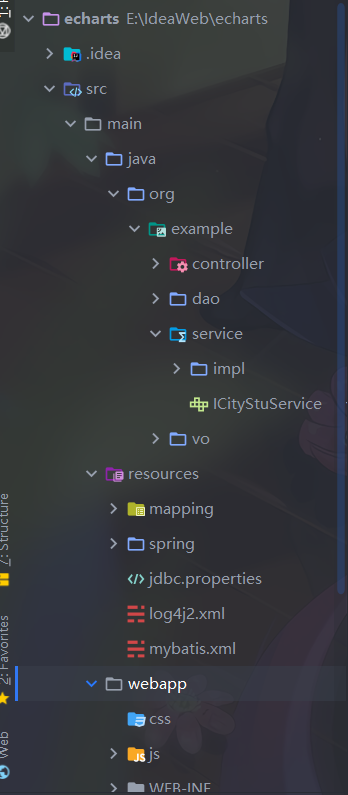
## 2.4 UML活动图（Activity Diagram）

### 2.4.1 地图显示



# 3 系统实现

## 3.1 项目结构



## 3.2 配置文件

### 3.2.1 Maven项目配置文件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>org.example</groupId>  
 <artifactId>echarts</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <packaging>war</packaging>  
  
 <name>echarts Maven Webapp</name>  
 *<!-- FIXME change it to the project's website -->* <url>http://www.example.com</url>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <maven.compiler.source>1.8</maven.compiler.source>  
 <maven.compiler.target>1.8</maven.compiler.target>  
 <springversion>5.0.0.RELEASE</springversion> *<!-- spring版本号 -->* <mybatisversion>3.3.0</mybatisversion> *<!-- mybatis版本号 -->* </properties>  
  
 <dependencies>  
 *<!-- 引入项目依赖包 -->  
 <!-- servlet API，发布时由容器提供 -->* <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>javax.servlet-api</artifactId>  
 <version>3.0.1</version>  
 *<!-- 用于编译和测试的环境，将不会被打包到lib目录下。 -->* <scope>provided</scope>  
 </dependency>  
 *<!--springframework -->* <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-aop</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-beans</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-core</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-expression</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-web</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-test</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-tx</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-jdbc</artifactId>  
 <version>${springversion}</version>  
 </dependency>  
  
 *<!-- log4j2，用于日志管理 -->* <dependency>  
 <groupId>org.apache.logging.log4j</groupId>  
 <artifactId>log4j-api</artifactId>  
 <version>2.11.2</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.logging.log4j</groupId>  
 <artifactId>log4j-core</artifactId>  
 <version>2.11.2</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.logging.log4j</groupId>  
 <artifactId>log4j-web</artifactId>  
 <version>2.11.2</version>  
 </dependency>  
  
  
 *<!-- junit单元测试 -->* <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.12</version>  
 *<!-- 表示开发的时候引入，发布的时候不会加载此包 -->* <scope>test</scope>  
 </dependency>  
  
 *<!-- 用于springMVC中的java对象和json自动转换 -->* <dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 <version>2.9.1</version>  
 </dependency>  
 *<!-- mybatis核心包 -->* <dependency>  
 <groupId>org.mybatis</groupId>  
 <artifactId>mybatis</artifactId>  
 <version>${mybatisversion}</version>  
 </dependency>  
 *<!-- mybatis/spring包 -->* <dependency>  
 <groupId>org.mybatis</groupId>  
 <artifactId>mybatis-spring</artifactId>  
 <version>1.2.2</version>  
 </dependency>  
  
 *<!--mysql驱动包 -->* <dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>5.1.49</version>  
 </dependency>  
 *<!-- 连接池，用来加载DruidDataSource -->* <dependency>  
 <groupId>com.alibaba</groupId>  
 <artifactId>druid</artifactId>  
 <version>1.0.9</version>  
 </dependency>  
 *<!-- JSTL标签库 -->* <dependency>  
 <groupId>jstl</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <finalName>echarts</finalName>  
 <pluginManagement>*<!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->* <plugins>  
 <plugin>  
 <artifactId>maven-clean-plugin</artifactId>  
 <version>3.1.0</version>  
 </plugin>  
 *<!-- see http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_war\_packaging -->* <plugin>  
 <artifactId>maven-resources-plugin</artifactId>  
 <version>3.0.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.0</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>2.22.1</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-war-plugin</artifactId>  
 <version>3.2.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-install-plugin</artifactId>  
 <version>2.5.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-deploy-plugin</artifactId>  
 <version>2.8.2</version>  
 </plugin>  
 </plugins>  
 </pluginManagement>  
 </build>  
</project>

### 3.2.2 web项目配置文件web.xml

<!DOCTYPE *web-app* PUBLIC  
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"  
 "http://java.sun.com/dtd/web-app\_2\_3.dtd">  
  
<web-app *xmlns:xsi*="http://www.w3.org/2001/XMLSchema-instance"  
 *xmlns*="http://java.sun.com/xml/ns/javaee"  
 *xsi:schemaLocation*="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"  
 *id*="WebApp\_ID" *version*="3.0"  
>  
 <display-name>Archetype Created Web Application</display-name>  
 <welcome-file-list>  
 <welcome-file>index.html</welcome-file>  
 <welcome-file>index.jsp</welcome-file>  
 </welcome-file-list>  
 *<!-- Spring和mybatis的配置文件 -->* <context-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>classpath:spring/spring-mybatis.xml</param-value>  
 </context-param>  
 *<!-- log4j2的配置文件 -->* <context-param>  
 <param-name>log4jConfigLocation</param-name>  
 <param-value>classpath:log4j2.xml</param-value>  
 </context-param>  
 *<!-- 编码过滤器 -->* <filter>  
 <filter-name>encodingFilter</filter-name>  
 <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  
 <async-supported>true</async-supported>  
 <init-param>  
 <param-name>encoding</param-name>  
 <param-value>UTF-8</param-value>  
 </init-param>  
 </filter>  
 <filter-mapping>  
 <filter-name>encodingFilter</filter-name>  
 <url-pattern>/\*</url-pattern>  
 </filter-mapping>  
 *<!-- Spring监听器 -->* <listener>  
 <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  
 </listener>  
 *<!-- 防止Spring内存溢出监听器 -->* <listener>  
 <listener-class>org.springframework.web.util.IntrospectorCleanupListener</listener-class>  
 </listener>  
 <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:spring/spring-mvc.xml</param-value>  
 </init-param>  
 <load-on-startup>1</load-on-startup>  
 <async-supported>true</async-supported>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>SpringMVC</servlet-name>  
 <url-pattern>\*.do</url-pattern>  
 </servlet-mapping>  
</web-app>

在main目录下新建Source Folder，名为resources

### 3.2.3 jdbc.properties文件

该配置文件主要是以文件形式保存数据库的驱动类名称，连接数据库的URL地址，访问数据库的用户名及对应的密码，程序运行时会读取该文件相关信息，避免硬编码，当相关信息发生变化时，只需修改配置文件而不用修改源代码，增加程序的可扩展型。

*driver*=com.mysql.jdbc.Driver  
*url*=jdbc:mysql://localhost:3306/echartsmap?useUnicode=true&characterEncoding=utf-8&useSSL=false&serverTimezone=UTC  
*username*=root  
*password*=huxuan  
*#定义初始连接数  
initialSize*=0  
*#定义最大连接数  
maxActive*=20  
*#定义最小空闲  
minIdle*=1  
*#定义最长等待时间  
maxWait*=60000  
*#验证连接是否可用，使用的SQL语句  
validationQuery* =SELECT 1

### 3.2.4 log4j.properties文件

日志文件配置

<?*xml version*="1.0" *encoding*="UTF-8"?>  
*<!--日志级别以及优先级排序: OFF > FATAL > ERROR > WARN > INFO > DEBUG > TRACE > ALL -->  
<!-- status配置Log4j2启动和加载配置文件时的日志输出登记 -->*<Configuration *status*="info">  
 *<!--appenders:定义输出内容,输出格式,输出方式,日志保存策略等,常用其下三种标签[console,File,RollingFile] -->  
 <!--Appender可以理解为日志的输出目的地 -->* <Appenders>  
 *<!--console :控制台输出的配置 -->* <Console *name*="Console" *target*="SYSTEM\_OUT">  
 *<!--  
 %d 输出日志时间点的日期或时间  
 %t 输出产生该日志事件的线程名  
 %p 输出优先级，即DEBUG,INFO,WARN,ERROR,FATAL  
 %F 输出日志消息产生时所在的文件名称  
 %c 输出所属的类目,通常就是所在类的全名  
 %L 输出代码中的行号  
 -->* <PatternLayout *pattern*="%d{YYYY-MM-dd HH:mm:ss} %-5p [%c,%L]- %msg%n" />  
 </Console>  
  
 <RollingFile *name*="RollingFile" *filename*="log/ZycTest.log"  
 *filepattern*="${logPath}/%d{YYYYMMddHHmmss}-fargo.log">  
 <PatternLayout *pattern*="%d{YYYY-MM-dd HH:mm:ss} %-5p [%c,%L] - %msg%n" />  
 <Policies>  
 <SizeBasedTriggeringPolicy *size*="100 MB" />  
 </Policies>  
 <DefaultRolloverStrategy *max*="20" />  
 </RollingFile>  
 </Appenders>  
 *<!--定义logger，只有定义了logger并引入的appender，appender才会生效 -->* <Loggers>  
 *<!-- Logger节点用来单独指定日志的形式，可以为通过name属性设置指定包下的class指定不同的日志级别等 ,  
 可以设置Logger的additivity="false"只在自定义的Appender中进行输出 -->* <logger *name*="dao" *level*="debug" *additivity*="false">  
 <appender-ref *ref*="Console" />  
 </logger>  
 *<!-- Root节点用来指定项目的根日志，如果没有单独指定Logger，那么就会默认使用该Root日志输出 -->* <Root *level*="info">  
 <AppenderRef *ref*="Console" />  
 <AppenderRef *ref*="RollingFile" />  
 </Root>  
 </Loggers>  
</Configuration>

### 3.2.5 mybatis.xml文件（配置了spring和mybatis整合可以不配置）

<?*xml version*="1.0" *encoding*="UTF-8"?>  
<!DOCTYPE *configuration* PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">  
<configuration>  
 <settings>  
 <setting *name*="logImpl" *value*="LOG4J2" />  
 </settings>  
</configuration>

### 3.2.6 在resources目录下新建一个mapping,存放mybatis的映射文件

接口映射文件：cityStuMapping.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*="org.example.dao.ICityStuDAO">  
 <select *id*="get" *resultType*="org.example.vo.cityStu">  
 select \* from t\_citystu order by studentNum desc  
 </select>  
</mapper>

### 3.2.7 在resources目录下新建一个spring文件夹，存放spring相关配置

SpringMvc配置文件：spring-mvc.xml

<?*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"  
 *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/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd">  
 <mvc:annotation-driven />  
 *<!-- 扫描类包，将标注Spring注解的类自动转化Bean，同时完成Bean的注入 -->* <context:component-scan *base-package*="org.example.controller" />  
 <context:component-scan *base-package*="org.example.service" />  
  
 *<!--配置视图解析器 -->  
 <!-- ViewResolver 视图解析器 用于将返回的ModelAndView对象进行分离  
 InternalResourceViewResolver：用于支持Servlet、JSP视图解析；  
 viewClass：JstlView表示JSP模板页面需要使用JSTL标签库，classpath中必须包含jstl的相关jar包； prefix和suffix：查找视图页面的前缀和后缀（前缀[逻辑视图名]后缀），  
 比如传进来的逻辑视图名为hello，则该该jsp视图页面应该存放在“/hello.jsp”； -->* <bean *class*="org.springframework.web.servlet.view.InternalResourceViewResolver">  
 <property *name*="prefix" *value*="/"></property> *<!-- 视图放在webroot/下 -->* <property *name*="suffix" *value*=".jsp"></property>  
 </bean>  
  
</beans>

Mybatis和spring整合文件：spring-mybatis.xml

<?*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"  
 *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">  
  
 *<!-- 扫描类包，将标注Spring注解的类自动转化Bean，同时完成Bean的注入 -->* <context:component-scan *base-package*="org.example.service" />  
 *<!-- 引入jdbc配置文件 -->* <bean *id*="propertyConfigurer"  
 *class*="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">  
 <property *name*="location" *value*="classpath:jdbc.properties" />  
 </bean>  
 *<!-- 数据库连接池 -->* <bean *id*="dataSource" *class*="com.alibaba.druid.pool.DruidDataSource"  
 *destroy-method*="close">  
 <property *name*="driverClassName" *value*="${driver}" />  
 <property *name*="url" *value*="${url}" />  
 <property *name*="username" *value*="${username}" />  
 <property *name*="password" *value*="${password}" />  
 *<!-- 初始化连接大小 -->* <property *name*="initialSize" *value*="${initialSize}"></property>  
 *<!-- 连接池最大数量 -->* <property *name*="maxActive" *value*="${maxActive}"></property>  
 *<!-- 连接池最小空闲 -->* <property *name*="minIdle" *value*="${minIdle}"></property>  
 *<!-- 获取连接最大等待时间 -->* <property *name*="maxWait" *value*="${maxWait}"></property>  
 *<!-- 验证连接是否可用，使用的SQL语句 -->* <property *name*="validationQuery" *value*="SELECT 1"></property>  
 </bean>  
 *<!-- spring和MyBatis整合，不需要mybatis的配置映射文件 -->* <bean *id*="sqlSessionFactory" *class*="org.mybatis.spring.SqlSessionFactoryBean">  
 <property *name*="dataSource" *ref*="dataSource" />  
 <property *name*="configLocation" *value*="classpath:mybatis.xml"></property>  
 *<!-- 自动扫描mapping.xml文件 -->* <property *name*="mapperLocations" *value*="classpath:mapping/\*.xml"></property>  
 </bean>  
 *<!-- DAO接口所在包名，Spring会自动查找其下的类 -->* <bean *class*="org.mybatis.spring.mapper.MapperScannerConfigurer">  
 <property *name*="basePackage" *value*="org.example.dao"/>  
 <property *name*="sqlSessionFactoryBeanName" *value*="sqlSessionFactory"></property>  
 </bean>  
 *<!-- (事务管理)transaction manager-->* <bean *id*="transactionManager"  
 *class*="org.springframework.jdbc.datasource.DataSourceTransactionManager">  
 <property *name*="dataSource" *ref*="dataSource" />  
 </bean>  
</beans>

## 3.3 VO类cityStu.java

数据库对应实体类：

*package* org.example.vo;  
  
*public class* cityStu {  
 *private* String cityName;  
 *private int* studentNum;  
  
 *public* cityStu() {  
 *super*();  
 }  
  
 *public* cityStu(String cityName, *int* studentNum) {  
 *this*.cityName = cityName;  
 *this*.studentNum = studentNum;  
 }  
  
 *public* String getCityName() {  
 *return* cityName;  
 }  
  
 *public void* setCityName(String cityName) {  
 *this*.cityName = cityName;  
 }  
  
 *public int* getStudentNum() {  
 *return* studentNum;  
 }  
  
 *public void* setStudentNum(*int* studentNum) {  
 *this*.studentNum = studentNum;  
 }  
  
 @Override  
 *public* String toString() {  
 *return* "cityStu{" +  
 "cityName='" + cityName + '\'' +  
 ", studentNum=" + studentNum +  
 '}';  
 }  
}

## 3.4 DAO接口类ICityStuDAO.java

数据库原子化操作的接口文件，具体实现在mapping的xml文件中

*package* org.example.dao;  
  
*import* org.apache.ibatis.annotations.Param;  
*import* org.example.vo.cityStu;  
  
*import* java.util.ArrayList;  
  
*public interface ICityStuDAO* {  
 *//查询t\_cityStu表，将数据以数组返回  
 public* ArrayList<cityStu> get();  
}

## 3.5 service接口类 ICityStuService

对dao层的原子型操作进行整合，实现简单的逻辑管理。

*package* org.example.service;  
  
*import* org.example.vo.cityStu;  
  
*import* java.util.*List*;  
  
*public interface ICityStuService* {  
 *public List*<cityStu> dataSet();  
}

## 3.6 service.impl接口实现类 cityStuService

对service层的接口进行实现

*package* org.example.service.impl;  
  
*import* org.example.dao.*ICityStuDAO*;  
*import* org.example.service.*ICityStuService*;  
*import* org.example.vo.cityStu;  
*import* org.springframework.beans.factory.annotation.Autowired;  
*import* org.springframework.stereotype.Service;  
  
*import* java.util.ArrayList;  
*import* java.util.*List*;  
  
@Service  
*public class* cityStuServiceImpl *implements ICityStuService* {  
 @Autowired  
 *private ICityStuDAO* cityStuDAO;  
  
 @Override  
 *public List*<cityStu> dataSet() {  
 *List*<cityStu> cityStuList=*new* ArrayList<>();  
 cityStuList=cityStuDAO.get();  
 *return* cityStuList;  
 }  
}

## 3.7 控制器类 cityStuController.java

接收前端请求和返回服务的servlet类

*package* org.example.controller;  
  
*import* org.example.service.*ICityStuService*;  
*import* org.example.vo.cityStu;  
*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.RequestMethod;  
*import* org.springframework.web.bind.annotation.ResponseBody;  
  
*import* java.util.*List*;  
  
@Controller  
*public class* cityStuController {  
 @Autowired  
 *private ICityStuService* cityStuService;  
  
 *//取数据* @RequestMapping(value = "/ajaxGetData.do", method = {RequestMethod.GET,RequestMethod.POST})  
 @ResponseBody  
 *public List*<cityStu> getData(){  
 *List*<cityStu> data=cityStuService.dataSet();  
 cityStuService.dataSet();  
 System.out.println(data.toString());  
  
 *return* data;*// 框架使用MappingJackson2HttpMessageConverter类将java类型转换为json字符串放在response  
 // body中* }  
}

## 3.8 前端界面 mapTest.html

<!DOCTYPE *html*>  
<html *style*="height: 100%">  
 <head>  
 <meta *charset*="utf-8">  
   
 </head>  
 <body *style*="height: 100%; margin: 0">  
 <div *id*="container" *style*="height: 100%"></div>  
 <script *type*="text/javascript" *src*="js/jquery-3.5.0.min.js"></script>  
 <script *type*="text/javascript" *src*="js/map.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts@4/dist/echarts.min.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts-gl@1/dist/echarts-gl.min.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts-stat@1/dist/ecStat.min.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts@4/dist/extension/dataTool.min.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts@4/map/js/china.js"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts@4/map/js/world.js"></script>  
 <script *type*="text/javascript" *src*="https://api.map.baidu.com/api?v=2.0&ak=t9gjGC0MLTETBedQLFzTSWrIvfWHPNQA"></script>  
 <script *type*="text/javascript" *src*="https://cdn.jsdelivr.net/npm/echarts@4/dist/extension/bmap.min.js"></script>  
   
 </body>  
</html>

在webapp下创建js目录，放js文件

## 3.9 map.js

地图显示和数据处理的js文件

*var* data;  
$(document).ready(*function* () {  
 *getData*();  
 *console*.log(data);  
 *mapSet*();  
});  
  
*function getData*(){  
 $.ajax({  
 type: "get",  
 url: "ajaxGetData.do",  
 data: *null*,  
 dataType: "json",  
 async:*false*,*//将ajax请求改成同步* success: *function* (response) {  
 *console*.log(*typeof* response);  
 *console*.log(response.length);  
 *console*.log(response);  
 data=response;  
 *console*.log(data);  
 *console*.log(*typeof* data);  
 *console*.log(data.length);  
 }  
 });  
}  
  
*function mapSet*(){  
 *//获取dom容器  
 var* dom = document.getElementById("container");  
 *//实例化echarts对象  
 var* myChart = echarts.init(dom);  
 *//alert('1111');  
 var* app = {};  
   
 option = *null*;  
 *console*.log(data);  
 *//console.log(data.length);  
 //填地理坐标绘制地图  
var* geoCoordMap = {  
 '海门':[121.15,31.89],  
 '鄂尔多斯':[109.781327,39.608266],  
 '招远':[120.38,37.35],  
 '舟山':[122.207216,29.985295],  
 '齐齐哈尔':[123.97,47.33],  
 '盐城':[120.13,33.38],  
 '赤峰':[118.87,42.28],  
 '青岛':[120.33,36.07],  
 '乳山':[121.52,36.89],  
 '金昌':[102.188043,38.520089],  
 '泉州':[118.58,24.93],  
 '莱西':[120.53,36.86],  
 '日照':[119.46,35.42],  
 '胶南':[119.97,35.88],  
 '南通':[121.05,32.08],  
 '拉萨':[91.11,29.97],  
 '云浮':[112.02,22.93],  
 '梅州':[116.1,24.55],  
 '文登':[122.05,37.2],  
 '上海':[121.48,31.22],  
 '攀枝花':[101.718637,26.582347],  
 '威海':[122.1,37.5],  
 '承德':[117.93,40.97],  
 '厦门':[118.1,24.46],  
 '汕尾':[115.375279,22.786211],  
 '潮州':[116.63,23.68],  
 '丹东':[124.37,40.13],  
 '太仓':[121.1,31.45],  
 '曲靖':[103.79,25.51],  
 '烟台':[121.39,37.52],  
 '福州':[119.3,26.08],  
 '瓦房店':[121.979603,39.627114],  
 '即墨':[120.45,36.38],  
 '抚顺':[123.97,41.97],  
 '玉溪':[102.52,24.35],  
 '张家口':[114.87,40.82],  
 '阳泉':[113.57,37.85],  
 '莱州':[119.942327,37.177017],  
 '湖州':[120.1,30.86],  
 '汕头':[116.69,23.39],  
 '昆山':[120.95,31.39],  
 '宁波':[121.56,29.86],  
 '湛江':[110.359377,21.270708],  
 '揭阳':[116.35,23.55],  
 '荣成':[122.41,37.16],  
 '连云港':[119.16,34.59],  
 '葫芦岛':[120.836932,40.711052],  
 '常熟':[120.74,31.64],  
 '东莞':[113.75,23.04],  
 '河源':[114.68,23.73],  
 '淮安':[119.15,33.5],  
 '泰州':[119.9,32.49],  
 '南宁':[108.33,22.84],  
 '营口':[122.18,40.65],  
 '惠州':[114.4,23.09],  
 '江阴':[120.26,31.91],  
 '蓬莱':[120.75,37.8],  
 '韶关':[113.62,24.84],  
 '嘉峪关':[98.289152,39.77313],  
 '广州':[113.23,23.16],  
 '延安':[109.47,36.6],  
 '太原':[112.53,37.87],  
 '清远':[113.01,23.7],  
 '中山':[113.38,22.52],  
 '昆明':[102.73,25.04],  
 '寿光':[118.73,36.86],  
 '盘锦':[122.070714,41.119997],  
 '长治':[113.08,36.18],  
 '深圳':[114.07,22.62],  
 '珠海':[113.52,22.3],  
 '宿迁':[118.3,33.96],  
 '咸阳':[108.72,34.36],  
 '铜川':[109.11,35.09],  
 '平度':[119.97,36.77],  
 '佛山':[113.11,23.05],  
 '海口':[110.35,20.02],  
 '江门':[113.06,22.61],  
 '章丘':[117.53,36.72],  
 '肇庆':[112.44,23.05],  
 '大连':[121.62,38.92],  
 '临汾':[111.5,36.08],  
 '吴江':[120.63,31.16],  
 '石嘴山':[106.39,39.04],  
 '沈阳':[123.38,41.8],  
 '苏州':[120.62,31.32],  
 '茂名':[110.88,21.68],  
 '嘉兴':[120.76,30.77],  
 '长春':[125.35,43.88],  
 '胶州':[120.03336,36.264622],  
 '银川':[106.27,38.47],  
 '张家港':[120.555821,31.875428],  
 '三门峡':[111.19,34.76],  
 '锦州':[121.15,41.13],  
 '南昌':[115.89,28.68],  
 '柳州':[109.4,24.33],  
 '三亚':[109.511909,18.252847],  
 '自贡':[104.778442,29.33903],  
 '吉林':[126.57,43.87],  
 '阳江':[111.95,21.85],  
 '泸州':[105.39,28.91],  
 '西宁':[101.74,36.56],  
 '宜宾':[104.56,29.77],  
 '呼和浩特':[111.65,40.82],  
 '成都':[104.06,30.67],  
 '大同':[113.3,40.12],  
 '镇江':[119.44,32.2],  
 '桂林':[110.28,25.29],  
 '张家界':[110.479191,29.117096],  
 '宜兴':[119.82,31.36],  
 '北海':[109.12,21.49],  
 '西安':[108.95,34.27],  
 '金坛':[119.56,31.74],  
 '东营':[118.49,37.46],  
 '牡丹江':[129.58,44.6],  
 '遵义':[106.9,27.7],  
 '绍兴':[120.58,30.01],  
 '扬州':[119.42,32.39],  
 '常州':[119.95,31.79],  
 '潍坊':[119.1,36.62],  
 '重庆':[106.54,29.59],  
 '台州':[121.420757,28.656386],  
 '南京':[118.78,32.04],  
 '滨州':[118.03,37.36],  
 '贵阳':[106.71,26.57],  
 '无锡':[120.29,31.59],  
 '本溪':[123.73,41.3],  
 '克拉玛依':[84.77,45.59],  
 '渭南':[109.5,34.52],  
 '马鞍山':[118.48,31.56],  
 '宝鸡':[107.15,34.38],  
 '焦作':[113.21,35.24],  
 '句容':[119.16,31.95],  
 '北京':[116.46,39.92],  
 '徐州':[117.2,34.26],  
 '衡水':[115.72,37.72],  
 '包头':[110,40.58],  
 '绵阳':[104.73,31.48],  
 '乌鲁木齐':[87.68,43.77],  
 '枣庄':[117.57,34.86],  
 '杭州':[120.19,30.26],  
 '淄博':[118.05,36.78],  
 '鞍山':[122.85,41.12],  
 '溧阳':[119.48,31.43],  
 '库尔勒':[86.06,41.68],  
 '安阳':[114.35,36.1],  
 '开封':[114.35,34.79],  
 '济南':[117,36.65],  
 '德阳':[104.37,31.13],  
 '温州':[120.65,28.01],  
 '九江':[115.97,29.71],  
 '邯郸':[114.47,36.6],  
 '临安':[119.72,30.23],  
 '兰州':[103.73,36.03],  
 '沧州':[116.83,38.33],  
 '临沂':[118.35,35.05],  
 '南充':[106.110698,30.837793],  
 '天津':[117.2,39.13],  
 '富阳':[119.95,30.07],  
 '泰安':[117.13,36.18],  
 '诸暨':[120.23,29.71],  
 '郑州':[113.65,34.76],  
 '哈尔滨':[126.63,45.75],  
 '聊城':[115.97,36.45],  
 '芜湖':[118.38,31.33],  
 '唐山':[118.02,39.63],  
 '平顶山':[113.29,33.75],  
 '邢台':[114.48,37.05],  
 '德州':[116.29,37.45],  
 '济宁':[116.59,35.38],  
 '荆州':[112.239741,30.335165],  
 '宜昌':[111.3,30.7],  
 '义乌':[120.06,29.32],  
 '丽水':[119.92,28.45],  
 '洛阳':[112.44,34.7],  
 '秦皇岛':[119.57,39.95],  
 '株洲':[113.16,27.83],  
 '石家庄':[114.48,38.03],  
 '莱芜':[117.67,36.19],  
 '常德':[111.69,29.05],  
 '保定':[115.48,38.85],  
 '湘潭':[112.91,27.87],  
 '金华':[119.64,29.12],  
 '岳阳':[113.09,29.37],  
 '长沙':[113,28.21],  
 '衢州':[118.88,28.97],  
 '廊坊':[116.7,39.53],  
 '菏泽':[115.480656,35.23375],  
 '合肥':[117.27,31.86],  
 '武汉':[114.31,30.52],  
 '大庆':[125.03,46.58]  
};  
 *//数据处理，将数据与地图地理坐标绑定  
 var* convertData = *function* (data) {  
 *console*.log("看下这个函数运行没有！");  
 *console*.log(data.length);  
 *var* res = [];  
 *for* (*var* i = 0; i < data.length; i++) {  
 *console*.log(data[i].cityName);  
 *var* geoCoord = geoCoordMap[data[i].cityName];  
 *if* (geoCoord) {  
 res.push({  
 name: data[i].cityName,  
 value: geoCoord.concat(data[i].studentNum)  
 });  
 }  
 }  
 *return* res;  
 };  
 *//设置配置项* option = {  
 title: {  
 text: '学生生源地分析图',  
 subtext: '数据来源：中国统计年鉴',  
 sublink: 'http://www.stats.gov.cn/tjsj/ndsj/2020/indexch.htm',  
 left: 'center'  
 },  
 tooltip : {  
 trigger: 'item'  
 },  
 bmap: {  
 center: [104.114129, 37.550339],  
 zoom: 5,  
 roam: *true*,  
 mapStyle: {  
 styleJson: [{  
 'featureType': 'water',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#d1d1d1'  
 }  
 }, {  
 'featureType': 'land',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#f3f3f3'  
 }  
 }, {  
 'featureType': 'railway',  
 'elementType': 'all',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'highway',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#fdfdfd'  
 }  
 }, {  
 'featureType': 'highway',  
 'elementType': 'labels',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'arterial',  
 'elementType': 'geometry',  
 'stylers': {  
 'color': '#fefefe'  
 }  
 }, {  
 'featureType': 'arterial',  
 'elementType': 'geometry.fill',  
 'stylers': {  
 'color': '#fefefe'  
 }  
 }, {  
 'featureType': 'poi',  
 'elementType': 'all',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'green',  
 'elementType': 'all',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'subway',  
 'elementType': 'all',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'manmade',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#d1d1d1'  
 }  
 }, {  
 'featureType': 'local',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#d1d1d1'  
 }  
 }, {  
 'featureType': 'arterial',  
 'elementType': 'labels',  
 'stylers': {  
 'visibility': 'off'  
 }  
 }, {  
 'featureType': 'boundary',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#fefefe'  
 }  
 }, {  
 'featureType': 'building',  
 'elementType': 'all',  
 'stylers': {  
 'color': '#d1d1d1'  
 }  
 }, {  
 'featureType': 'label',  
 'elementType': 'labels.text.fill',  
 'stylers': {  
 'color': '#999999'  
 }  
 }]  
 }  
 },  
 series : [  
 {  
 name: '学生人数',  
 type: 'scatter',  
 coordinateSystem: 'bmap',  
 data: convertData(data.slice(10)),  
 symbolSize: *function* (val) {  
 *return* val[2] / 20000;  
 },  
 encode: {  
 value: 2  
 },  
 label: {  
 formatter: '{b}',  
 position: 'right',  
 show: *false* },  
 itemStyle: {  
 color: 'purple'  
 },  
 emphasis: {  
 label: {  
 show: *true* }  
 }  
 },  
 {  
 name: 'Top10 学生人数',  
 type: 'effectScatter',  
 coordinateSystem: 'bmap',  
 data: convertData(data.slice(0, 10)),  
 symbolSize: *function* (val) {  
 *return* val[2] / 30000;  
 },  
 encode: {  
 value: 2  
 },  
 showEffectOn: 'render',  
 rippleEffect: {  
 brushType: 'stroke'  
 },  
 hoverAnimation: *true*,  
 label: {  
 formatter: '{b}',  
 position: 'right',  
 show: *true* },  
 itemStyle: {  
 color: 'purple',  
 shadowBlur: 10,  
 shadowColor: '#333'  
 },  
 zlevel: 1  
 }  
 ]  
 };;  
 *//将ecahrts对象与配置结合  
 if* (option && *typeof* option === "object") {  
 myChart.setOption(option, *true*);  
 }  
}

# 4 系统测试

运行效果：



# 5 系统总结

改系统主要在于学习ecahrts组件和bmap百度地图API，并学会如何将它们组合起来，主要遇到的问题有echarts组件的配置项的设置，通过官方文档学习后可以解决，还有百度地图api的调用，在网上查阅资料，通过百度开发者账号的创建和申请应用得到了解决，之后是数据处理过程中的问题，起初我的系统老是地图显示出来了但是数据还没有填入，之后发现是ajax异步请求不行，不能在地图加载前将数据写入前端，于是将ajax请求改成同步请求，让地图加载等待数据的传入在运行，最后得以解决，这次课程设计对我的学习提升较大，使我学会了echarts组件的使用，和百度地图api的相关开发设置，之后我也会继续学习java，争取了解更多其他的组件，是自己的能力再提升一筹。 自评：90