一.用户注册总流程

1.1.准备数据库

```
CREATE DATABASE PRODUCTDB;
USE PRODUCTDB;
CREATE TABLE t_user(
   id int(6) PRIMARY key auto_increment,
   username VARCHAR(20) not null,
   password VARCHAR(20) not null,
   idcard VARCHAR(20) null UNIQUE,
   phone VARCHAR(20) null UNIQUE,
   address VARCHAR(300) not null,
   email VARCHAR(40) UNIQUE
);
```

1.2.创建Web项目

1.3.准备JDBC的环境 这里使用的是mysql5，mysql8的链接看xmind中Java核心技术卷中内容

- 配置信息

```
driver=com.mysql.jdbc.Driver
url=jdbc:mysql://localhost:3306/productdb?
useSSL=false&characterEncoding=utf8
user=root
password=123456
```

- JDBC工具类

```
package com.product.utils;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;

public class DBUtil {
    private static String url;
    private static String user;
```

```
14      private static String password;
15
16      static {
17          Properties properties = new Properties();
18          try {
19
properties.load(DBUtil.class.getClassLoader().getResourceAsStr
eam("dbconfig.properties"));
20              String driverName =
properties.getProperty("driver");
21              url = properties.getProperty("url");
22              user = properties.getProperty("user");
23              password = properties.getProperty("password");
24              Class.forName(driverName);
25          } catch (IOException e) {
26              // TODO Auto-generated catch block
27              e.printStackTrace();
28          } catch (ClassNotFoundException e) {
29              // TODO Auto-generated catch block
30              e.printStackTrace();
31          }
32      }
33
34      public static Connection getConnection() {
35          Connection conn = null;
36          try {
37              conn = DriverManager.getConnection(url, user,
password);
38          } catch (SQLException e) {
39              e.printStackTrace();
40          }
41          return conn;
42      }
43
44      public static void closeLink(Connection conn, Statement
st, ResultSet rs) {
45          if (rs != null) {
46              try {
47                  rs.close();
48              } catch (SQLException e) {
49                  // TODO Auto-generated catch block
50                  e.printStackTrace();
51              }
52          }
53          if (st != null) {
54              try {
55                  st.close();
```

```
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }

    public static void closeLink(Connection conn, Statement st) {
        if (st != null) {
            try {
                st.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}

```

1.4.创建实体类

```
/**
 * @author bruceliu
 * @time 2019年8月14日上午9:54:33
 * @Description
 */
package com.product.bean;

/**
```

```
 * @author bruceliu
 * @time 2019年8月14日上午9:54:33
 * @Description User实体类
 */
public class User {

    private int id;
    private String username;
    private String password;
    private String idcard;
    private String phone;
    private String address;
    private String email;

    public User() {
        super();
    }


    public User(int id, String username, String password, String idcard, String phone, String address, String email) {
        super();
        this.id = id;
        this.username = username;
        this.password = password;
        this.idcard = idcard;
        this.phone = phone;
        this.address = address;
        this.email = email;
    }
```

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public String getUsername() { return username; }

public void setUsername(String username) { this.username = username; }

public String getPassword() { return password; }

public void setPassword(String password) { this.password = password; }

public String getIdcard() { return idcard; }

public void setIdcard(String idcard) { this.idcard = idcard; }

public String getPhone() { return phone; }

public void setPhone(String phone) { this.phone = phone; }

public String getAddress() { return address; }

public void setAddress(String address) { this.address = address; }

public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }

@Override public String toString() { return "User [id=" + id + ", username=" + username + ", password=" + password + ", idcard=" + idcard + ", phone=" + phone + ", address=" + address + ", email=" + email + "]"; }

}

```
1
2     •
3
4   #### 1.5.准备注册的页面
5
6   ```html
7   <%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
8
9   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
10  <html>
11    <head>
12      <title>用户注册</title>
13      <meta http-equiv="Content-Type"
    content="text/html;charset=utf-8" />
14      <meta http-equiv="pragma" content="no-cache">
15      <meta http-equiv="cache-control" content="no-cache">
16      <meta http-equiv="expires" content="0">
17      <meta http-equiv="keywords"
    content="keyword1,keyword2,keyword3">
18      <meta http-equiv="description" content="This is my page">
19      <link href="css/common.css" rel="stylesheet" type="text/css"
    />
20      <link href="css/style.css" rel="stylesheet" type="text/css"
    />
21    </head>
22
23    <body>
24      <div class="wrap">
25        <form action="user.do?act=reg" method="post">
26          <div class="zclf login logns">
27            <h1 class="blue">用户注册</h1>
```

```
28            <dl>
29              <dd>
30                <label> <small>*</small>用户名</label>
31                <input type="text" class="inputh lf" name="name" />
32
33              </dd>
34              <dd>
35                <label> <small>*</small>密码</label>
36                <input type="text" class="inputh lf" name="pwd" />
37
38              </dd>
39              <dd>
40                <label> <small>*</small>身份证号</label>
41                <input type="text" class="inputh lf" name="idCard"
    />
42
43              </dd>
44              <dd>
45                <label> <small>*</small>电话</label>
46                <input type="text" class="inputh lf" name="tel" />
47
48              </dd>
49              <dd>
50                <label> <small>*</small>住址</label>
51                <input type="text" class="inputh lf" name="address"
    />
52              </dd>
53              <dd>
54                <label> <small>*</small>邮箱</label>
55                <input type="text" class="inputh lf" name="email"
    />
56              </dd>
57              <dd class="hegas">
58                <label> </label>
59                <input type="submit" value="立即注册" class="spbg
    buttombg buttombgs f14 lf" />
60                <input type="button" value="登录" class="spbg
    buttombg buttombgs f14 lf"/>
61              </dd>
62            </dl>
63            <span style="color:red"></span>
64          </div>
65        </form>
66      </div>
67    </body>
68  </html>
```

1.6.准备注册的Servlet

```
1  package com.product.servlet;
2
3  import java.io.IOException;
4
5  import javax.servlet.ServletException;
6  import javax.servlet.annotation.WebServlet;
7  import javax.servlet.http.HttpServlet;
8  import javax.servlet.http.HttpServletRequest;
9  import javax.servlet.http.HttpServletResponse;
10
11 import com.product.bean.User;
12
13 @WebServlet("/RegisterServlet.do")
14 public class RegisterServlet extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16
17     protected void service(HttpServletRequest request,
   HttpServletResponse response) throws ServletException,
   IOException {
18
19         //解决POST请求的中文乱码问题
20         request.setCharacterEncoding("UTF-8");
21
22         String username = request.getParameter("username");
23         String password = request.getParameter("password");
24         String idcard = request.getParameter("idcard");
25         String phone = request.getParameter("phone");
26         String address = request.getParameter("address");
27         String email = request.getParameter("email");
28         //封装成对象(封装思想)
29         User u=new User(username, password, idcard, phone,
   address, email);
30         System.out.println(u);
31
32
33     }
34
35 }
36
```

1.7.准备业务逻辑层接口层

```
1  /**
```

```
 * @author bruceliu
 * @time 2019年8月14日上午10:38:22
 * @Description
 */
package com.product.service;

import com.product.bean.User;

/**
 * @author bruceliu
 * @time 2019年8月14日上午10:38:22
 * @Description 业务逻辑层接口层
 */
public interface UserService {

    //抽象方法:注册
    public abstract int register(User u);

}

```

1.8.准备业务逻辑层接口层实现层


```
/**
 * @author bruceliu
 * @time 2019年8月14日上午10:40:57
 * @Description
 */
package com.product.service.impl;

import com.product.bean.User;
import com.product.service.UserService;

/**
 * @author bruceliu
 * @time 2019年8月14日上午10:40:57
 * @Description 业务逻辑层实现层
 */
public class UserServiceImpl implements UserService{

    @Override
    public int register(User u) {
        return 0;
    }

}

```

1.9.准备数据库访问层的接口层

```java
/**
 * @author bruceliu
 * @time 2019年8月14日上午10:46:40
 * @Description
 */
package com.product.dao;

import com.product.bean.User;

/**
 * @author bruceliu
 * @time 2019年8月14日上午10:46:40
 * @Description
 */
public interface UserDao {

    //用户注册
    public abstract int register(User u);
}

```

1.10.准备数据库访问层的接口层实现层(真正操作数据库)

```java
/**
 * @author bruceliu
 * @time 2019年8月14日上午10:48:35
 * @Description
 */
package com.product.dao.impl;

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

import com.product.bean.User;
import com.product.dao.UserDao;
import com.product.utils.DBUtil;

/**
 * @author bruceliu
 * @time 2019年8月14日上午10:48:35
 * @Description 数据库访问层的实现层
 */
public class UserDaoImpl implements UserDao{

```

```java
    /**
     * 真正的去操作数据库
     */
    @Override
    public int register(User u) {
        Connection conn = DBUtil.getConnection();
        PreparedStatement ps=null;
        try {
            String sql="insert into t_user values
(null,?,?,?,?,?,?)";
            ps = conn.prepareStatement(sql);
            ps.setObject(1, u.getUsername());
            ps.setObject(2, u.getPassword());
            ps.setObject(3, u.getIdcard());
            ps.setObject(4, u.getPhone());
            ps.setObject(5, u.getAddress());
            ps.setObject(6, u.getEmail());
            return ps.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }finally{
            DBUtil.closeLink(conn, ps);
        }
        return 0;
    }

}

```

1.11.设置三层之间的调用关系

1.12.启动项目测试

这张照片中的内容是：
MVC三层架构和JavaWeb经典三层架构之间的对比