

卓越软件开发基础 Project 说明文档

——19302010009 钱麒丹

一、项目概述

在本次卓越软件开发基础的 Project 中，通过对前后端的独立开发。我基本实现了项目中的基本功能，并适当完成了部分 Bonus 任务。在技术实现上，我选择 Tomcat 作为容器，在前端页面的构建中使用到了 Bootstrap 框架，在后端则使用了 JSP+Servlet+JavaBean 的 MVC 架构来实现，具体代码编写过程中还使用到了如用于压缩图片的 Thumbnailator 这般的类库。

项目服务器部署地址: <http://cymggdwebfundamental.ink:8080/project/>

项目 GitHub 地址: <https://github.com/HakureiReimuOVO/WebProject-2020-7/>

二、项目结构

/Vacation

-/out 项目编译完毕的输出目录
-/src 存放项目中使用的 Java 类
 -/bean **MODEL** 存放 JavaBean
 -/DAO 存放访问数据库的 DAO
 -/servlet **CONTROLLER** 存放各类 Servlet
 -/service 存放响应页面请求的 Servlet
 -/viewer 存放实现主页面跳转解析的 Servlet
-/utils 存放项目中的实用类
-/web
 -/ajax 存放响应 Ajax 请求的 JSP 页面
 -/pages **VIEW** 存放项目中的 JSP 主页面
 -/src 存放项目中使用到的资源
 -/bootstrap-4.5.0-dist 存放 Bootstrap 框架
 -/jquery-3.5.1 存放 JQuery
 -/CSS 存放各主页面的 CSS
 -/JavaScript 存放 JavaScript
 -/images
 -/html-images 存放 html 页面中使用到的图片
 -/travel-images 存放用户上传的图片
 -/large 大尺寸
 -/medium 中尺寸
 -/small 小尺寸
-/WEB-INF
 -/classes 存放编译完毕的 Java 类
 -/lib 存放项目中使用的 Jar 包
 -/web.xml 项目的配置文件
-/travels.sql 项目的原始数据库文件

Java 类

/src

...../bean (1)

...../DAO (2)

...../servlet

...../service (3)

...../viewer (4)

...../utils (5)

(1) bean

bean 包中存放了项目中使用到的 **JavaBean**，用于封装项目中使用到的重要对象。我在项目中构建并使用到了 **User**、**Photo** 和 **Message** 三个 bean。其中 User 对象用于封装数据库中 与用户相关的所有信息；Photo 对象用于封装与用户上传照片相关的所有信息；Message 用于封装用户发送的实时消息的所有信息。构建此类 JavaBean 可以使得在 Controller 向页面传递信息以及页面调用相关信息时，代码更加清晰可读且实用。

(2) DAO

DAO 包中存放了项目中使用到的 **DAO**，用于简化数据访问的流程。我在其中构建了 **DAO**、**UserDAO**、**PhotoDAO** 和 **MessageDAO** 四个 Java 类。DAO 类是其余三个类的父类，其通过 DBCP 连接池获得数据库连接，可以使用通过 DAO 类中的方法来更新数据库，查询数据库并返回相应的结果集 Map、List<Map>等等。而其余三个类则是用于读取指定数据的 DAO 类，例如 UserDAO 返回的结果集为 User 或 List<User>类型，这样可以极大程度上方便对数据库数据的读取。

(3) service

service 包中存放了项目中负责**响应页面请求**的 Servlet。例如 FavorServlet 用于响应收藏和取消收藏图片的请求，LoginServlet 用于响应登录请求等等。其作为页面和数据库进行数据交互的中介，分析处理页面请求中数据，并调用数据库返回数据对应的结果。

(4) viewer

viewer 包中存放了项目中负责**页面跳转**的 Servlet。例如其中的 Index 在用户访问/index 时被调用，其通过与数据库的交互获得 PopularPhotos 以及 NewPhotos，并将其作为参数发送至 index.jsp，最终使 index.jsp 完成数据的呈现。其作为页面初始化调用数据时必要的 Servlet。

(5) utils

utils 包中存放了项目中常用的**实用方法**。例如根据 CityCode 或 CountryCode 查询对应的 CityName 和 CountryName，以及根据 ImageID 获得 Favor 数量等等。使得编写过程不必写重复且冗长的功能代码，增加了代码的可复用性。

JSP 文件

```
/pages
...../chat.jsp
...../details.jsp
...../favor.jsp
...../friend.jsp
...../index.jsp
...../login.jsp
...../photo.jsp
...../register.jsp
...../search.jsp
...../upload.jsp
...../welcome.jsp
```

项目中共 10 个主要页面，分别有与之相对应的 JSP 文件。有一个特殊的 JSP 文件为 welcome.jsp，其用于访问项目根目录时，跳转至/index 路径下。这么做是因为在 Tomcat 的首页配置中，只能够选择具体的 JSP 页面作为首页，因此我选择将 welcome.jsp 作为首页，并进行相应跳转，以导航至正确的首页位置。

Ajax 实现

```
/ajax
...../chatAjax.jsp
...../favorAjax.jsp
...../photoAjax.jsp
...../searchAjax.jsp
```

注意到响应 Ajax 的是 JSP 文件。在本次 PJ 项目中，我首先将页面发出 Ajax 请求后所得到的响应格式确定为文本，并因此使用了 **Page->ServiceServlet->AjaxJSP** 这样的结构。页面首先将 Ajax 请求发送至 ServiceServlet，ServiceServlet 即 service 包中的部分 Servlet，其负责读取并处理 Ajax 请求的参数信息，再将其发送至相应 JSP 页面。JSP 页面负责将相应数据处理进文本中，最后 JSP 将处理完毕的文本发送回原页面中，至此便实现了 Ajax 请求的步骤。

数据库结构

```
Travels
|----geocities 存放与 CityCode 对应的城市信息
|----geocountries_regions 存放与 CountryCode 对应的国家信息
|----travelimage 存放用户上传的图片信息
|----travelimagefavor 存放用户收藏图片的信息
|----traveluser 存放用户信息
|----traveluserfriend 存放用户好友的信息
|----travelusermessage 存放用户好友间聊天的信息
```

三、项目功能

基本功能实现

项目中的基本功能大体上均可以通过 JSP+Servlet+JavaBean 的模型实现。此次项目中，各页面的基本功能我均已在以上结构的基础上进行实现，由于功能较多且实现原理多重复，仅选择**部分功能**实现作解释。

(1) 表单验证

在本项目中，大部分表单信息我都选择在**前端**使用 JS 进行验证，包括文件上传和注册登录等等，以减轻服务器压力。但并不包括部分比如说重复用户名以及登录失败，此类必须与服务器进行交互的验证内容。

(2) Session 与 Cookie

登录状态保持我使用了 **Session** 作为实现基础。登录和注册完后跳转至先前页面，我也使用了 **Session** 作为实现基础，原理为访问每一个页面过后，都更新一次 Session 中最后访问页面的相关属性。足迹功能我使用了 **Cookie** 作为实现基础，同时在删除和修改图片过后都有必要对 **Cookie** 进行更新，因为这两个操作都会改变足迹的原先内容。

(3) Cookie 标题保存

在将标题保存至 **Cookie** 中时，我选择将标题先 **URLEncode** 一次，并在调用时 Decode 一次。原因是 Cookie 的旧标准中使得不能包括空格引号等的特殊字符，在存储图片标题可能出现报错。

Bonus 实现

Bonus 中我基本实现了图片局部放大功能，注册与登录验证码功能，好友用户实时聊天功能以及项目的云部署。

(1) 图片局部放大功能

图片局部放大功能的实现基于 JS。首先在页面中内置另一张根据比例放大的图片，并进行隐藏。在光标经过图片触发事件时，显示放大的图片，并通过光标当前的坐标相对原图的位置进行计算，对放大的图片进行**裁剪**以及**偏移**，进而实现图片局部放大。



（2）注册与登录验证码功能

注册与登录验证码功能的实现，关键在于通过 Servlet 生成一张验证码的图片并将其作为 response 返回，并在生成的同时，将验证码的文本内容存储至 **Session** 中以便之后验证。在前端页面，则需要将验证码图片的 src 设置为该 Servlet 的映射路径，以便成功显示验证码图片。在用户注册与登录时，服务器可调用 **Session** 中的正确验证码内容进行比对验证。

（3）好友用户实时聊天功能

好友用户实时聊天功能的实现主要基于 Ajax，利用对服务器的定时请求实现用户间的聊天，在聊天页面加载完毕、自己的消息发送以及间隔一定时间段后，页面都会再次发送 Ajax 请求以获取聊天信息。

（4）项目的云部署

我使用了 Ubuntu 系统的服务器进行了项目的云部署。至于具体实现，首先需要在服务器端进行 **Tomcat**、**mysql** 以及 **JDK** 的安装，并配置相关环境。然后将本地的项目打包输出为 **war** 文件并上传至服务器端，根据服务器端环境的不同完成最终调试，实现云部署。