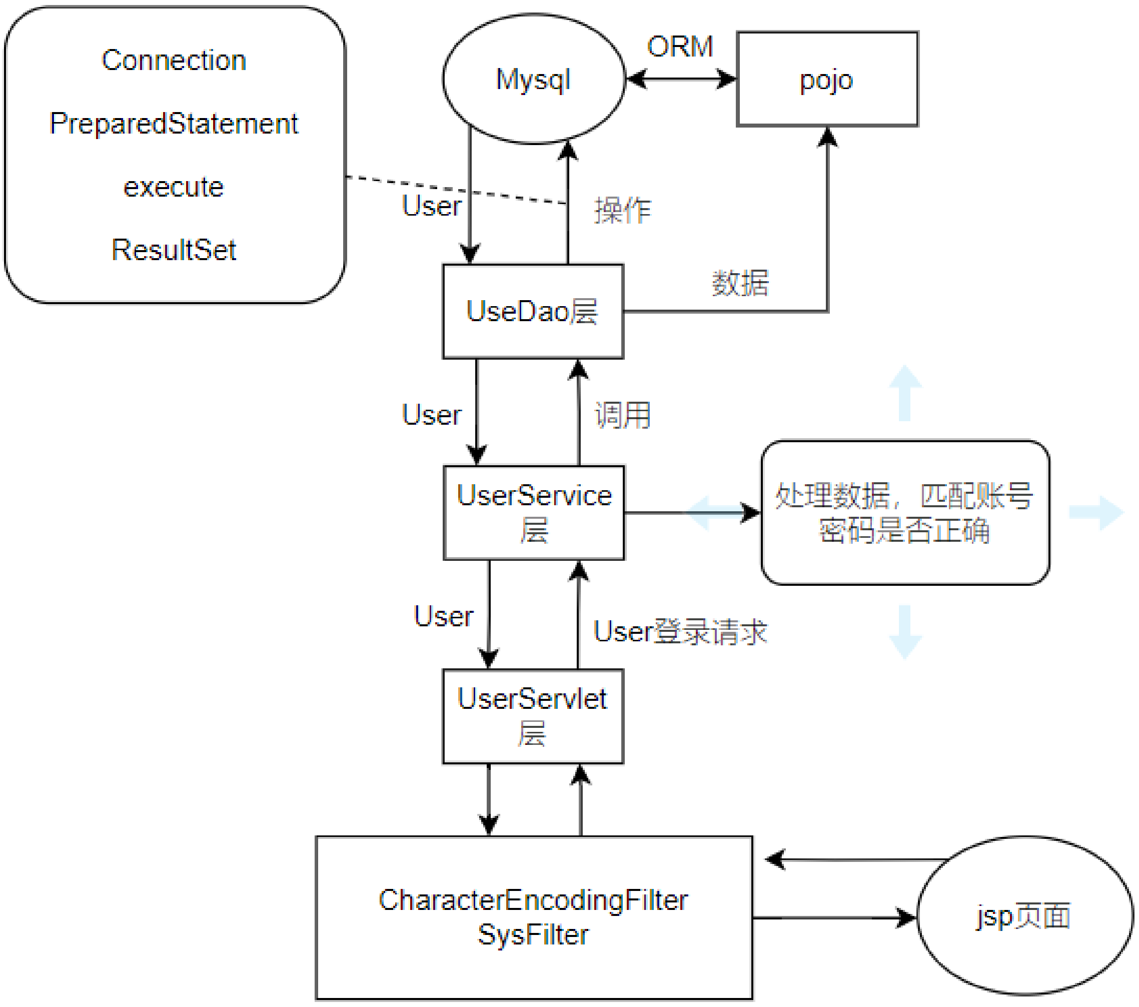
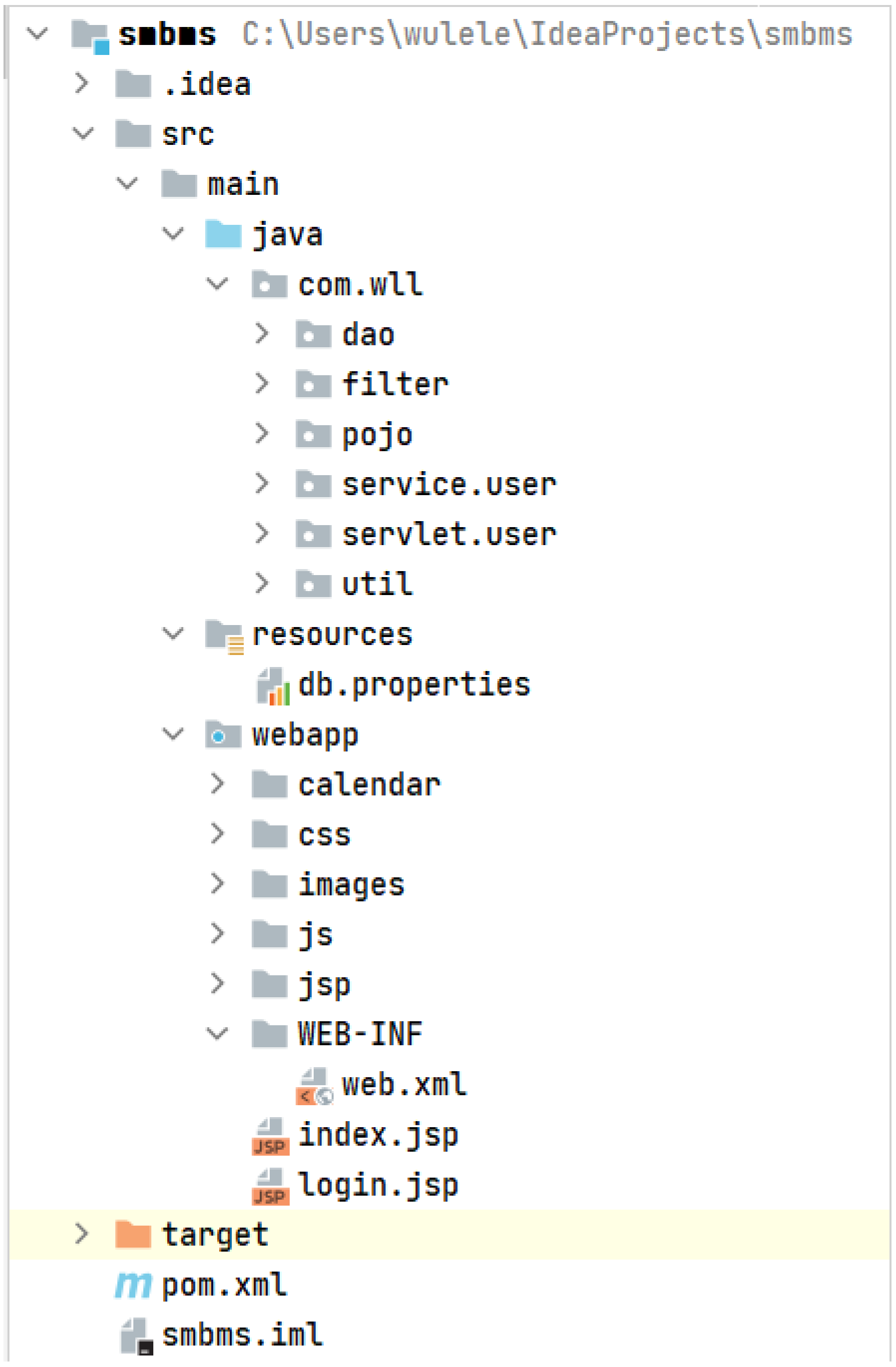
smbms学习笔记（服务器端）

大致的一个流程，从用户的pojo**到**dao**层创建，**service**层和登录、登出**servlet**层的书写。**



文件目录



```
smbms  C:\Users\wulele\IdeaProjects\smbms
  .idea
  src
    main
      java
        com.wll
          dao
          filter
          pojo
          service.user
          servlet.user
          util
      resources
        db.properties
      webapp
        calendar
        css
        images
        js
        jsp
        WEB-INF
          web.xml
        index.jsp
        login.jsp
  target
  pom.xml
  smbms.iml
```

pojo层（User）

```java
package com.wll.pojo;

import lombok.*;

import java.util.Calendar;
import java.util.Date;

/**
 * @author wulele
 * <p>
 * 用户表
 */
@Getter
@Setter
```

```java
@NoArgsConstructor
@AllArgsConstructor
@ToString
public class User {
    private Integer id;
    private String userCode;
    private String userName;
    private String userPassword;
    private Integer gender;
    private Date birthday;
    private String phone;
    private String address;
    private Integer userRole;
    private Integer createdBy;
    private Date creationDate;
    private Integer modifyBy;
    private Date modifyDate;

    private Integer age;
    private String userRoleName;

    public void setUserRoleName(String userRoleName) {
        this.userRoleName = userRoleName;
    }

    public Integer getAge() {
        Calendar calendar = Calendar.getInstance();
        Calendar date = Calendar.getInstance();
        date.setTime(this.birthday);
        return calendar.get(Calendar.YEAR) -
date.get(Calendar.YEAR);
    }
}
```

公共类

```java
package com.wll.dao;

import java.io.IOException;
import java.io.InputStream;
import java.sql.*;
import java.util.Properties;

/**
 * @author wulele
 * <p>
 * 操作数据库公共类
 */
```

```
public class BaseDao {
    private static String driver;
    private static String url;
    private static String username;
    private static String password;

    static {
        Properties properties = new Properties();
        //通过类加载器读取资源
        InputStream rs =
BaseDao.class.getClassLoader().getResourceAsStream("db.properties"
);
        try {
            properties.load(rs);
        } catch (IOException e) {
            e.printStackTrace();
        }
        driver = properties.getProperty("driver");
        url = properties.getProperty("url");
        username = properties.getProperty("username");
        password = properties.getProperty("password");
    }

    public static Connection getConnection() throws
ClassNotFoundException, SQLException {
        Connection connection = null;
        //反射获取对象
        Class.forName(driver);
        connection = DriverManager.getConnection(url, username,
password);
        return connection;
    }

    /**
     * 查询公共类
     */
    public static PreparedStatement execute(Connection connection,
String sql, Object[] params, PreparedStatement preparedStatement,
ResultSet resultSet) throws SQLException {
        preparedStatement = connection.prepareStatement(sql);
        for (int i = 0; i < params.length; i++) {
            //preparedStatement占位符从1开始
            preparedStatement.setObject(i + 1, params[i]);
        }
        return preparedStatement;
    }

    /**
     * 增删改共类
     */
    public static PreparedStatement executeUpdate(Connection
connection, String sql, Object[] params, PreparedStatement
preparedStatement) throws SQLException {
        preparedStatement = connection.prepareStatement(sql);
        for (int i = 0; i < params.length; i++) {
            //preparedStatement占位符从1开始
            preparedStatement.setObject(i + 1, params[i]);
        }
        return preparedStatement;
    }

    /**
     * 关闭资源
     */
    public static boolean closeResources(Connection connection,
PreparedStatement preparedStatement, ResultSet resultSet) {
        boolean flag = true;
        if (resultSet != null) {
            try {
                resultSet.close();
                //GC回收
                resultSet = null;
            } catch (SQLException throwables) {
                throwables.printStackTrace();
                flag = false;
            }
        }
        if(preparedStatement!=null){
            try {
                preparedStatement.close();
                preparedStatement = null;
            } catch (SQLException throwables) {
                throwables.printStackTrace();
                flag = false;
            }
        }
        if(connection != null){
            try {
                connection.close();
                connection = null;
            } catch (SQLException throwables) {
                throwables.printStackTrace();
                flag = false;
            }
        }
        return flag;
    }
}
```

持久层（UserDao、UserDaoImpl）

```java
package com.wll.dao.user;

import com.wll.pojo.User;

import java.sql.Connection;
import java.sql.SQLException;

/**
 * @author wulele
 */
public interface UserDao {
    /**
      * get user info.
      * @param connection
      * @param userCode
      * @return User
      * @throws SQLException getUserInfo
      */
    public User getUserInfo(Connection connection, String
userCode) throws SQLException;
}
```

```java
package com.wll.dao.user;

import com.wll.dao.BaseDao;
import com.wll.pojo.User;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 * @author wulele
 */
public class UserDaoImpl implements UserDao {
    @Override
    public User getUserInfo(Connection connection, String
userCode) throws SQLException {
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;
        User user = null;
        if (connection ≠ null) {
            String sql = "select * from smbms_user where userCode
= ?";
            Object[] params = {userCode};
            preparedStatement = BaseDao.execute(connection, sql,
params, preparedStatement, resultSet);
            resultSet = preparedStatement.executeQuery();
            if (resultSet.next()) {
                user = new User();
                user.setId(resultSet.getInt("id"));
                user.setUserCode(resultSet.getString("userCode"));
                user.setUserName(resultSet.getString("userName"));

 user.setUserPassword(resultSet.getString("userPassword"));
                user.setGender(resultSet.getInt("gender"));
                user.setBirthday(resultSet.getDate("birthday"));
                user.setPhone(resultSet.getString("phone"));
                user.setAddress(resultSet.getString("address"));
                user.setUserRole(resultSet.getInt("userRole"));
                user.setCreatedBy(resultSet.getInt("createdBy"));

 user.setCreationDate(resultSet.getTimestamp("creationDate"));
                user.setModifyBy(resultSet.getInt("modifyBy"));

 user.setModifyDate(resultSet.getTimestamp("modifyDate"));
            }
            BaseDao.closeResources(connection, preparedStatement,
resultSet);
        }
        return user;
    }
}
```

Service层（UserService、UserServiceImpl）

```
package com.wll.service.user;

import com.wll.pojo.User;

/**
 * @author wulele
 */
public interface UserService {
    /**
     * method for user login.
     * @param userCode
     * @param password
     * @return
     */
    public User login(String userCode,String password);
}
```

```
package com.wll.service.user;

import com.wll.dao.BaseDao;
import com.wll.dao.user.UserDao;
import com.wll.dao.user.UserDaoImpl;
import com.wll.pojo.User;

import java.sql.Connection;
import java.sql.SQLException;

/**
 * @author wulele
 */
public class UserServiceImpl implements UserService{
    /** 业务层调用Dao层 */
    private UserDao userDao;
    public UserServiceImpl(){
        userDao = new UserDaoImpl();
    }
    @Override
    public User login(String userCode, String password) {
        Connection connection  = null;
        User userInfo = null;
        try {
            connection = BaseDao.getConnection();
            userInfo = userDao.getUserInfo(connection, userCode);
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        } finally {
            BaseDao.closeResources(connection,null,null);
        }
        if (userCode.equals(userInfo.getUserCode()) &&
password.equals(userInfo.getUserPassword())){
            return userInfo;
        }else {
            return null;
        }
    }
}
```

Servlet层（LoginServlet 、LogutServlet）

```java
package com.wll.servlet.user;

import com.wll.pojo.User;
import com.wll.service.user.UserService;
import com.wll.service.user.UserServiceImpl;
import com.wll.util.Constant;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author wulele
 */
public class LoginServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req,
HttpServletResponse resp) throws ServletException, IOException {
        String userCode = req.getParameter("userCode");
        String userPassword = req.getParameter("userPassword");
        UserService userService = new UserServiceImpl();
        User user = userService.login(userCode, userPassword);
        if(user≠null){
            //登录保存session

 req.getSession().setAttribute(Constant.USER_SESSION,user);
            //重定向地址发生变化，项目地址为 smbms
            resp.sendRedirect("/smbms/jsp/frame.jsp");

 //resp.sendRedirect(req.getContextPath()+"/jsp/frame.jsp");
        }else {
            //错误信息只在一次请求中有效
            req.setAttribute("error","账号或密码有误");
            //请求转发，地址不发生变化，request对象未销毁，error信息保留

 req.getRequestDispatcher("/login.jsp").forward(req,resp);
        }
    }

    @Override
    protected void doPost(HttpServletRequest req,
HttpServletResponse resp) throws ServletException, IOException {
        doGet(req, resp);
    }
}
```

```java
package com.wll.servlet.user;

import com.wll.util.Constant;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author wulele
 */
public class LogoutServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req,
HttpServletResponse resp) throws ServletException, IOException {
        req.getSession().removeAttribute(Constant.USER_SESSION);
        resp.sendRedirect(req.getContextPath()+"/login.jsp");
    }

    @Override
    protected void doPost(HttpServletRequest req,
HttpServletResponse resp) throws ServletException, IOException {
        doGet(req, resp);
    }
}
```

字符过滤器、登出过滤器

```java
package com.wll.filter;

import javax.servlet.*;
import java.io.IOException;

/**
 * @author wulele
 */
public class CharacterEncoding implements Filter {
    @Override
    public void init(FilterConfig filterConfig) throws
ServletException {

    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse
response, FilterChain chain) throws IOException, ServletException
{
        request.setCharacterEncoding("utf-8");
        response.setCharacterEncoding("utf-8");
        chain.doFilter(request,response);
    }

    @Override
    public void destroy() {

    }
}
```

```java
package com.wll.filter;

import com.wll.pojo.User;
import com.wll.util.Constant;

import javax.servlet.*;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author wulele
 */
public class SysFilter implements Filter {
    @Override
    public void init(FilterConfig filterConfig) throws
ServletException {

    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse
response, FilterChain chain) throws IOException, ServletException
{
        HttpServletRequest req = (HttpServletRequest) request;
        HttpServletResponse resp = (HttpServletResponse) response;
        User user = (User)
req.getSession().getAttribute(Constant.USER_SESSION);
        if(user ==null){
            resp.sendRedirect("/smbms/jsp/error.jsp");
        }else {
            chain.doFilter(request,response);
        }
    }
    @Override
    public void destroy() {

    }
}
```

```xml
<!--CharacterEncoding filter-->
<filter>
    <filter-name>CharacterEncoding</filter-name>
    <filter-class>com.wll.filter.CharacterEncoding</filter-class>
</filter>
<filter-mapping>
    <filter-name>CharacterEncoding</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
<!--login filter-->
<filter>
    <filter-name>SysFilter</filter-name>
    <filter-class>com.wll.filter.SysFilter</filter-class>
</filter>
<filter-mapping>
    <filter-name>SysFilter</filter-name>
    <url-pattern>/jsp/frame.jsp</url-pattern>
</filter-mapping>
```