** **

**课程实验报告**

课程名： WEB程序设计

学 院： 信息工程学院 系 计算机科学与技术系

专 业： 计算机科学与技术可视化

班 级： 计算机科学与技术可视化171班

学 号： 6109117016

姓 名： 程婧

任课教师： 刘韬，万明

授课学期： 2019 年~~~~ 2020 年 秋季 学期

目 录

[实验一 JSP开发环境的配置运行 3](#_Toc20488)

[实验二 利用Github发布个人博客 6](#_Toc23353)

[实验三 Serlvet技术 8](#_Toc20060)

[实验四 Servlet容器模型 10](#_Toc21828)

[实验五 会话管理 14](#_Toc1096)

[实验六 JSP+JDBC开发 17](#_Toc15864)

[实验七 JSP+JavaBean+Servlet开发 21](#_Toc18144)

[实验八 Spring项目搭建 25](#_Toc11180)

[实验九 SpringMVC项目搭建 29](#_Toc23151)

[实验十 SpringBoot框架使用 31](#_Toc25506)

[实验十一 会议报名通用系统 35](#_Toc16580)

## 实验一 JSP开发环境的配置运行

一、实验目的

1．掌握设置JDK环境。

2．掌握设置Tomcat环境。

3．安装、掌握Eclipse、MyEclipse、IntelliJ IDEA软件，为学习 JSP和进行 JSP的实验做好准备工作。

4．浏览JSP 程序并运行JSP页面。

二、实验项目内容

（一）JDK安装与配置

略

（二）Tomcat安装与配置

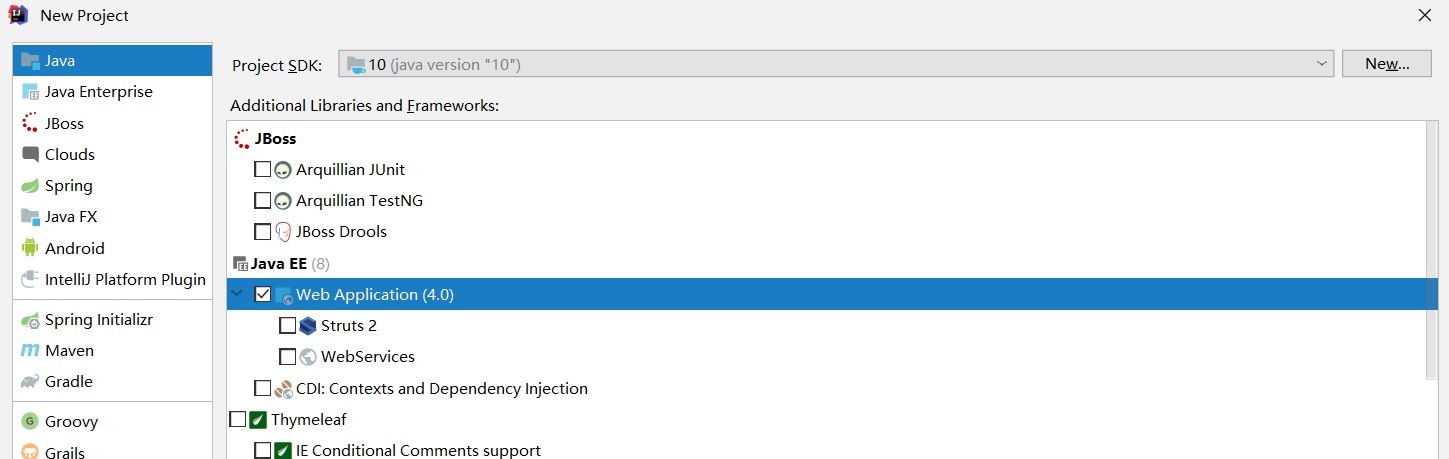
略

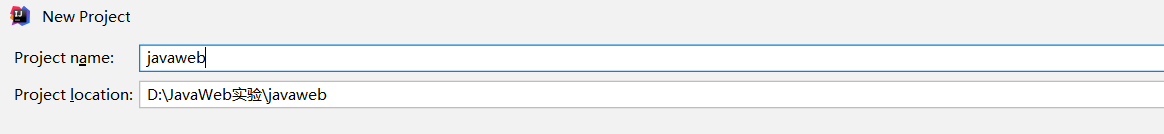
（三）IDE（Eclipse、MyEclipse或IntelliJ IDEA）的安装与配置

略

（四）在IDE上建立一个项目（javaweb）在项目中一个JSP （first.jsp）

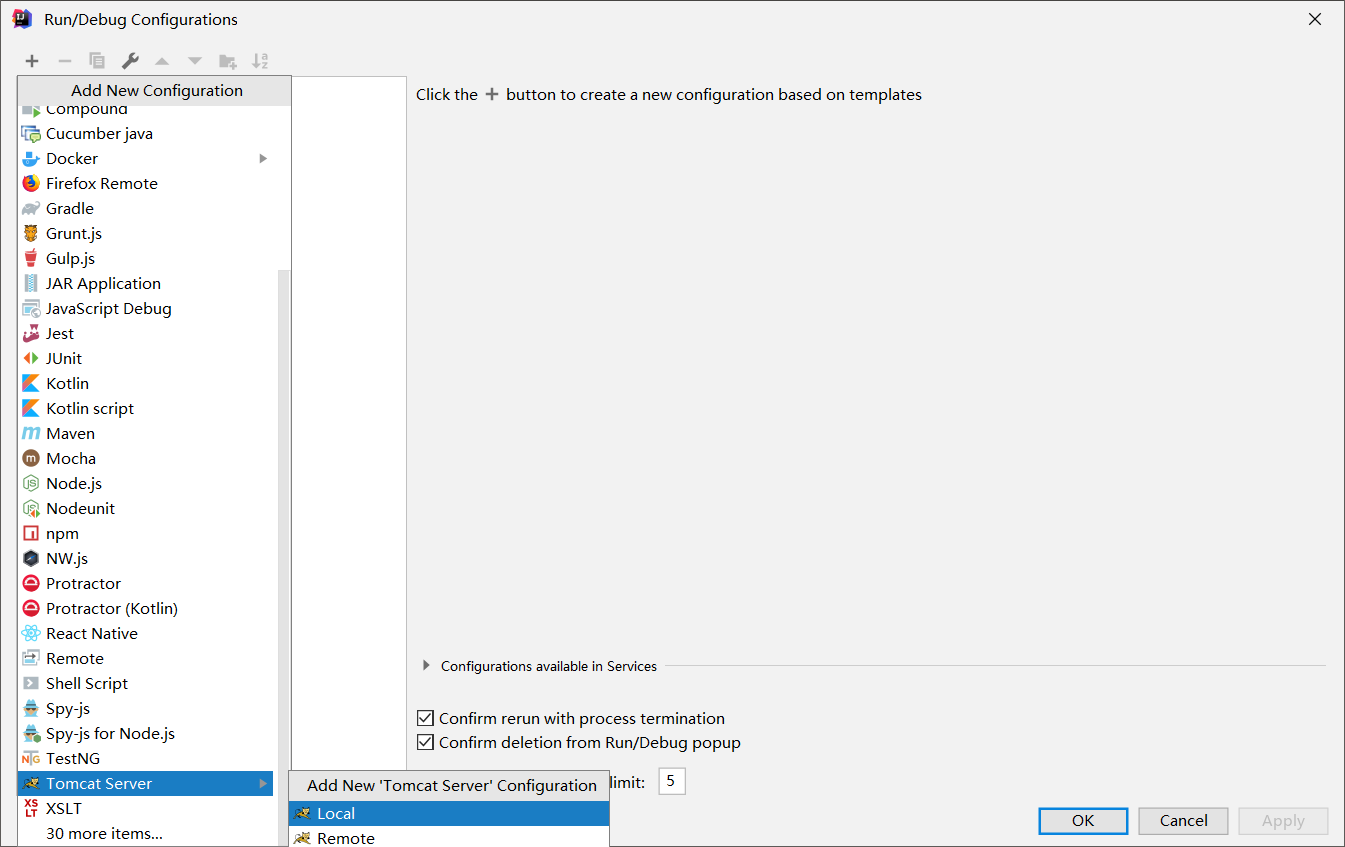
以intelliJ IDEA为例







（五）部署JSP工程文件

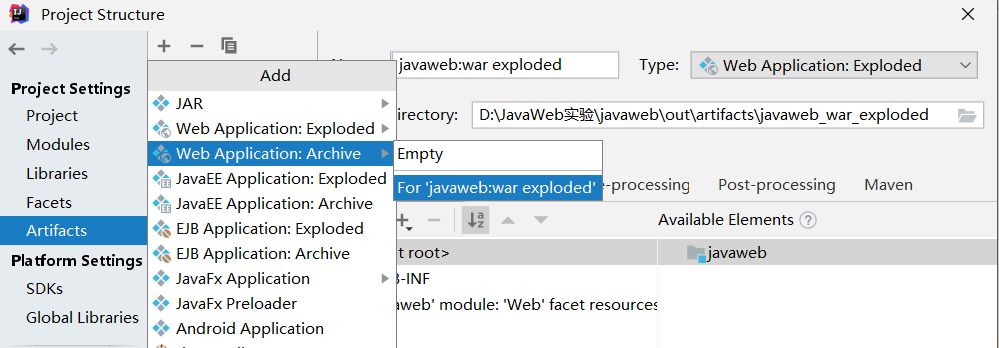


（六）启动Tomcat服务器（IDE方式）

（七）浏览JSP



（八）发布WAR包

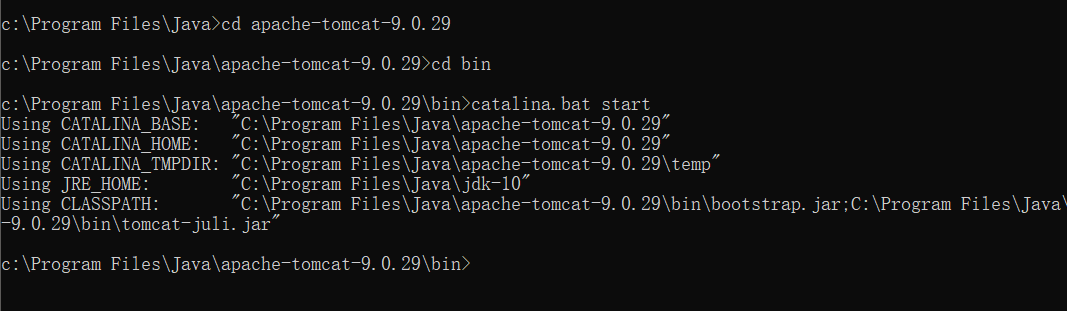


编译生成war包

访问之前设定的文件目录路径，可看到生成的war包文件。

将项目生成war包，放置tomcat/webapps下，

通过命令或tomcat控制台启动tomcat，访问项目。



访问地址：<http://localhost:8080/javaweb>（注意：先停止IDE的web服务）

（九）修改tomcat发布端口

将tomcat访问端口修改为8088，重新启动tomcat，访问[http://localhost:8088/javaweb](http://localhost:8088/javaweb/first.jsp)

修改文件为：tomcat/conf/server.xml

四、思考题

1、为什么访问项目时不需要输入全路径，如：<http://localhost:8080/javaweb/index.jsp>

答：URL引用路径中最后一个目录中的默认文件（通常对应于主页），这个文件常常被称为 index.html 或 default.htm。而我们war包在webapps中解压后的文件目录里，就是index.html文件

1. 如何将项目发布在其他目录下（非/tomcat/webapps目录）

答：首先，获取项目war包的路径地址（到上一级文件夹，不包含自己名称）；其次，需要将tomcat安装目录下的webapps文件夹中的docs、manage、ROOT文件夹复制拷贝到存放项目war包的文件夹中(不然无法正常启动tomcat)；然后，编辑tomcat安装目录下conf文件夹中server.xml文件，在文件底部有标签，修改其appBase路径——war包的路径地址；最后，重启tomcat，成功后通过localhost:+端口号+项目名即可访问项目，localhost:+端口号访问tomcat主页。

实验二 利用Github发布个人博客

一、实验目的

1. 了解github基本使用方式；

2. 初步掌握利用HTML进行网页的设计方法；

3. 掌握web项目中相对路径和绝对路径概念；

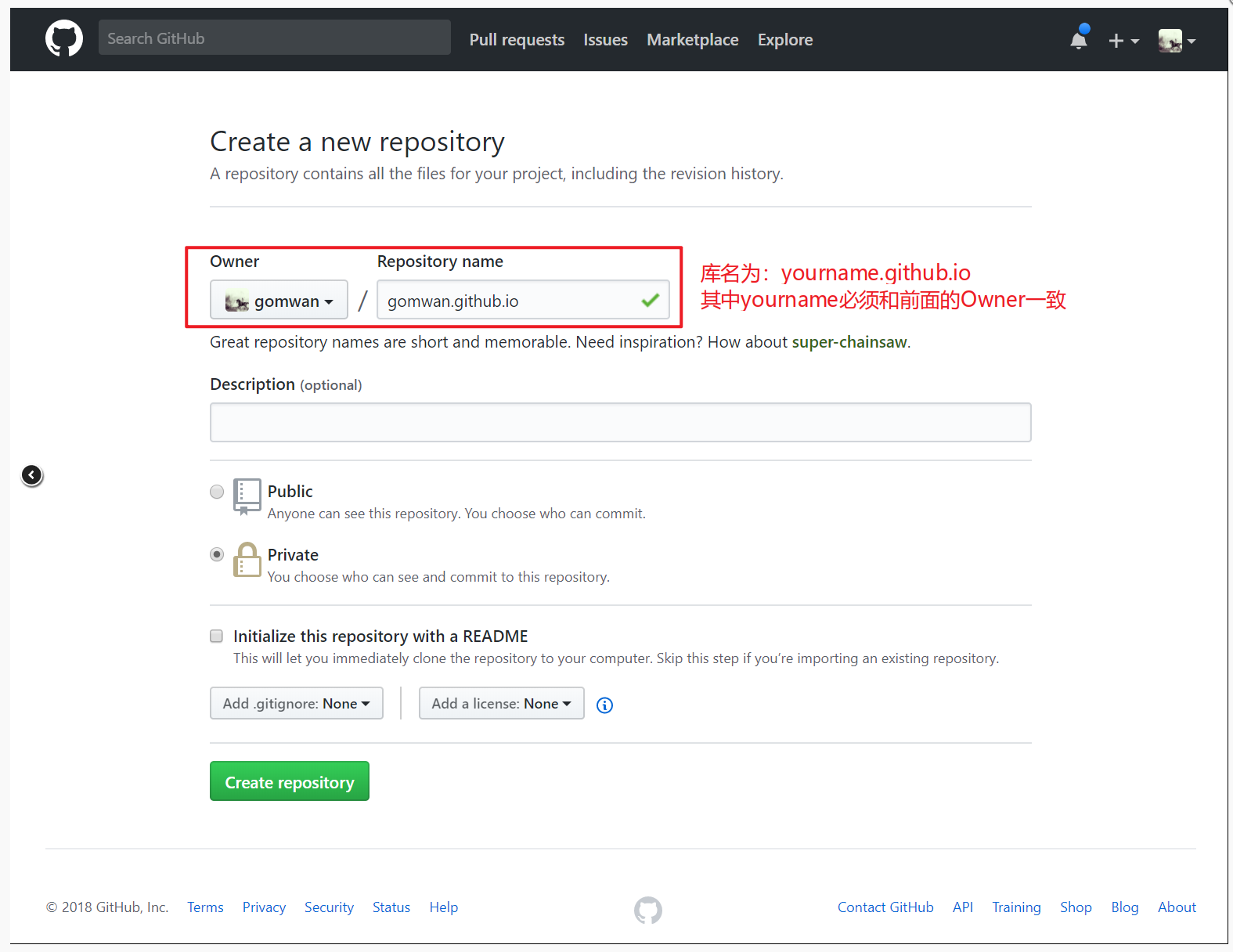
二、实验内容与步骤

1、注册github

<https://github.com>

2、配置github.io

在自己的主页，点击”New repository”，即可新建一个版本库



参考：

<https://gomwan.github.io/2018/03/11/Github%E6%B3%A8%E5%86%8C%E4%B8%8E%E9%85%8D%E7%BD%AE/>

1. 发布个人播客，可使用任意方式完成实验要求（不限定为以下的两种方式）

3.1使用静态模版框架

参考：

<https://gomwan.github.io/2018/03/11/hexo%E9%85%8D%E7%BD%AE/>

3.2 本地建立html静态页面，然后通过命令行、工具或网页方式上传至自己的github.io库中

三、实验要求

1、在发布的博客页面中显示至少两张图片

2、图片要求为其中一张引用方式为绝对路径（http://xxxx.com/1.png），另一张为相对路径（../image/1.png）

四、思考题

1. 在页面中是否可以引入本地绝对路径图片，如：d:/images/1.png？及原因

答：不能，因为GitHub的博客是托管在GitHub的服务器上的，所以使用本地的绝对路径的时候是引用的该服务器的资源。先不论我们的页面有没有权限访问，该服务器上也不存在这个图片文件。

1. 由于github.io为静态页面托管，如何为自己的博客添加评论功能？

答：(1)$npm install gitment --save  #安装gitment

(2)然后在 https://github.com/settings/applications/new 进行注册，获取Client ID和Client Secret

(3)打开themes/yilia目录下的\_config.yml文件进行修改并保存：

gitment\_owner:改成自己的github id

gitment\_repo:放在存储评论的repo

Gitment\_oauth:

client\_id:填写client\_id

Client\_secret:填写client密码

(4)再次生成网站和提交网站:$hexo g $hexo d

实验三 Serlvet技术

一、实验目的

1.掌握利用Eclipse、MyEclipse、IDEA创建动态web项目的方法

2.学会手工方式进行Servlet程序的编写、编译、部署、访问方法

3.学会在Eclipse、MyEclipse、IDEA环境下进行Servlet程序的编写、编译、部署、访问方法

4.掌握在web.xml中编辑servlet和servlet-mapping元素的方法

5.掌握在Tomcat服务器中发布web程序的方法。

二、实验项目内容

1、软件准备：

1）安装JDK

2）设置环境变量JAVA\_HOME。

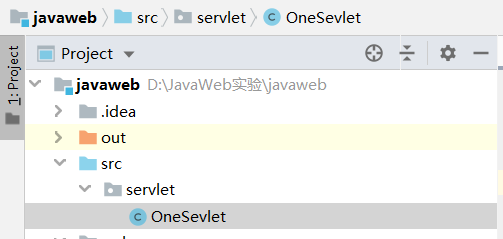
3）解压Eclipse、MyEclipse、IDEA

4）解压tomacat

2、手工方式进行Servlet程序的编写、编译、部署、访问方法

1）启动Eclipse、MyEclipse、IDEA，创建一个web项目，并指定Server Runtime Environments（若不存在，则创建Tomcat服务器）。

2）在所建项目中建立Servlet程序。



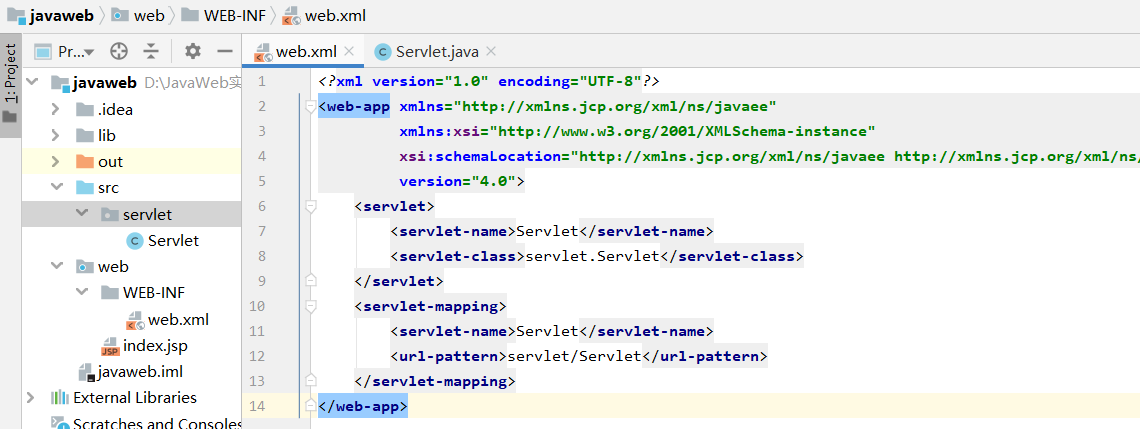
3）在Tomcat服务器上部署web程序。

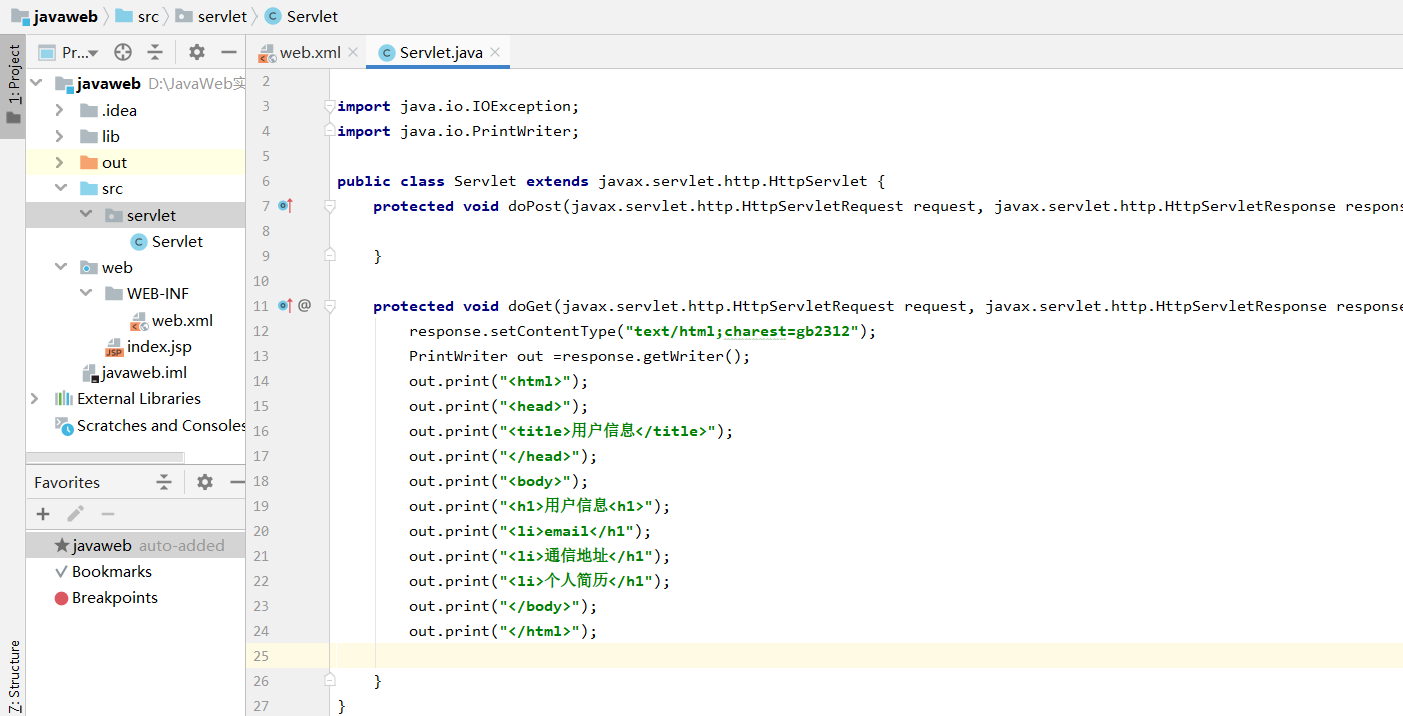
4）编写程序（选下列问题之一完成）

5）编写一个使用doGet方法的servlet，返回一个HTML文档，文档内容包括用户名、Email、通信地址及个人简历，这些内容由一个HTML页面提交。

6）编写一段HTML代码，用来创建一个表单，该表单用于收集用户姓名和电话号码。电话号码的格式必须为ddd-dddd-dddd。编写一个servlet，用来检查被提交的电话号码的格式是否符合格式要求，将是否正确的信息返回给用户。

7）编写一段HTML代码，以提交用户个人兴趣爱好调查信息，编写一个Servlet，对兴趣爱好进行统计分析，并显示统计分析结果。





运行效果显示：



## 实验四 Servlet容器模型

一、实验目的

1. 掌握Web应用程序的DD文件web.xml的配置方法和作用；

2. 掌握如何通过ServletConfig获取Servlet的初始化参数；

3. 掌握ServletContext的概念，如何获得ServletContext对象；

4. 掌握如何通过RequestDispatcher转发请求。

二、实验原理

Web应用程序有严格定义的目录结构，它需要一个称为部署描述文件的文件web.xml来配置。对应用程序的每个Servlet，可以在web.xml中定义若干个初始化参数，然后通过ServletConfig对象获取这些参数。

ServletContext对象表示该应用程序的环境，通过该对象可以获得应用程序范围的初始化参数、容器版本信息、资源的URL、在其中存放共享数据以及登录日志等。

三、实验内容与步骤

（一）建立一个名为bookstore的Web应用程序

在Tomcat的webapps目录下，建立一个如下所示的目录结构：

|- bookstore

|- html (包含所有的HTML文件)

|- jsp (包含所有的JSP文件)

|- images (包含所有的GIF、JPEG和BMP图象文件)

|- index.html (默认的HTML文件)

|- WEB-INF

|-classes

├ lib

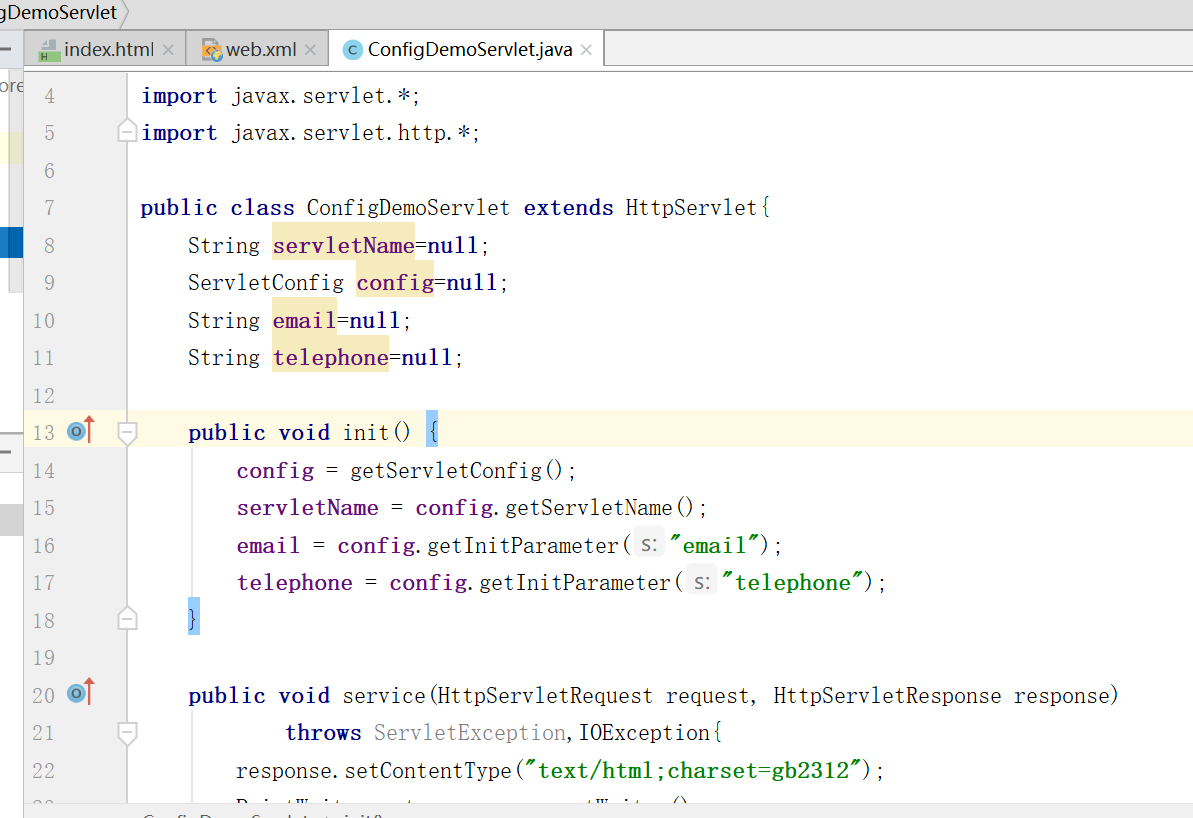
├ web.xml

图4.1 bookstore应用程序的目录结构

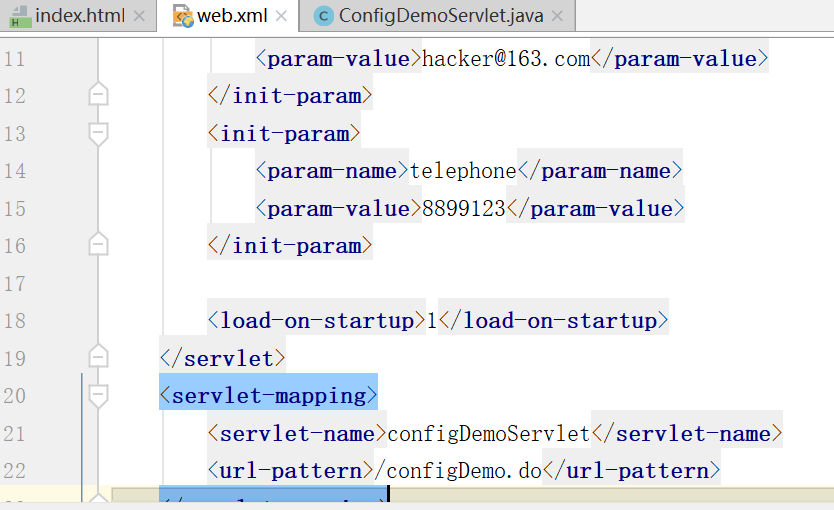
这样就建立了一个名为bookstore的Web应用程序，本实验后面的题目都在该应用程序下面完成。

（二）使用ServletConfig检索Servlet的初始化参数

【步骤1】编写ConfigDemoServlet程序，内容如下：



【步骤2】修改web.xml文件，在其中添加下面Servlet定义和初始化参数定义

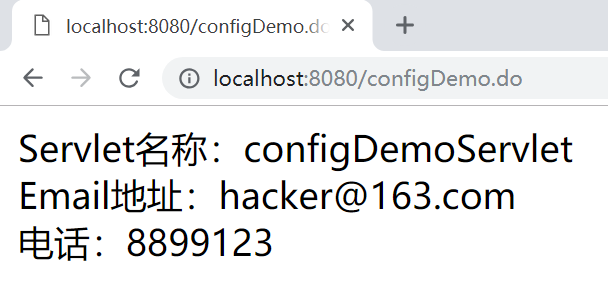
【步骤3】在浏览器的地址栏中输入下面URL访问该Servlet：

http://localhost:8080/bookstore/configDemo.do

程序运行结果如下图所示：



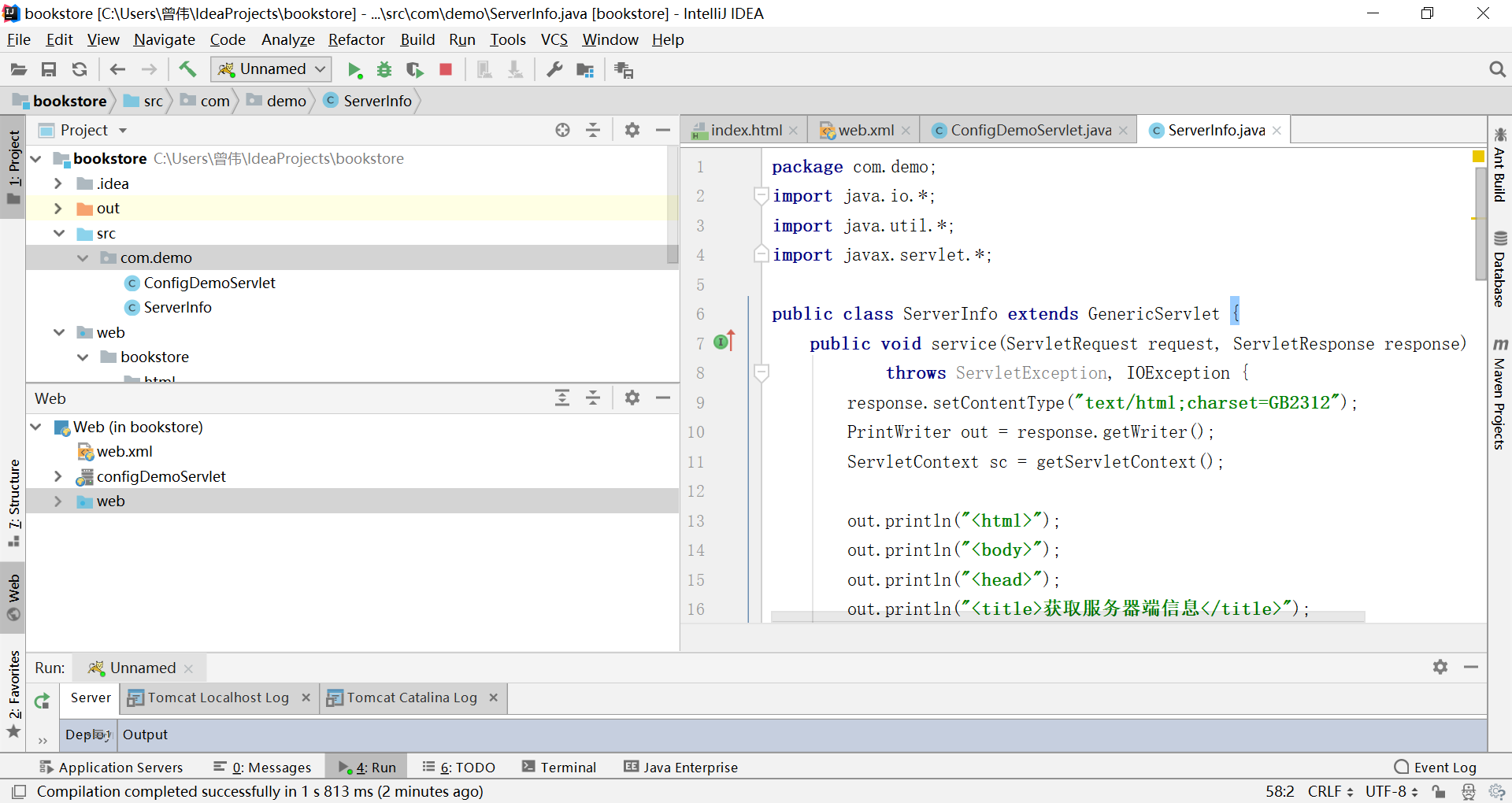
ConfigDemoServlet程序的运行结果

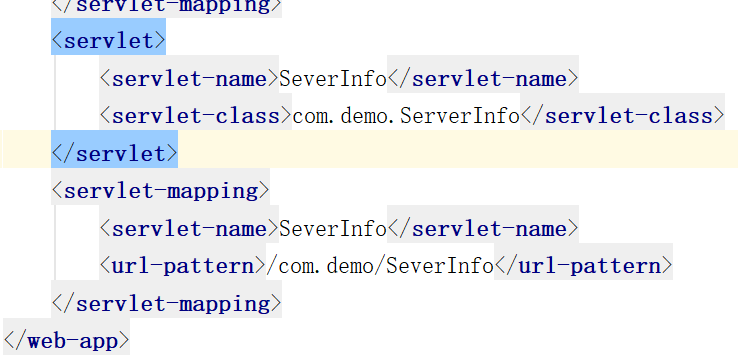


（三）ServletContext的使用

修改上面程序，将email地址和电话使用ServletContext初始化参数传递给ConfigDemoServlet。使用ServletContext对象将email地址和电话信息写入日志文件中，并查看日志文件。

（五）下面是一个Servlet，它可以获得有关服务器的信息。





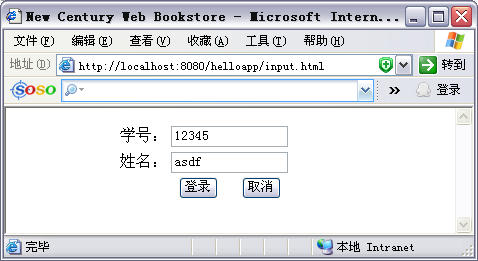
（六）简单的综合应用

【步骤1】创建一个名为input.html的HTML页面，其中包括一个表单，表单中包含两个文本域，分别供用户输入学号和姓名，该页面也包含提交和重置按钮。

【步骤2】定义一个名为com.demo.Student类，其中包括学号sno和姓名name两个private的成员变量，定义访问和修改sno和name的方法。

【步骤3】编写名为FirstServlet的Servlet，要求当用户在input.html中输入信息后点击“提交”按钮，请求FirstServlet对其处理。在FirstServlet中使用表单传递的参数（学号和姓名）创建一个Student对象并将其作为属性存储在ServletContext对象中，然后获得通过ServletContext的getRequestDispatcher()方法获得RequestDispatcher()对象，将请求转发到SecondServlet。

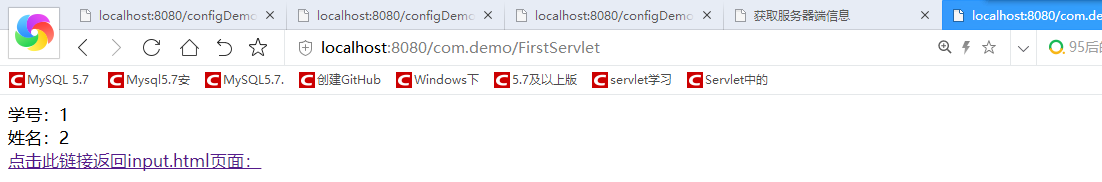
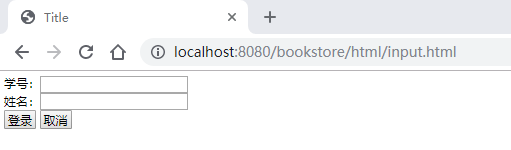
【步骤4】在SecondServlet中取出ServletContext上存储的Student对象，并显示输出该学生的学号和姓名。在SecondServlet的输出中应该包含一个超链接，点击该连接可以返回input.html页面。



input.html页面显示结果



SecondServlet显示结果



四、思考题

1. 如何通过ServletConfig获得Servlet参数？

答：通过ServletConfig接口中的方法获取参数，通过getServletName()获取Servlet的实例名，通过getInitParameter(“name”)获取name的初始化参数。

1. 如何获得ServletContext对象，如何通过该对象获得上下文参数？

答：通过ServletConfig接口中的方法获取参数，通过getServletName()获取Servlet的实例名，通过getInitParameter(“name”)获取name的初始化参数。

1. 如何使用ServletContext共享数据？

答：首先将数据作为属性通过setAttribute("name",name)方法存入ServletContext对象中，然后如果在其他地方使用，则需要从ServletContext对象中通过getAttribute("name")方法取出。

## 实验五 会话管理

一、实验目的

1. 了解Web服务器对客户跟踪的各种方法；

2. 重点掌握使用HttpSession对象跟踪会话的方法；

3. 掌握使用Cookie技术跟踪会话的方法；

4. 了解URL重写和隐藏表单域的方法。

二、实验原理

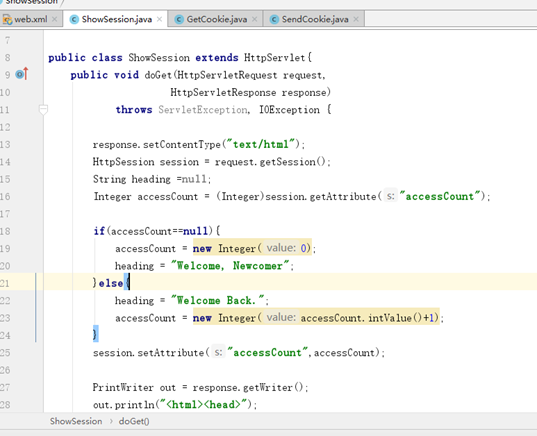
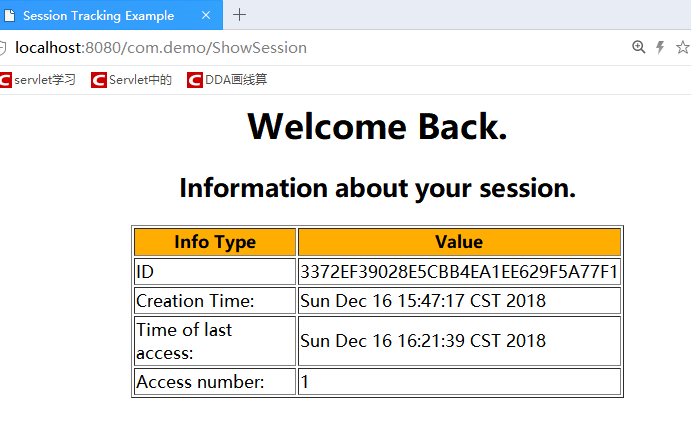
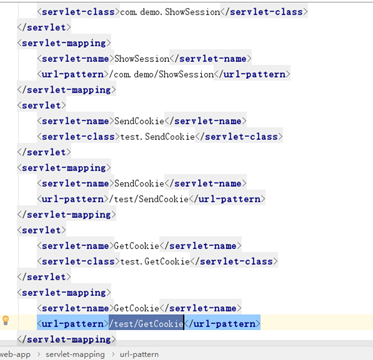
HTTP协议是无状态的协议。在很多情况下，Web服务器必须能够跟踪客户的状态。比如，对于一个购物网站，在一个时刻可能有多个客户购物，Web服务器必须能够区分不同的客户。一般情况下，Web服务器为每个客户配置了虚拟的购物车（ShoppingCart）。当某个客户请求将一个商品放入购物车时，Web服务器必须根据发出请求的客户的身份，找到该客户的购物车，然后把商品放入其中。

Web服务器跟踪客户的状态通常有4种方法：

（1）使用HttpSession对象管理会话；（2）使用持久的Cookie对象；（3）使用URL重写机制；（4）使用隐藏的表单域。

三、实验内容与步骤

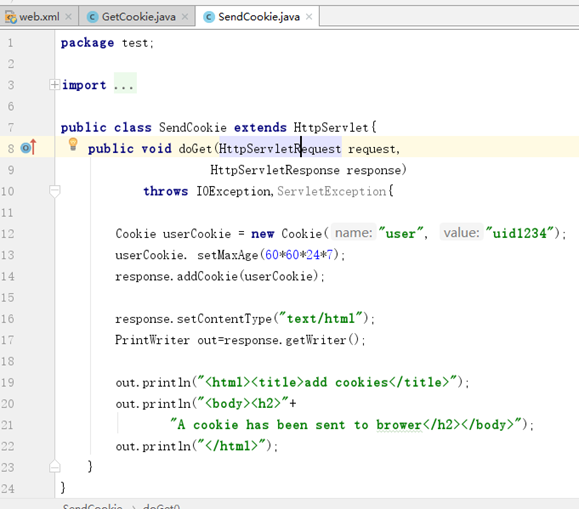
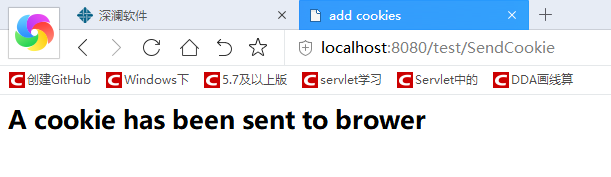
（一） 使用HttpSession对象管理会话。

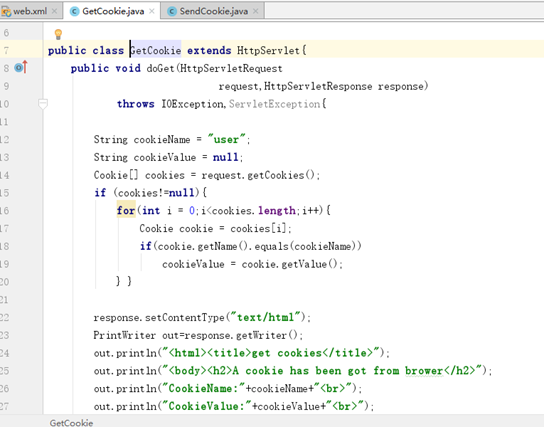
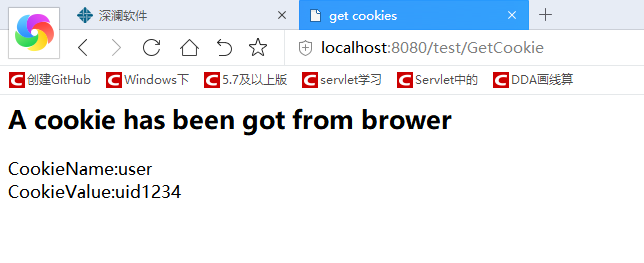
（二）使用Cookie跟踪会话

使用Cookie跟踪会话涉及到向客户端写入Cookie和从客户端读取Cookie。

【步骤1】向客户端写入Cookie对象

【步骤2】从客户端读取Cookie对象

四、思考题

1. 简述会话管理的常用方法有哪些？

答：使用Servlet API的Session机制，

使用持久的Cookie对象，

使用URL重写机制，

使用隐藏的表单域。

1. 如何使用HttpSession对象共享数据？

答： Public void setAttribute(String name,Object value)j将一个指定名称和值的属性绑定到会话对象上，Public void getAttribute(String name)返回绑定到会话上的指定名称属性值，Public void Enumeratinon getAttributeName()返回所用绑定到会话上的属性名的Enumeratinon对象，Public void removeAttribute(String name,)从会话中删除绑定的指定名称的属性。

实验六 JSP+JDBC开发

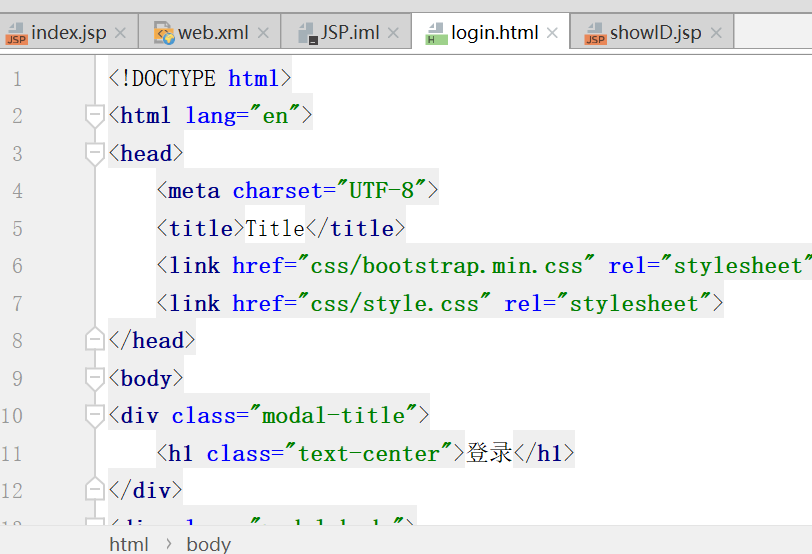
### 一、实验目的

综合使用之前学习的JSP、JDBC、Ajax等技术实现数据库查询功能。

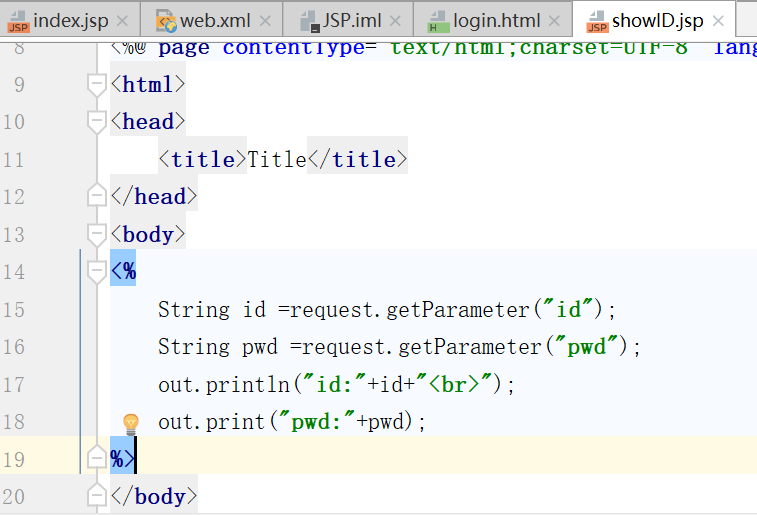
### 二、实验项目内容

* 新建web项目：
* web目录下新建项目所需文件夹
* 修改index.jsp样式

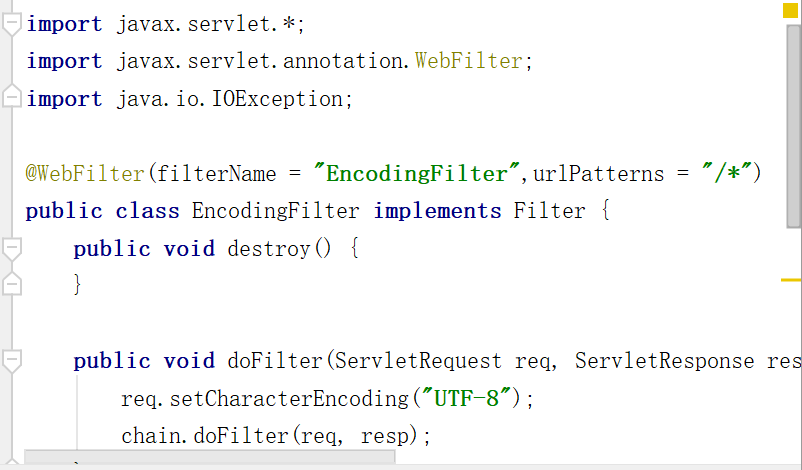




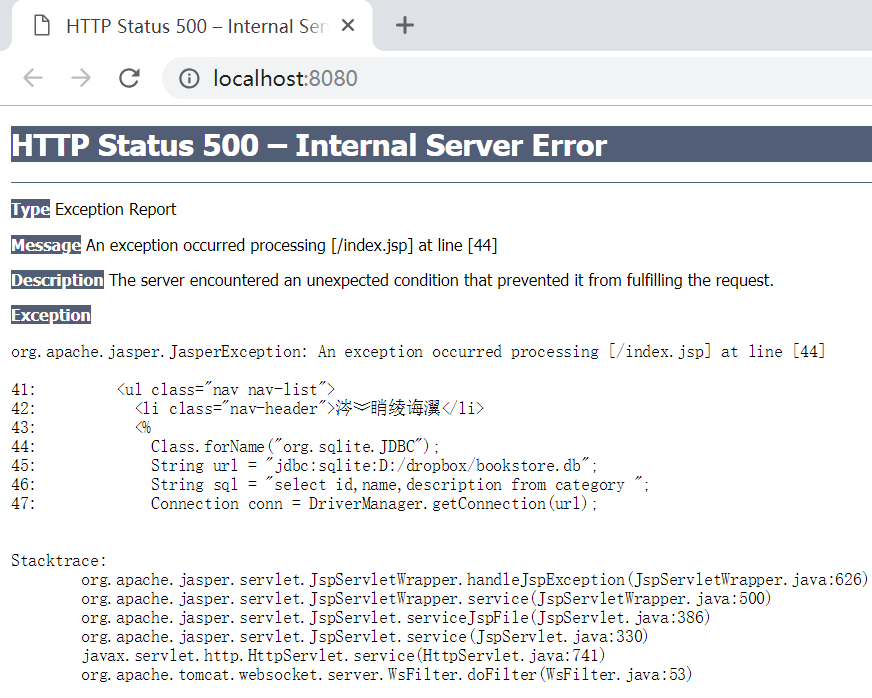
* 新建传值显示页面showID.jsp



* 添加Filter处理中文乱码问题



* 使用JSP+JDBC查询数据

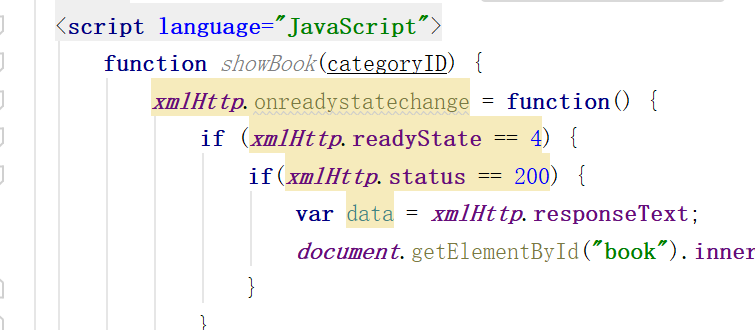


运行项目，访问项目地址



成功获取类别信息

* 添加ajax方法



* 修改左侧列表对应的链接



* 新建getBook.jsp接收参数并进行查询，在<body>添加相应代码



* 运行项目观察效果



### 三、思考题

1、<body onload="initAJAX()"> 中initAJAX()函数的作用？

答：实例化XMLHTTP对象，为防止IE浏览器内置对象，该对象具有异步提交数据和提交结果的功能。

2、如何使得书籍显示样式对齐？

答：display: -webkit-box; /\*\* 将对象作为伸缩盒子模型显示 \*\*/

-webkit-box-orient: vertical; /\*\* 设置或检索伸缩盒对象的子元素的排列方式 \*\*/

-webkit-line-clamp: 3; /\*\* 显示的行数 \*\*/

overflow: hidden; /\*\* 隐藏超出的内容 \*\*/

3、Ajax请求的为整个getBook.jsp页面数据，思考通过什么技术能够让数据请求更为优雅？

答：可以通过fetch技术来实现。

实验七 JSP+JavaBean+Servlet开发

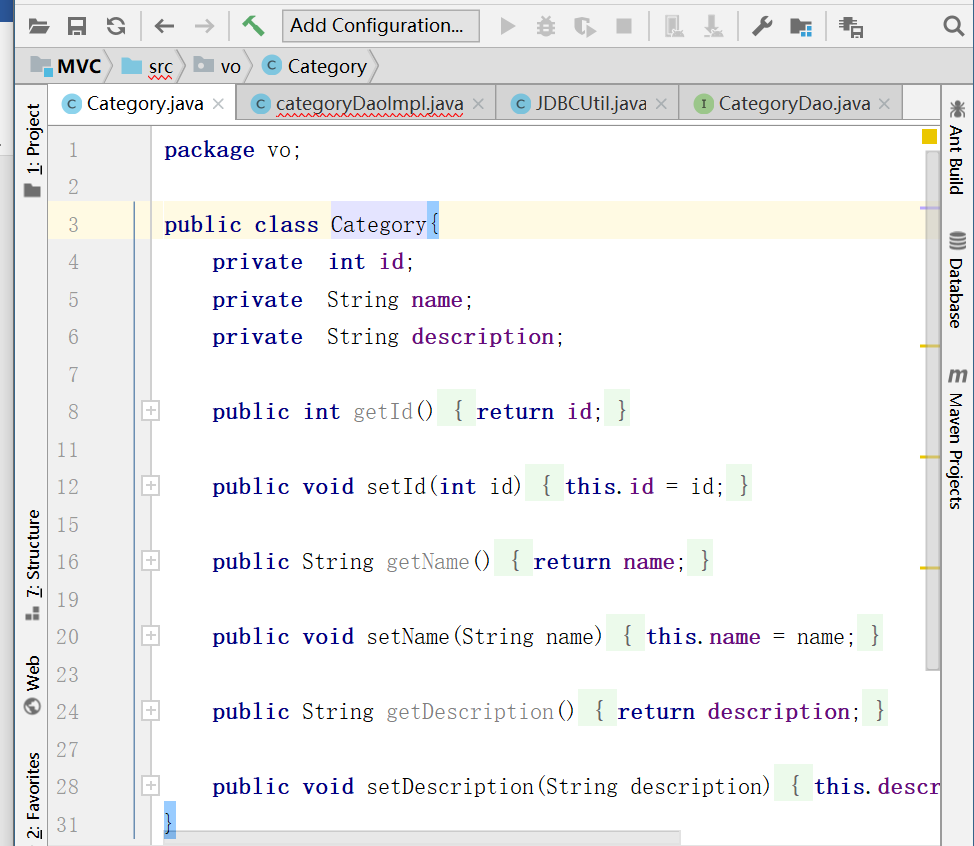
### 一、实验目的

基于MVC模式，使用JSP+JavaBean+Servlet+JDBC技术完成之前实验的功能呢。

### 二、实验项目内容

* 新建web项目：
* web目录下新建项目所需文件夹
* 创建category对应的vo
  + 先完成类别查询功能，先创建类别的vo

在vo下创建category类



根据表结构建立相应属性

自动生成Getter和Setter方法

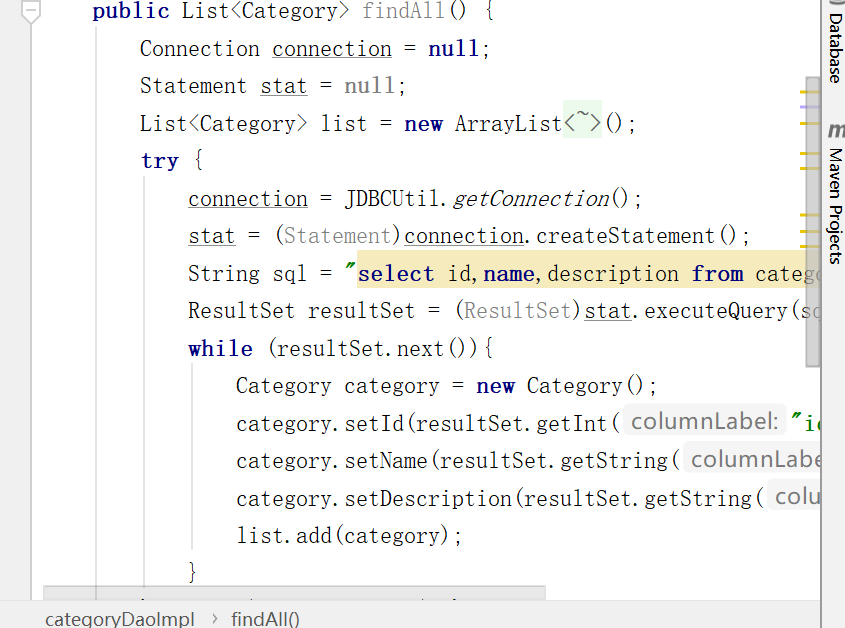
* 完成category对应的dao
  + 首先完成JDBC公共工具类，JDBCUtil



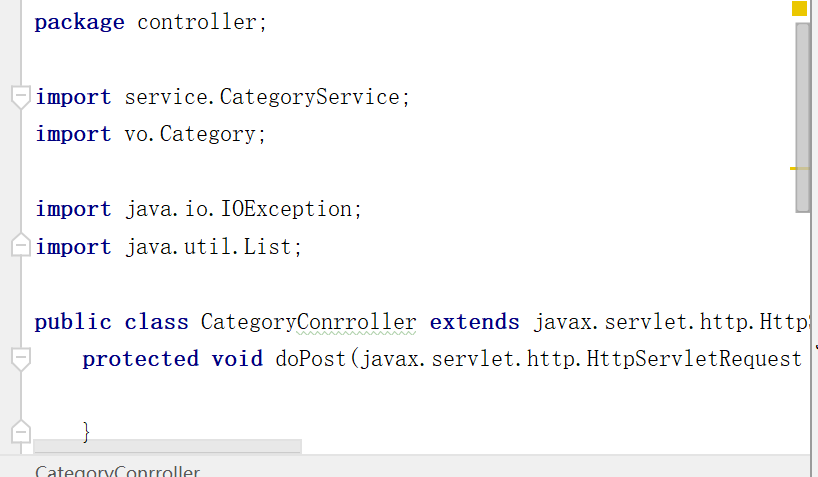
* + 完成categoryDao接口注意引入正确的包



* + 完成categoryDao实现类



* 完成category对应的service
* 实现controller功能



至此controller端完成

* view端开发
* category显示功能完成，开始进行book查询、显示功能开发
* 进行book查询功能的view开发

运行项目，访问地址：

<http://localhost:8080/category>

发现book显示正常，但是类别信息会消失

* 使用json传输数据
  + 导入jaskson文件夹中3个jar包
  + 修改BookController为json返回

运行项目访问地址

<http://localhost:8080/getBookByCategoryID?id=1>

* + 同样修改CategoryController代码

访问地址：

http://localhost:8080/category

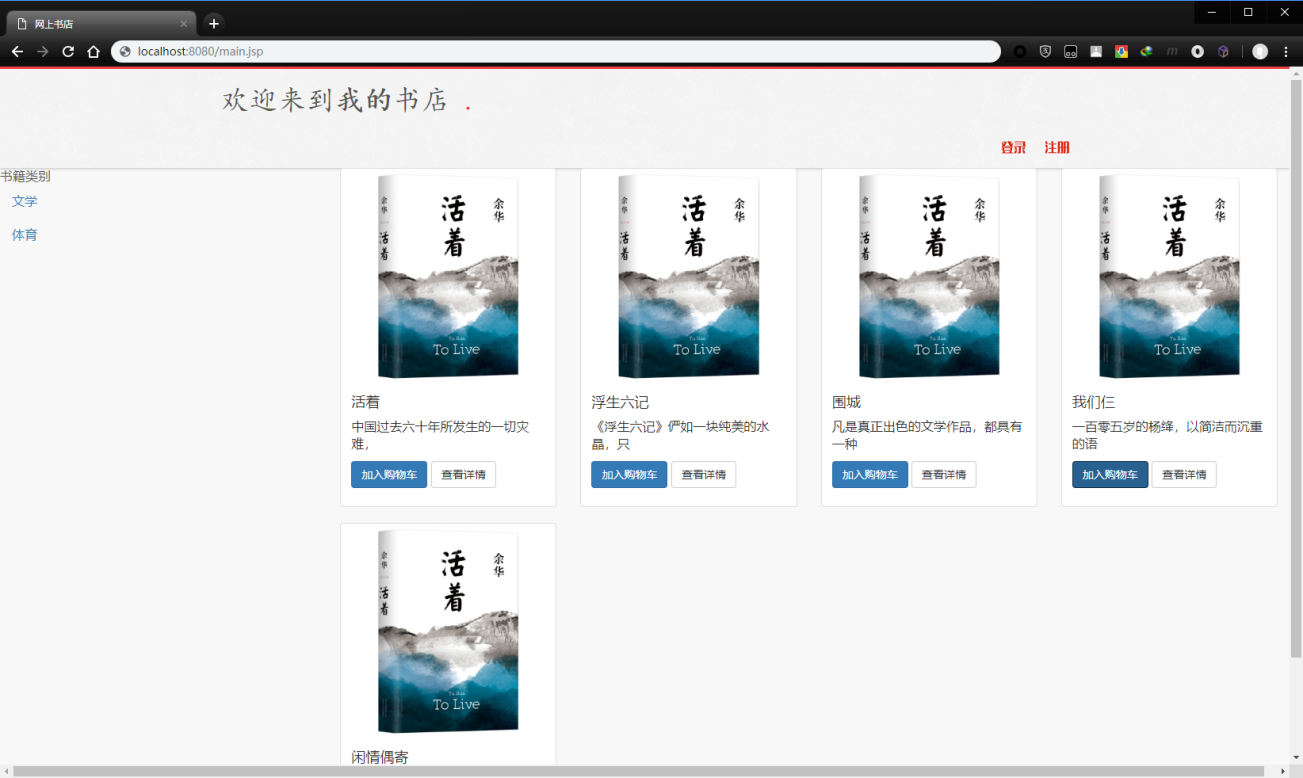
* + 修改main.jsp代码使用ajax方法请求数据



添加ajax函数获取书籍分类信息



访问项目地址：<http://localhost:8080/main.jsp>



### 三、思考题

1、JDBCUtil的作用，以及它好处和不足？

JDBC的优点：

直接底层操作，提供了很简单、便捷的访问数据库的方法，跨平台性比较强。灵活性比较强，可以写很复杂的SQL语句。

JDBC的缺点：

因为JAVA是面向对象的，JDBC没有做到使数据能够面向对象的编程，使程序员的思考仍停留在SQL语句上。

操作比较繁琐，很多代码需要重复写很多次。

如果遇到批量操作，频繁与数据库进行交互，容易造成效率的下降

2、在完成book功能查询和显示时发现类别信息会丢失，如何用最小代码修改量修复此bug？

答：因为在book的controller中没有向request中加入category的参数，可以在该controller中加入：

List<Category> list = categoryService.findAll();

         request.setAttribute("categories",list);

3、在controller中转换java对象为json数据格式的代码可否放到service中？为什么？

答：可以，但是相应的service层的返回值应该改成String类型。

实验八 Spring项目搭建

### 实验目的

### 使用Spring搭建项目框架，并熟悉SpringBean的属性。

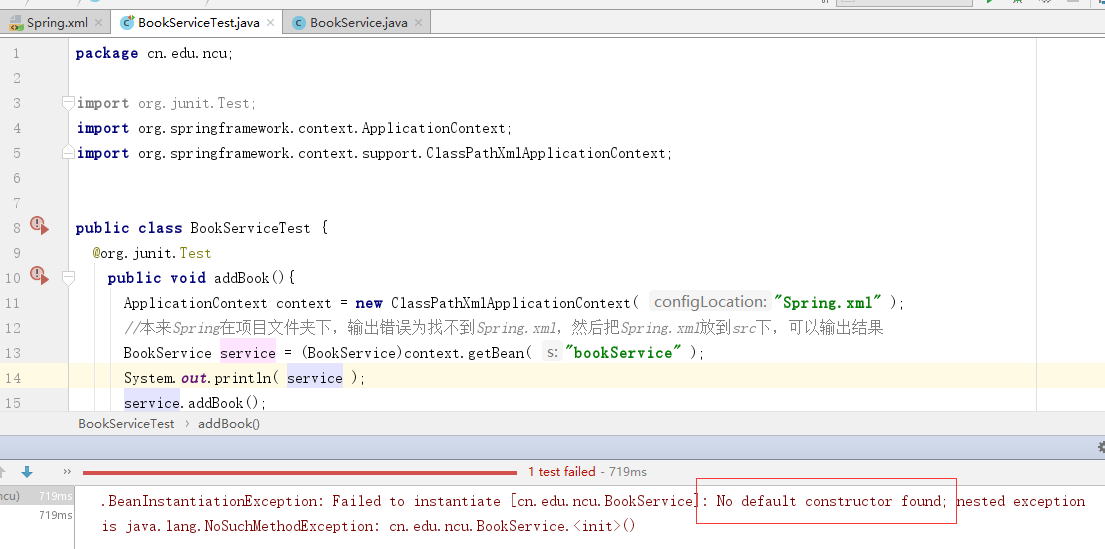
### 二、实验项目内容

* 1、Spring安装
* 2、Spring Bean管理

SpringBean三种实例化方式

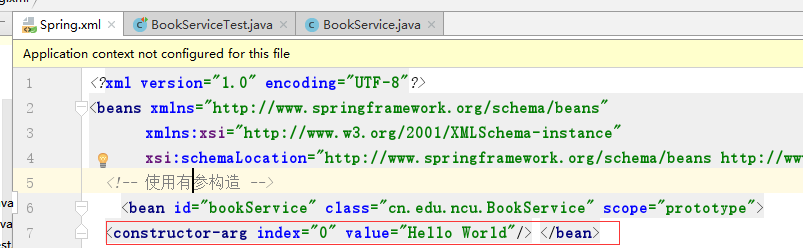
1. 使用无参数构造方法

在BookService类中添加一个带参数的构造方法，执行单元测试



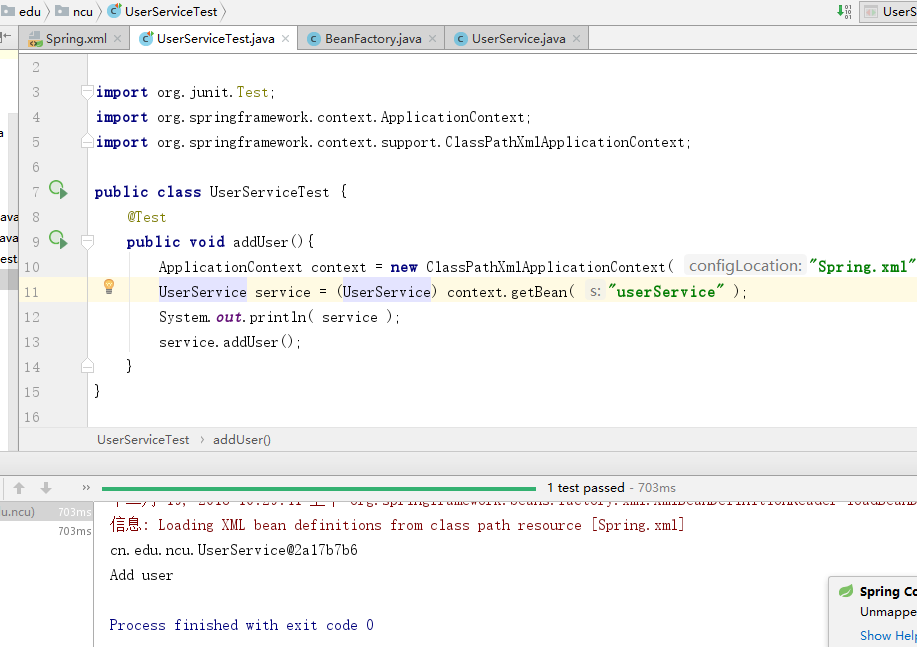
1.1有参构造

修改xml内容，指定构造函数参数



修改完xml文件之后，修改下BookService中的addBook方法，使其显示bookID的值;

1. 使用静态工厂创建



1. 使用实例工厂创建



3、Bean的作用范围

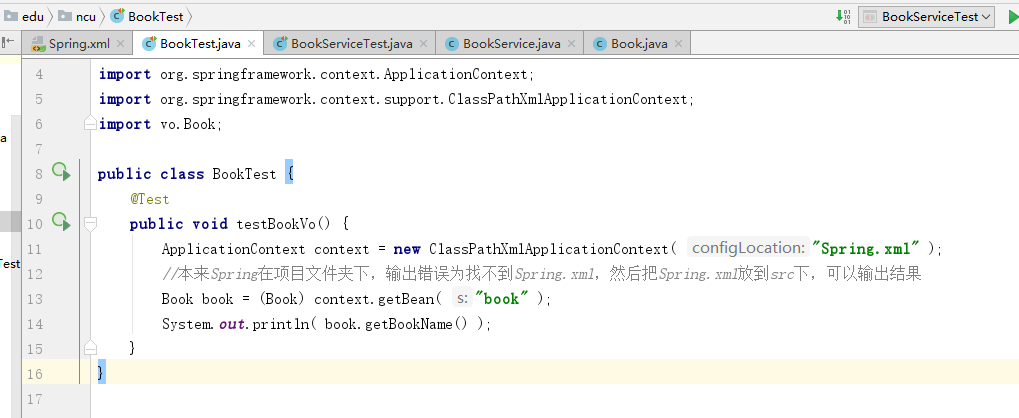
1. Singleton(单例)
2. prototype（多例）

* 4、Spring的属性注入

1. 有参构造

通过xml指定构造函数参数

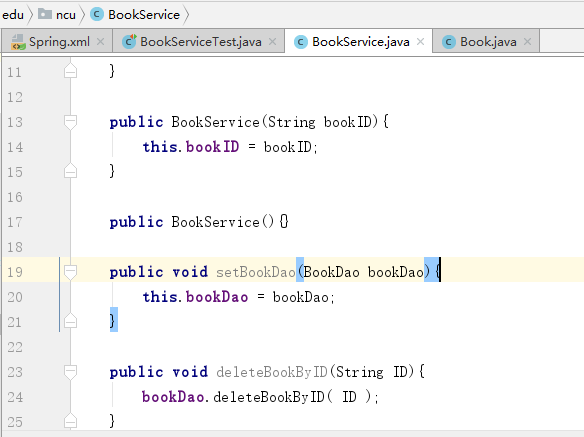
1. Setter方法



* 5、Spring Bean注入

实际项目中Controller调用Service，Service调用Dao等等，在代码中需要注入相应的bean，实现类之间的解耦。

1、setter方法注入

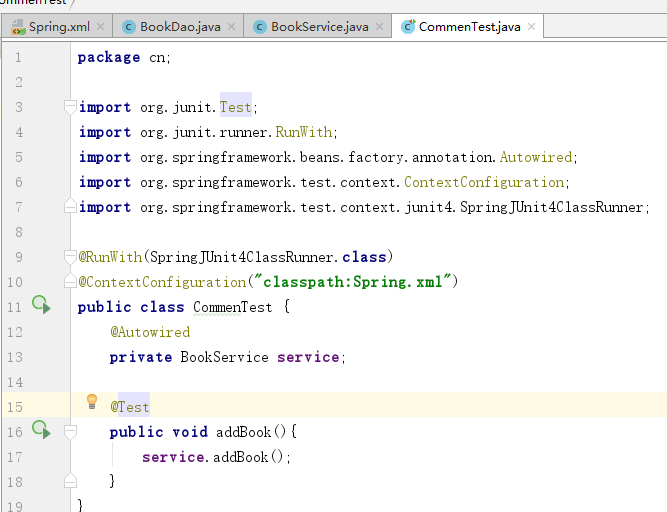


修改xml配置文件

先创建下面的bookDao的配置描述，再上方bookService中添加属性

注意：这里不再是一个字符串，而是对象，需要使用ref来指向一个对象

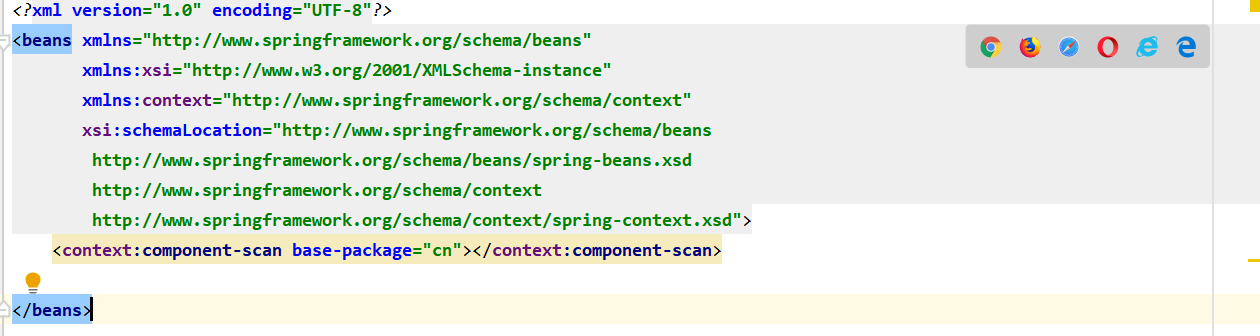
2、注解方法注入



三、思考题

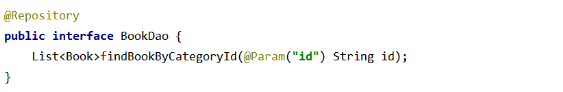
1、修改实验7中的项目使用spring框架，实现多个模块之间的解耦。

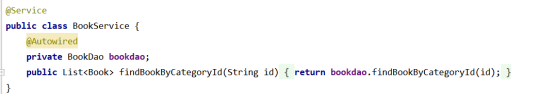
建立spring项目，配置自动扫描

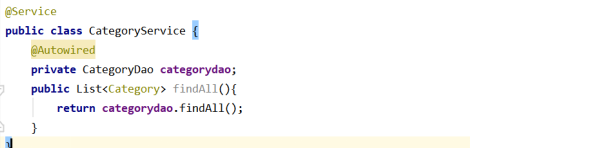


建立vo,Service,dao,controller层

在dao层中加入@Repository,代表这个类是处于数据访问层；在Service层加上@Service代表业务逻辑层并用@Autowired注入dao层，在controller层中加入@Controller代表业务调度层，利用Spring提供的注解进行解耦操作。







@Service, @Controller, @Repository是spring注解，注解后可以被spring框架所扫描并注入到spring容器来进行管理。

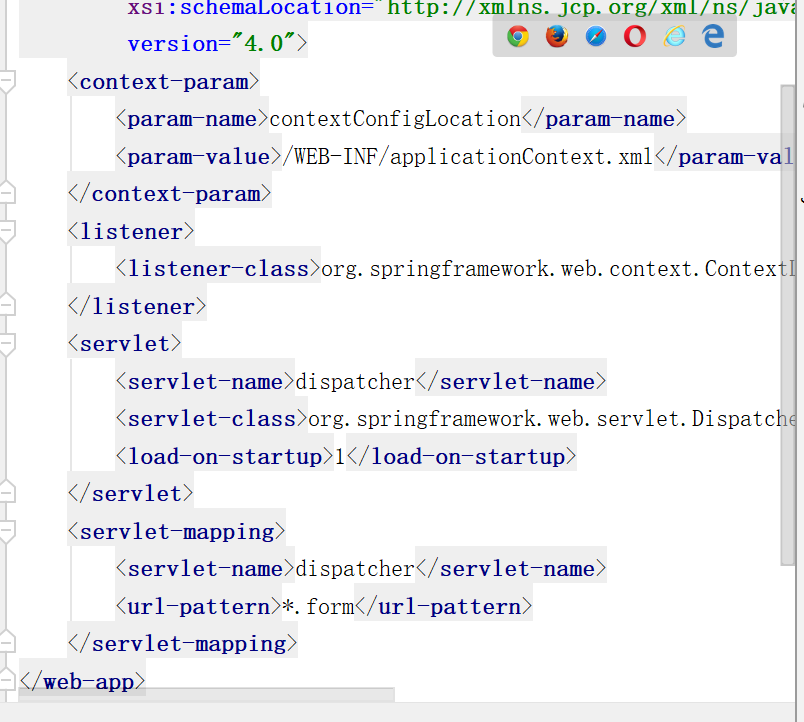
实验九 SpringMVC项目搭建

### 一、实验目的

使用SpringMVC搭建项目框架。

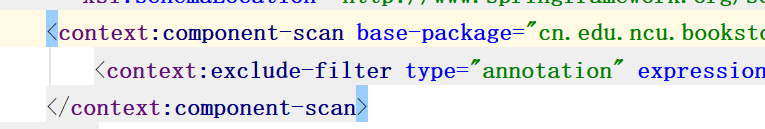
### 二、实验项目内容

* **1、SpringMVC安装**
* 2、修改配置文件

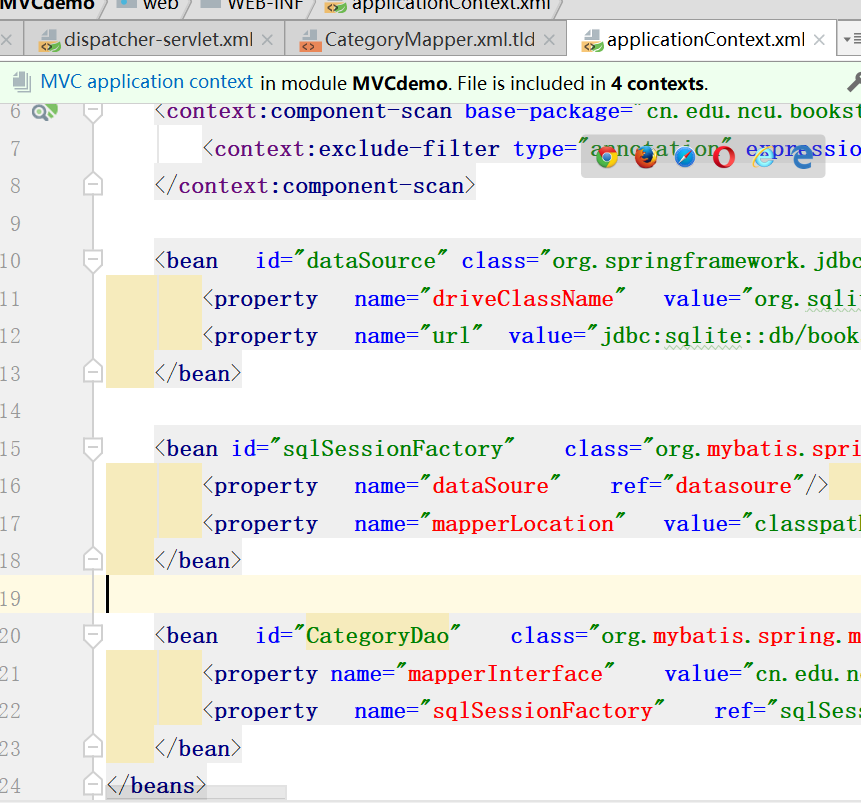


applicationContext.xml

设置自动扫描路径



* 3、配置MyBatis环境



* 4、Mapper（Dao）开发
* 5、Service开发
* 6、Service单元测试
* 7、Controller开发
* 8、view开发
* 9、修改Controller，使用modelandview
* 10、使用参数的查询

1. 新建Book实体
2. 新建Mapper(BookDao的接口)
3. 新建Mapper XML文件
4. 新建BookService
5. 新建BookController

* 11、使用json格式传递数据

使用@ ResponseBody处理数据

作用：

该注解用于将Controller的方法返回的对象，通过适当的HttpMessageConverter转换为指定格式后，写入到Response对象的body数据区。

使用时机：

返回的数据不是html标签的页面，而是其他某种格式的数据时（如json、xml等）使用

注：HttpMessageConverter 早期的SpringMVC需要通过配置SpringMVC的xml文件进行相应的映射

本例中的SpringMVC可以自动转JSON格式，无需配置，只需导入SpringMVC支持的json工具包

### 三、思考题

1、修改实验7和实验8中的项目使用springMVC框架。

分层创建文件并配置xml

配置applicationContext.xml

配置mapper映射文件

在controller层中利用ModelAndView跳转页面

完善view文件夹中的jsp文件，最后结果如下所示

## 实验十 SpringBoot框架使用

一、实验目的

1. 了解SpringBoot项目创建；

2. 了解JPA使用方法。

二、实验原理

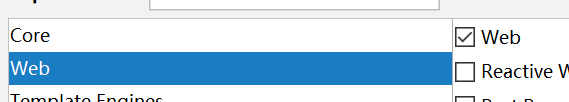
SpringBoot框架合成JPA（Java持久层API）操作Mysql数据库

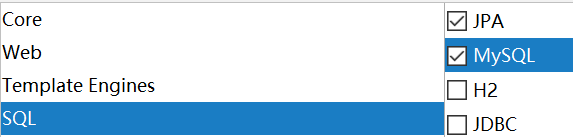
实验内容与步骤

创建SrpingBoot项目

通过IDEA创建Boot项目(File->New->Project)

选择需要的依赖：web、jpa、mysql





1. 未使用IDEA或MyEclipse 等IDE的同学，可以通过官方网站创建项目

- 打开URL： start.spring.io，同样输入项目相关信息及所需的依赖

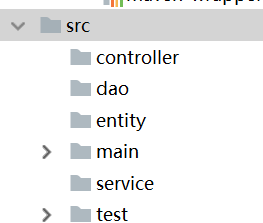
- 下载解压后通过Maven方式导入（由于项目为Maven项目，所以必须已经安装Maven环境）

1. 进行项目相关配置

修改配置文件类型为yml（可选）

配置项目相关信息：数据库连接信息、JPA SQL语句输出、日志打印级别；也可以在这设置项目端口、路径等

1. 编码实现
2. 创建相应的package：entity、dao、service、controller



1. 书籍分类查询功能

按照表结构创建分类entity，并通过idea自动生成getter、setter及toString方法类标记注解为@Entity，并在主键（@Id）和自增字段（@GeneratedValue）上添加相应的注解

创建dao接口，接口类 extends JpaRepository<T,ID>，T和ID分别为相应的实体和主键类型

创建相应的service

创建service的单元测试方法

运行测试方法，观察日志输入是否正常（可先在数据库表中添加测试数据）

创建controller类（\*\*BookCategoryController\*\*）及方法（\*\*findAl\*\*l）

运行整个项目，并使用postman访问（get方法也可以通过浏览器访问）

Postman获取方式：(https://www.getpostman.com/ 或者 添加chrome插件方式获取)

1. insert功能

简单的CRUD功能JPA已经提供，不需要手工完成dao的实现类

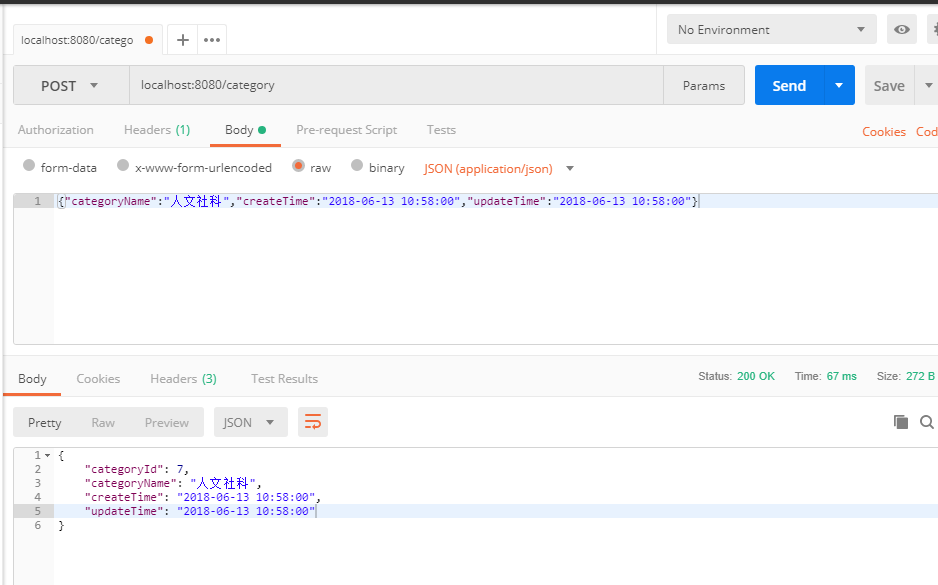
dao不用修改，直接在service（\*\*BookServiceService\*\*）中添加插入功能（\*\*saveBookCategory\*\*）

创建测试方法

因为JPA是懒加载，所以需要在测试类上加上注解@Transactional

执行测试，查看结果（由于添加@Transactional 之后在测试类中并不会commit，所以数据库中不会有新增记录）

- 添加contorller方法



1. 手动映射及SQL

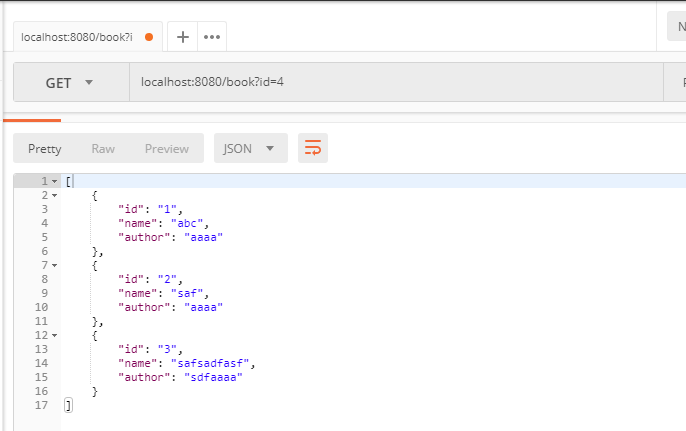
1. 表和字段

- 建立一个非驼峰风格表

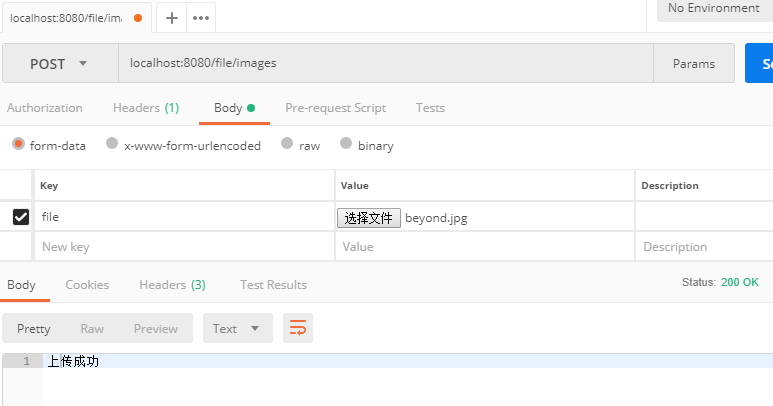
进行单元测试，观察输出结果是否正确。

\*\*自己创建controller类及相应方法，并使用Postman或浏览器访问路径，并打印结果\*\*

@RestController  
@RequestMapping(value = **"book"**)  
**public class** BookController {  
 @Autowired  
 **private** BookService bookService;  
  
 @GetMapping(**""**)  
 **public** List<Book> query(@Param(**"id"**) String id) {  
 List<Book> books = bookService.queryBook(id);  
 **return** books;  
 }  
}



1. 文件上传
2. 建立上传文件夹
3. 在controller package下创建 UploadFileController类
4. 在controller package下创建 UploadFileController类
5. 也可以启动Nginx或Apache等Web服务器，编写html页面进行测试（也可以直接打开）



四、思考题

1. **JPA如何将post中的数据存入数据库的？**

答：需要首先建立java bean到关系数据库的映射关系（使用[java注释](https://www.baidu.com/s?wd=java%E6%B3%A8%E9%87%8A&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)或xml），然后新建/修改java bean实例后调用jpa相关的接口实现对象的持久化，相关的sql操作由jpa底层自动完成。

2. 使用JPA和Mybais的区别

答：JPA是对象与对象之间的映射，而Mybatis是对象和结果集的映射。

JPA的可移植性比Mybatis的好。

实验十一 会议报名通用系统

1. 实验目的

运用所学的JavaWeb知识设计一个会议报名通用系统。掌握设计实现运行web程序的能力。

1. 实验项目名称

会议报名通用系统:技术要求：B/S架构，JSP，框架自选，必须支持WEB（微信小程序可自选且不加减分），必须考虑会议分发情况例如：生成唯一会议号，生成会议号二维码（可以加分5%-10%）。

1. 实验步骤

1、需求分析

会议组织者：

(1)会议组织者登录进入系统，

(2).填写会议的相关信息（时间，地点，人物，宾馆，等等），

(3)选取需要参会人员填写的信息项(姓名，工作单位，身份证号，电话，参会时间，性别，是否需要安排房间，等等)这些项目组织者可以根据需要勾选，

(4)自动生成会议唯一编号,

(5)可以以把本次参会的人员信息下载生成EXCEL文件，

会议参加人员:

登录方式:通过注册账号和密码登入会议通用报名系统，填写信息，提交。

系统管理员:

可以看见会议报名通用系统中组织者以及发起的会议并有权删除会议。

2、数据库设计

2.1会议基本信息表 huiyi

meetingname time place hotel holder number creater

其ER图如下：



2.2参会人员信息表

会议号 姓名 工作单位 身份证号 电话 参会时间 性别 是否安排房间 会议名称

ER图如下：



2.3参与者账号表

jaccount password

2.4组织者账号表

laccount passord

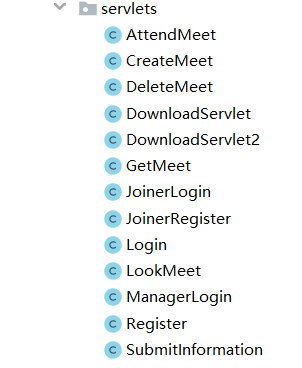
2.5管理员账号表

maccount password

3、架构设计

3.1会议组织者、参与者和管理员可以注册账号后可以登陆系统进行角色相应的操作。相应的操作用servlet处理。

项目类和servlet目录如下：



3.2类的设计及其作用：

Huiyi和HuiyiDao用来对会议基本信息表进行查询，获取会议信息

Joiner和JoinerDao用来对参与者登陆进行验证；Laucher和LaucherDao用来对组织者登陆进行验证；Manager和ManagerDao用来对管理员登陆进行验证

CommonExcel是一个通用的导出excel文件的类：用于组织者下载时导出相应的参会人员的信息并形成excle文件供组织者下载。

Servlet的设计及其作用：

AttendMeet:参与者参加会议；CreateMeet:组织者创建会议：DeleteMeet:管理员删除会议；SubmitInformation：提交参与者填写的想要参加的会议要求填写的个人信息。

DownloadServlet用于处理组织者下载会议，Download2用于处理组织者创建会议后下载会议。

GetMeet用于获取当前会议报名通用系统中所有存在的会议。

JoinerLogin用于验证参与者登陆。JoinerRegister用于参与者创建账号。；Login用于验证组织者登陆。Register用于组织者注册账号

管理员账号与密码在数据库中已经设置，故不需要注册即可登陆。

4、功能实现

4.1组织者模块：

会议组织者登陆与注册功能实现：通过LauncherDao类将Launcher类实例化为ArrayList，实例化得到包含所有组织者账号密码信息的ArrayList类型的对象lauchers，遍历获取lauchers中的组织者对象，判断用户输入的账号密码是否与lachers中的组织者的账号密码相等，如果相等即组织者已经注册，输出查看“我的会议”和“创建我的会议”供组织者用户选择；若组织者用户未注册则输出提示去注册。

关键代码：

int d=0;String account = request.getParameter("account");  
 System.out.println(account);String password = request.getParameter("pwd");  
 HttpSession session=request.getSession();  
 session.setAttribute("account",account); LauncherDao launcherDao=new LauncherDao();  
 ArrayList launchers= null; try {launchers = launcherDao.queryAllLaunchers();  
 } catch (Exception e) { e.printStackTrace();}  
 for(int i=0;i<launchers.size();i++){Launcher launcher=(Launcher) launchers.get(i);  
 if(launcher.getAt().equals(account)&&launcher.getPd().equals(password)) {  
 d=1;break; }}  
 if(d==1){out.print("<a href=\"LookMeet?account=" +account+ "\">查看我的会议</a>"+"<br>");  
 out.println("<a href=\"http://localhost:8080/testjdbc4\_war\_exploded/CreateMeet.html\">创建我的会议</a>");  
 }else{ out.println("对不起，查无此用户！"+"

<a href=\"http://localhost:8080/testjdbc4\_war\_exploded/Register.html\">去注册</a>"); }}

注册：int i= 0;System.out.println(sql);i = stat.executeUpdate(sql);  
out.println("成功创建"+i+"位组织者用户！"+ "<br>"+"<a href=\"http://localhost:8080/testjdbc4\_war\_exploded/Launcher.html\">去登陆</a>");

组织者查看自己创建的会议功能的实现：通过sql语句在包含会议信息数据库表格huiyi中查询creater列的值等于组织者账号的数据项，将这些数据项信息输出显示并提供下载会议参与人员相关信息表格的超链接。

关键代码：

HttpSession session=request.getSession();  
String id=(String) session.getAttribute("id");System.out.println("session:"+id);  
String account=(String) session.getAttribute("account");System.out.println(account);  
try {Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver"); Connection conn = DriverManager.getConnection("jdbc:sqlserver://localhost:1433;DatabaseName=meeting;user=sa;password=notbad");Statement stat= conn.createStatement();  
 String sql="select \*from huiyi where huiyi.creater ='"+account+"'";  
 System.out.println(sql);ResultSet rs=stat.executeQuery(sql);  
 while(rs.next()) {  
 out.println("会议名称:" + rs.getString("meetingname")+"<BR>");  
 out.println("时间:" + rs.getString("time")+"<BR>");  
 out.println("地点:" + rs.getString("place")+"<BR>");  
 out.println("宾馆:" + rs.getString("hotel")+"<BR>");  
 out.println("主讲人:" + rs.getString("holder")+"<BR>");  
 out.println("会议编号：" + rs.getString("number")+"<BR>");  
 out.print("<a href=\"DownloadServlet2?id=" +id+ "\">下载此会议</a><br>");  
 }rs.close(); stat.close(); conn.close();  
} catch (ClassNotFoundException | SQLException e) { e.printStackTrace();}

组织者创建会议功能的实现：用字符串数组check获取组织者勾选的会议要求的信息，初始化字符串数组index为{"jname","workplace","idcard","phonenumber","attendtime","sex","room"}。当index中的元素能在check中找到时，将此元素置为“是”，代表此项信息需要填写，将index中剩下的不能在check中找到的元素置为“否”，代表此项信息不需要填写。将处理好index元素依次作为会议信息的是否需要填写的个人信息数据与会议号和会议名称一同用sql语句插入数据库中的人员信息表。插入数据成功后，输出刚创建的会议信息和查看已经创建的会议的超链接。

PrintWriter out = response.getWriter();  
HttpSession session=request.getSession();String account=(String) session.getAttribute("account");  
String meetingname = request.getParameter("title");System.out.println(meetingname);  
String time = request.getParameter("time");String place = request.getParameter("place");  
String people = request.getParameter("people");String hotel = request.getParameter("hotel");  
String id=UUID.randomUUID().toString().replace("-", "");  
System.out.println(id);session.setAttribute("id",id);  
try { Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");  
 Connection conn = DriverManager.getConnection("jdbc:sqlserver://localhost:1433;DatabaseName=meeting;user=sa;password=notbad");Statement stat= conn.createStatement();  
 String sql="insert into huiyi(meetingname,time,place,hotel,holder,number,creater) values('"+meetingname+"','"+time+"','"+place+"','"+hotel+"','"+people+"','"+id+"','"+account+"')";  
 System.out.println(request.getParameter("account"));  
 System.out.println(sql);stat.executeUpdate(sql);  
 String[] check= request.getParameterValues("attender");  
 String[] index={"jname","workplace","idcard","phonenumber","attendtime","sex","room"};  
 for(int j=0;j<check.length;j++){  
 for(int t=0;t<index.length;t++){  
 if(check[j].equals(index[t])){index[t]="是"; }}}  
 for(int j=0;j<index.length;j++){if(!index[j].equals("是")){index[j]="否"; }}  
 String sql1="insert into 参会人员信息(会议号,姓名,工作单位,身份证号,电话,参会时间,性别,是否安排房间,会议名称) values(" + "'"+id+"','"+index[0]+"','"+index[1]+"','"+index[2]+"','"+index[3]+"','"+index[4]+"','"+index[5]+"','"+index[6]+"','"+meetingname+"')";  
 stat.executeUpdate(sql1);  
 out.println("您已成功创建会议:"+request.getParameter("title")+"<br>");  
 String sql2="select *\**from huiyi where huiyi.number like '"+id+"'";  
 System.*out*.println(sql2);  
 ResultSet rs=stat.executeQuery(sql2);  
 while(rs.next()) {  
 out.println("会议名称:" + rs.getString("meetingname")+"<BR>");  
 out.println("时间:" + rs.getString("time")+"<BR>");  
 out.println("地点:" + rs.getString("place")+"<BR>");  
 out.println("宾馆:" + rs.getString("hotel")+"<BR>");  
 out.println("主讲人:" + rs.getString("holder")+"<BR>");  
 out.println("会议编号：" + rs.getString("number")+"<BR>");  
 out.print("<a href=\"LookMeet?id=" + id+ "\">查看我的会议</a>");  
 } rs.close(); stat.close();conn.close();  
} catch (ClassNotFoundException | SQLException e) { e.printStackTrace();}

把本次参会的人员信息下载生成EXCEL文件功能的实现：构建一个通用的CommonExcle类用于将数据库文件转换为String title = **"会议报表"**; String[] rowsName = **new** String[]{**"序号"**, **"会议名称"**, **"姓名"**, **"工作单位"**, **"身份证号"**, **"电话"**, **"性别"**,**"是否安排房间"**,**"会议号"**}List<Object[]> dataList = **new** ArrayList<Object[]>();  
 Object[] obj1 = **null**;  
 String sql = **"SELECT 会议名称,姓名,工作单位,身份证号,电话,性别,是否安排房间,会议号 from 参会人员信息 where 会议号='"**+s+**"'"**;  
 System.***out***.println(sql);  
 ResultSet rs = stat.executeQuery(sql);  
 **while** (rs.next()) {  
 obj1 = **new** Object[rowsName.**length**];  
 obj1[1] = rs.getString(**"会议名称"**);  
 obj1[2] = rs.getString(**"姓名"**);  
 obj1[3] = rs.getString(**"工作单位"**);  
 obj1[4] = rs.getString(**"身份证号"**);  
 obj1[5] = rs.getString(**"电话"**);  
 obj1[6] = rs.getString(**"性别"**);  
 obj1[7] = rs.getString(**"是否安排房间"**);  
 obj1[8] = rs.getString(**"会议号"**);  
 dataList.add(obj1);  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 String fileName = **"会议报表"** + String.*valueOf*(System.*currentTimeMillis*()).substring(4, 13) + **".xls"**;  
 CommonExcel ex = **new** CommonExcel(title, rowsName, dataList, response, fileName);  
 **try** {  
 ex.downloadExcel();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }

4.2会议参与者模块：

参与者登陆与注册功能：通过JoinerDao类将Joiner类实例化为ArrayList，实例化得到包含所有参与者账号密码信息的ArrayList类型的对象joiners，遍历获取joiners中的组织者对象，判断用户输入的账号密码是否与joiners中的组织者的账号密码相等，如果相等即参与者已经注册，输出已创建的会议供参与者用户选择；若参与者用户未注册则输出提示去注册。

关键代码：

PrintWriter out = response.getWriter();  
String account = request.getParameter(**"account"**);  
String password = request.getParameter(**"pwd"**);  
JoinerDao joinerDao=**new** JoinerDao();  
**int** d=0;ArrayList joiners= **null**;  
**try** {joiners = joinerDao.queryAllJoiners();  
} **catch** (Exception e) {e.printStackTrace();}  
**for**(**int** i=0;i<joiners.size();i++){  
 Joiner joiner=(Joiner) joiners.get(i);  
 **if**(joiner.getAt().equals(account)&&joiner.getPd().equals(password)) {  
 d=1;**break**;}}  
**if**(d==1){  
 HuiyiDao meetingDao=**new** HuiyiDao();  
 ArrayList meetings = **null**;  
 **try** {  
 meetings = meetingDao.queryAllHuiyis();  
 } **catch** (Exception e) {  
 e.printStackTrace();HttpSession session=request.getSession();  
 session.setAttribute(**"number"**, account);  
 **for** (**int** j= 0; j< meetings.size(); j++)  
 {Huiyi meeting = (Huiyi) meetings.get(j);  
 out.println(**"<div style=\"color:black;width:auto;height:auto;border-style: groove;float:left;margin-right:60px;background-color:#00ffff\">"**);  
 out.println(**"会议名:"**+meeting.getMeetingname()+ **"<br>"**);  
 out.println(**"会议时间:"**+meeting.getTime() + **"<br>"**);  
 out.println(**"会议地点:"**+meeting.getPlace() + **"<br>"**);  
 out.println(**"会议主讲人:"**+meeting.getHolder() + **"<br>"**);  
 out.println(meeting.getHotel() + **"<br>"**);  
 out.println(**"会议号码:"** + meeting.getNumber() + **"<br>"**);  
 out.println(**"<br>"**); out.println(**"<a href=/testjdbc4\_war\_exploded/servlets/AttendMeet?number="**+meeting.getNumber()+**"&joiner="**+account+**"&name="**+meeting.getMeetingname()+**">参加会议</a>"**);  
 out.println(**"</div>"**);}}**else**{  
 out.println(**"对不起，查无此用户！"**+**"<a href=\"http://localhost:8080/testjdbc4\_war\_exploded/joinerregister.html\">去注册</a>"**);  
}注册：Statement stat= conn.createStatement();  
String sql=**"insert into joiner(jaccount,password) values('"**+account+**"','"**+password+**"')"**;  
**int** i= 0;System.***out***.println(sql);i = stat.executeUpdate(sql);  
out.println(**"成功创建"**+i+**"位参与者用户！"**+**"<br>"**+**"<a href=\"http://localhost:8080/testjdbc4\_war\_exploded/Joiner.html\">去登陆</a>"**);  
 参与者参加想要参加的会议功能:通过获取用户点击的会议对应的会议名称在参会人员信息表中查找会议记录并读取会议要求填写的参会者个人信息情况，当要求填写的个人参会信息为是输出相应的文本框供用户输入个人信息，用户输入完成后点击提交提交个人信息，用户报名会议成功。

关键代码：

String account = request.getParameter(**"account"**);  
String id = request.getParameter(**"number"**);

Stringnumber1=request.getParameter(**"number"**);  
System.***out***.println(account);String meetingname = request.getParameter(**"name"**);  
PrintWriter out = response.getWriter();

String sql = **"select 姓名,工作单位,身份证号,电话,参会时间,性别,是否安排房间 from 参会人员信息 where 会议名称='"** + meetingname + **"'"**;  
 ResultSet rs = stat.executeQuery(sql);  
 String[] str = **new** String[7];String[] str2 = **new** String[7];  
 String[] str1 = **new** String[]{**"姓名"**, **"工作单位"**, **"身份证号"**, **"电话"**, **"参会时间"**, **"性别"**, **"是否安排房间"**};  
**while** (rs.next()) {  
 str[0] = rs.getString(**"姓名"**);str[1] = rs.getString(**"工作单位"**);  
 str[2] = rs.getString(**"身份证号"**);str[3] = rs.getString(**"电话"**);  
 str[4] = rs.getString(**"参会时间"**);tr[5] = rs.getString(**"性别"**);  
 str[6] = rs.getString(**"是否安排房间"**);  
}  
 **for** (**int** k = 0; k < 100; ) {  
 **for** (**int** j = 0; j <= 6; j++) {  
 **if** (str[j].equals(**"是"**)) {  
 str2[j] = str1[j];  
 } **else** str2[j] = **"无"**;}**break**;}  
out.println(**"<form action=\"SubmitInformation\">"**);  
out.println(**"会议名称为"** + **"<input type=\"text\" name=meetingname value=\""** + meetingname + **"\"><br>"**);  
**for** (**int** j = 0; j < 7; j++) {  
 **if** (!(str2[j].equals(**"无"**)))  
 out.println(**"请填写你的"** + str2[j] + **"<input type=\"text\" name=str2><br>"**);  
}out.println(**"<input type=\"submit\" value=\"提交\">"**);  
out.println(**"<input style=\"display:none\" type=\"text\" name=number value=\""** + number1 + **"\">"**);  
out.println(**"</form>"**);  
HttpSession session = request.getSession();  
session.setAttribute(**"arrayList"**, str2);

提交参与者填写会议要求填写的个人信息功能：将用户输入的参会信息数据以数组的方式存入session，在SubmiInformationServlet中获取数据并将数据信息处理好插入参会人员信息表。

关键代码:

String[] name=**new** String[7];  
String[] s=**new** String[15];  
HttpSession session=request.getSession();  
name = (String[]) session.getAttribute(**"arrayList"**);  
String meetingname=request.getParameter(**"meetingname"**);  
String number=request.getParameter(**"number"**);  
String str[] = request.getParameterValues(**"str2"**);  
**for**(**int** i=0;i<str.**length**;i++)  
{  
 s[i]=str[i];  
}  
**for**(**int** i=0;i<7;i++)  
{  
 **if**(s[i]==**null**)  
 {  
 s[i]=**"无"**;  
 }  
}  
**for**(**int** i=0;i<name.**length**;i++)  
{  
 **if**(name[i].equals(**"无"**)){  
 **for**(**int** j=14;j>i;j--){  
 s[j]=s[j-1];s[i]=**"无"**; }}  
String sql = **"insert into 参会人员信息(会议号,姓名,工作单位,身份证号,电话,参会时间,性别,是否安排房间,会议名称)"** +  
**"values('"**+number+**"','"**+s[0]+**"','"**+s[1]+**"','"**+s[2]+**"','"**+s[3]+**"','"**+s[4]+**"','"**+s[5]+**"','"**+s[6]+**"','"**+meetingname+**"')"**;  
stat.executeUpdate(sql);out.println(**"成功参加会议"+meetingname**);

4.3管理员模块：

管理员账号登陆验证：通过ManagerDao类将Maneger类实例化为ArrayList，实例化得到包含所有管理者账号密码信息的ArrayList类型的对象managers，遍历获取managers中的管理者者对象，判断用户输入的账号密码是否与managers中的管理者的账号密码相等，如果相等即用户是管理员，输出已创建的会议供管理者用户选择；若不是管理员用户则输出提示你的账号不是管理员。

关键代码：

**int** d=0;  
String account = request.getParameter(**"account"**);  
String password = request.getParameter(**"pwd"**);  
ManagerDao managerDao=**new** ManagerDao();ArrayList managers= **null**;  
**try** {managers = managerDao.queryAllManagers();  
} **catch** (Exception e) { e.printStackTrace();}  
**for**(**int** i=0;i<managers.size();i++){  
 Manager manager=(Manager) managers.get(i);  
 **if**(manager.getAt().equals(account)&&manager.getPd().equals(password)) {  
 d=1; **break**; }  
}**if**(d==1){  
 request.getRequestDispatcher(**"/servlets/GetMeet"**).forward(request, response);}  
**else**{out.println(**"您不是管理员！"**);}

查看当前存在的会议功能：

通过在GetMeetServlet中用HuiyiDao实例化会议对象，逐个输出显示会议信息，并提供删除此会议的超链接。

关键代码：PrintWriter out = response.getWriter();  
 **for** (**int** j= 0; j< meetings.size(); j++)  
 {  
 Huiyi meeting = (Huiyi) meetings.get(j);  
 out.println(**"<div style=\"color:black;width:auto;height:auto;border-style: groove;float:left;margin-right:60px;background-color:#00ffff\">"**);  
 out.println(**"会议名:"**+meeting.getMeetingname()+ **"<br>"**);  
 out.println(**"会议时间:"**+meeting.getTime() + **"<br>"**);  
 out.println(**"会议地点:"**+meeting.getPlace() + **"<br>"**);  
 out.println(**"会议主讲人:"**+meeting.getHolder() + **"<br>"**);  
 out.println(meeting.getHotel() + **"<br>"**);  
 out.println(**"会议号码:"** + meeting.getNumber() + **"<br>"**);  
 out.println(**"<br>"**); out.println(**"<a href=/testjdbc4\_war\_exploded/servlets/DeleteMeet?name="**+meeting.getMeetingname()+**">删除此会议</a>"**); out.println(**"</div>"**);}

删除想要删除的会议功能：通过获取想要删除会议对应的会议名并以此为关键词通过sql删除会议名与之相等的会议。

关键代码：

PrintWriter out = response.getWriter();  
String name = request.getParameter(**"name"**);  
 String sql=**"delete from meeting.dbo.huiyi where meetingname='"**+name+**"'"**;  
 **int** i= 0;  
 i = stat.executeUpdate(sql);  
 out.println(**"成功删除"**+i+**"个会议！"**);

1. 实验数据及处理结果

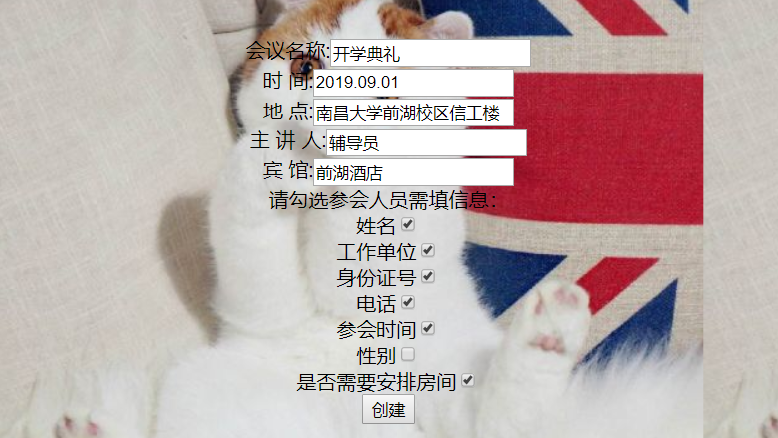
入口主页面：



组织者模块：







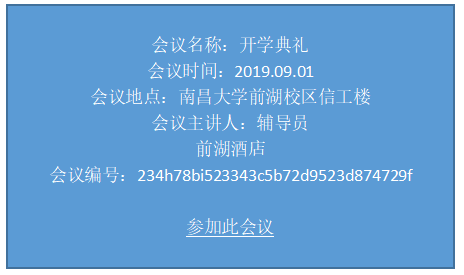


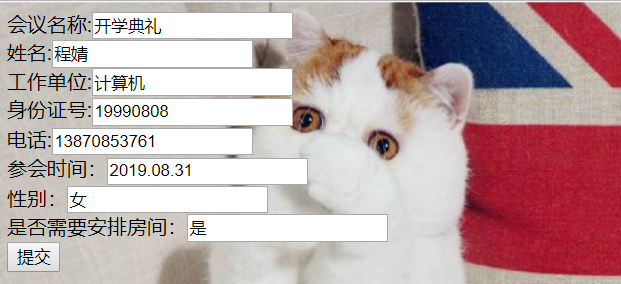


参与者模块：





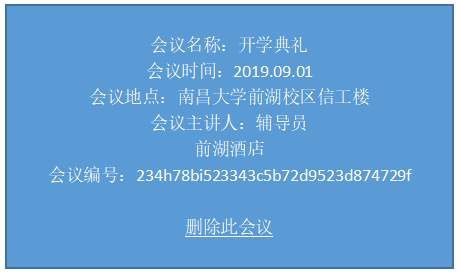






管理者模块：







1. 思考讨论题或体会或对改进实验的建议

本项目虽然是期中项目，但是动手时对web项目的完整架构还理解不透彻，所以在初期的时候走了很多弯路，并且容易被参考别人的思路误导，但是通过逐渐进行实验和最后的期末项目的推进逐渐找到节奏，期中项目的代码相较期末项目而言比较呆板繁琐，前端将数据通过form标签提交到servlet的，servlet将数据传入service层调用dao层的方法，dao层用来与数据库进行查询以及增删改查，操作会议组织者登陆与注册功通过LauncherDao类将Launcher类实例化为ArrayList，实例化得到包含所有组织者账号密码信息的ArrayList类型的对象lauchers，遍历获取lauchers中的组织者对象，判断输入的账号密码是否与lachers中的组织者的账号密码相等从而跳转不同页面，而参与者登陆与注册功能：通过JoinerDao类将Joiner类实例化为ArrayList，实例化得到包含所有参与者账号密码信息的ArrayList类型的对象joiners，遍历获取joiners中的组织者对象，判断用户输入的账号密码是否与joiners中的组织者的账号密码相等继而跳转，用户输入的参会信息数据以数组的方式存入session，在SubmiInformationServlet中获取数据并将数据信息处理好插入参会人员信息表；管理员账号验证则通过ManagerDao类将Maneger类实例化为ArrayList，实例化得到包含所有管理者账号密码信息的ArrayList类型的对象managers，遍历获取managers中的管理者者对象，判断用户输入的账号密码是否与managers中的管理者的账号密码相等 。在数据库连接的时候遇到瓶颈，通过大量资料的查找和代码理解，最终实现了数据库的连接，在整个功能设计时，由于一开始对需求分析没有考虑全面，所以在后期很多功能难以实现，导致整个逻辑部分大改，，这也是今后完成项目需要好好改进的地方。通过本次实验我的编程能力和逻辑理解能力都提高很多，但是加分项的功能没有得到实现，项目完成的效率不高，今后的代码学习需要好好沉淀。