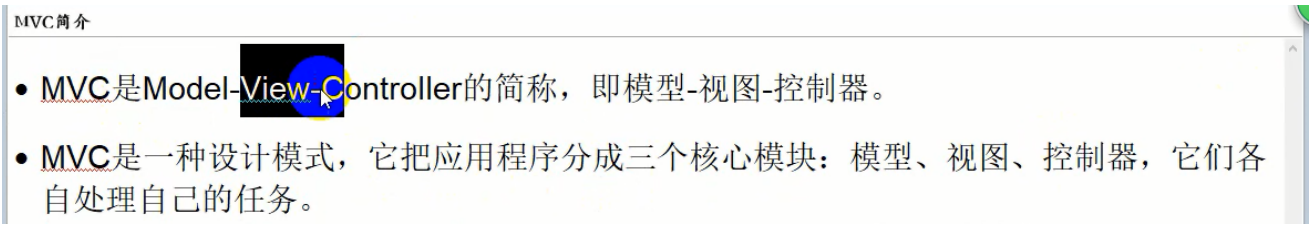
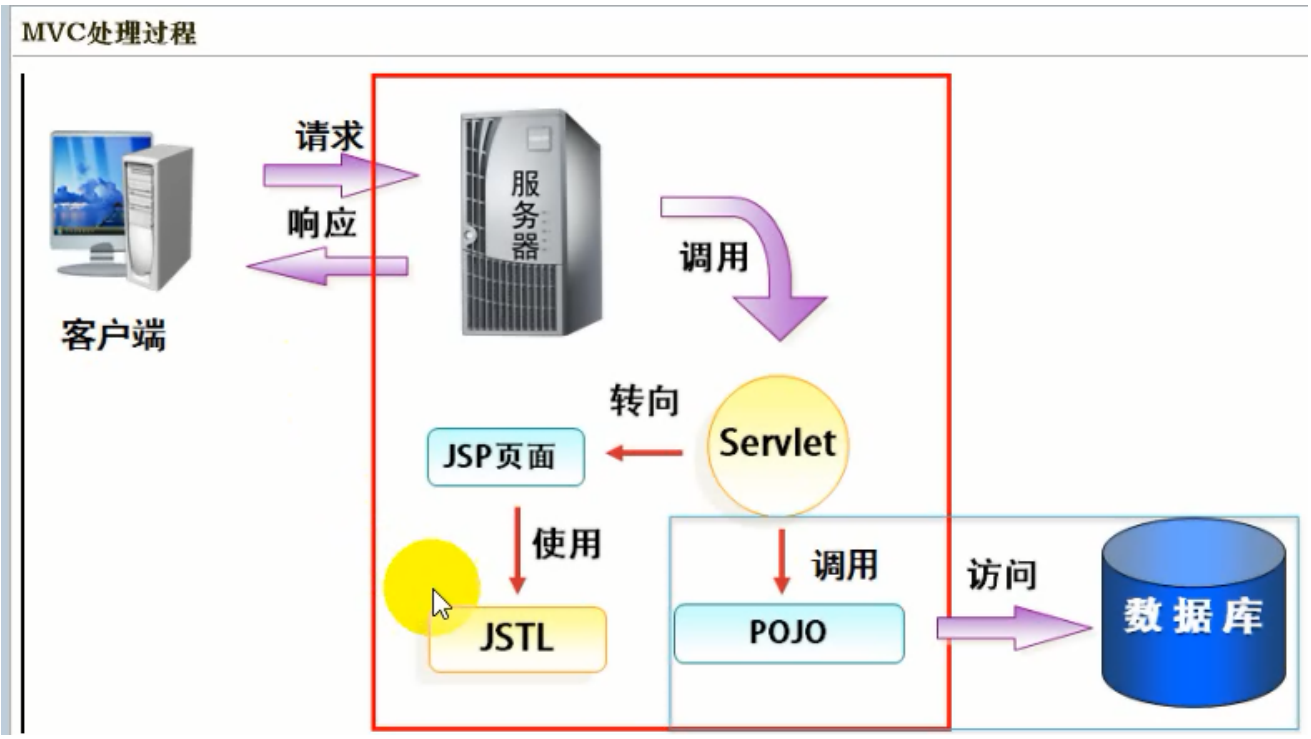
MVC学习

MVC简介





M

V

V

- 视图是用户看到并与之交互的界面，作用如下：
  - 视图向用户显示相关的数据。
  - 接受用户的输入。
  - 不进行任何实际的业务处理。

C

c

- 控制器接受用户的输入并调用模型和视图去完成用户的需求。
- 控制器接收请求并决定调用哪个模型组件去处理请求，然后决定调用哪个视图来显示模型处理返回的数据。

MVC的使用

项目查询流程分析

项目删除流程分析

utils层

```
public class JDBCUtils {
    private static DataSource ds = null;
    static{
        //数据源只能被创建一次
        ds = new ComboPooledDataSource("mvcdemo");
    }
    /**
     * 释放Connection链接
     * @param connection
     */
```

```
    public static void releaseConnection(Connection connection){
        try{
            if(connection != null){
                connection.close();
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
    /*
     * 返回数据源的一个Connection对象
     */
    public static Connection getConnection() throws SQLException {
        return ds.getConnection();
    }
    //获取数据源
    public  static  DataSource getDataSource()
    {
        return ds;
    }

}
```

pojo层

```
public class student {
    private int sid;
    private String sname;
    private int sage;
    private String sprovince;
    private int stuition;

    public student(){

    }

    public student(String sname, int sage, String sprovince, int stuition) {
        this.sname = sname;
        this.sage = sage;
        this.sprovince = sprovince;
        this.stuition = stuition;
    }

    public student(int sid, String sname, int sage, String sprovince, int stuition) {
        this.sid = sid;
        this.sname = sname;
        this.sage = sage;
        this.sprovince = sprovince;
        this.stuition = stuition;
    }

    public int getSid() {
        return sid;
```

```
    }

    public void setSid(int sid) {
        this.sid = sid;
    }

    public String getSname() {
        return sname;
    }

    public void setSname(String sname) {
        this.sname = sname;
    }

    public int getSage() {
        return sage;
    }

    public void setSage(int sage) {
        this.sage = sage;
    }

    public String getSprovince() {
        return sprovince;
    }

    public void setSprovince(String sprovince) {
        this.sprovince = sprovince;
    }

    public int getStuition() {
        return stuition;
    }

    public void setStuition(int stuition) {
        this.stuition = stuition;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        student student = (student) o;
        return sid == student.sid &&
                sage == student.sage &&
                stuition == student.stuition &&
                Objects.equals(sname, student.sname) &&
                Objects.equals(sprovince, student.sprovince);
    }

    @Override
    public int hashCode() {
        return Objects.hash(sid, sname, sage, sprovince, stuition);
```

```
    }

    @Override
    public String toString() {
        return "student{" +
                "sid=" + sid +
                ", sname='" + sname + '\'' +
                ", sage=" + sage +
                ", sprovince='" + sprovince + '\'' +
                ", stuition=" + stuition +
                '}';
    }
}
```

dao层

studentDao接口

```
public interface studentDao {
    //查询所有学生信息
    public List<student> findAllStu() throws SQLException;
    //删除学生信息
    public void deleteStu(int id) throws SQLException;
    //查询一条学生信息
    public student findOneStu(int id) throws SQLException;
    //更新学生信息
    public void  updateStu(student student) throws SQLException;
    //添加一条数据
    public void addStu(student student) throws SQLException;

}
```

studentDaoImpl类

```
public class studentDaoImpl implements studentDao {

    //创建queryRunner对象
    QueryRunner queryRunner = new QueryRunner(JDBCUtils.getDataSource());

    @Override
    //查询所有学生信息
    public List<student> findAllStu() throws SQLException {
        List<student> studentList = null;
        String sql = "select * from student";
        studentList = queryRunner.query(sql,new BeanListHandler<>(student.class));
        return studentList;
    }

    @Override
    //删除学生信息
    public void deleteStu(int id) throws SQLException {
        String sql = "delete  from student where sid = ?";
        queryRunner.update(sql,id);
```

```
    }

    @Override
    //查询一条学生信息
    public student findOneStu(int id) throws SQLException {
        student student = null;
        String sql = "select * from student where sid = ?";
        student = queryRunner.query(sql,new BeanHandler<>(student.class),id);
        return student;
    }

    @Override
    //更新
    public void updateStu(student student) throws SQLException {
        String sql = "update student set sname=? ,sage =? ,sprovince=?,stuition=? where
sid=?";

 queryRunner.execute(sql,student.getSid(),student.getSname(),student.getSage(),student.
getSprovince(),student.getSprovince());
    }

    @Override
    //添加一条数据
    public void addStu(student student) throws SQLException {
        String sql = "insert into student(sid,sname,sage,sprovince,stuition)
values(?,?,?,?,?)";

 queryRunner.update(sql,student.getSid(),student.getSname(),student.getSage(),student.g
etSprovince(),student.getStuition());
    }
}
```

service层

studentService接口

```
public interface studentService {
    //查询所有学生信息
    public List<student> getAllStu();
    //删除学生信息
    public void deleteStu(int id);
    //查询一条学生信息
    public student findOneStu(int id);
    //更新学生信息
    public void updateStu(student student);
}
```

studentServiceImpl类

```
public class studentServiceImpl implements studentService {
    studentDao studentDao = new studentDaoImpl();
```

```
    @Override
    //查询所有学生信息
    public List<student> getAllStu() {
        List<student> studentList = null;
        try {
            studentList = studentDao.findAllStu();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return studentList;
    }


    @Override
    //删除学生信息
    public void deleteStu(int id) {
        try {
            studentDao.deleteStu(id);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    @Override
    public student findOneStu(int id) {
        student student = null;
        try {
            student = studentDao.findOneStu(id);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return student;
    }

    @Override
    public void updateStu(student student) {
        try {
            studentDao.updateStu(student);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

}
```

test测试

```
public class test {

    studentDao studentDao = new studentDaoImpl();
```

```
    @Test
    //获取链接的测试
    public void  jdbcUtilsTest() throws SQLException {
     Connection connection = JDBCUtils.getConnection();
        System.out.println("======================="+ connection);
    }

    @Test
    //查询所有学生的信息
    public void findAllStuTest() throws SQLException {
        List<student>  studentList = studentDao.findAllStu();
        for (student str:studentList){
            System.out.println(str);
        }
    }

    @Test
    //删除学生信息
    public void deleteStuTest() throws SQLException {
        studentDao.deleteStu(7);
        List<student>  studentList = studentDao.findAllStu();
        for (student str:studentList){
            System.out.println(str);
        }
    }

    @Test
    //查询一条学生信息
    public void findOneStuTest() throws SQLException {
      student student =  studentDao.findOneStu(1);
        System.out.println(student);

    }

    @Test
    //更新
    public  void updateStuTest() throws SQLException {
        student student1 = new student(3,"二狗",23,"山西",23);
        studentDao.equals(student1);
        System.out.println(student1);
    }

    @Test
    //添加
    public void addStuTest() throws SQLException {
        student student = new student("二狗",23,"山西",34);
        studentDao.addStu(student);
    }
}
```

controller层

```java
@WebServlet("/studentServlet")
public class studentServlet extends HttpServlet {

    studentService studentServlet = new studentServiceImpl();

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        doGet(request, response);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        /*
            1、如何使一个Servlet处理多个请求
            2、如何在发送一个get/post请求后，执行某个方法?
            3、接收到请求后Servlet处理哪个请求呢? 在请求中添加参数
        */
        /*if (method.equals("queryAllStu")){
            queryAllStu(request, response);
        }else if(method.equals("deleteStu")){
            deleteStu(request, response);
        }*/
        String  method = request.getParameter("method");
        switch (method){
            case "queryAllStu":
                queryAllStu(request, response);
                break;
            case "deleteStu":
                deleteStu(request, response);
                break;
            case "findOneStu" :
                findOneStu(request, response);
                break;
            case "updateStu":
                updateStu(request, response);
                break;
        }
    }

    //查询所有学生的信息
    public void queryAllStu(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        List<student> studentList =  studentServlet.getAllStu();
        request.setAttribute("studentList",studentList);
        request.getRequestDispatcher("/studentList.jsp").forward(request,response);
    }

    //删除一条学生信息
    public void deleteStu(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        //获取jsp页面传递的参数
        String sid = request.getParameter("sid");
        //将(String类型的)id转为int类型
```

```
        int id = Integer.valueOf(sid);
        //执行操作
        studentServlet.deleteStu(id);
        queryAllStu(request, response);
    }

    //查询一条学生信息
    public void findOneStu(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        String sid = request.getParameter("sid");
        int id = Integer.valueOf(sid);
        //执行操作,获取一条学生信息
        student student =studentServlet.findOneStu(id);
        //将学生信息放到request中
        request.setAttribute("student",student);
        //将request(请求)转发到updateStu.jsp中
        request.getRequestDispatcher("/updateStu.jsp").forward(request,response);
    }

    //更新学生信息
    public void updateStu(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        System.out.println("aaaaaaaaaaaaaaaaaaaaaaaaaaaaa");
        String sid = request.getParameter("sid");
        int id = Integer.valueOf(sid);
        String name = request.getParameter("sname");
        name = new String(name.getBytes("iso-8859-1"),"utf-8");
        String sage = request.getParameter("sage");
        int age = Integer.valueOf(sage);
        String province = request.getParameter("sprovince");
        province =new String(province.getBytes("iso-8859-1"),"utf-8");
        String stuition = request.getParameter("stuition");
        int tuition = Integer.valueOf(stuition);
        student student = new student(id,name,age,province, tuition);
        studentServlet.updateStu(student);
        System.out.println(student);
        queryAllStu(request, response);


    }
}
```

jsp页面

index.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
  <head>
    <title>首页</title>
  </head>
  <body>
  <h2>
      <a href="/studentServlet?method=queryAllStu">学生信息列表</a>

  </h2>
  </body>
</html>
```

studentList.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>学生信息列表</title>
</head>
<body>
    <h2>学生信息列表</h2>
    <table border="1px" cellspacing="0px">
        <%--表头--%>
        <thead>
            <th>id</th>
            <th>姓名</th>
            <th>年龄</th>
            <th>省份</th>
            <th>学费</th>
            <th>操作</th>
            <th>操作</th>
        </thead>
        <%--表体：将学生信息放在表体中--%>
        <tbody>
            <%

                List<student> studentList = (List<student>)
request.getAttribute("studentList");
                //使用for()循环来将数据放到每行中
                for (student student:studentList){
             %>

            <%--表行--%>
            <tr>
                <td><%=student.getSid()%></td>
                <td><%=student.getSname()%></td>
                <td><%=student.getSage()%></td>
                <td><%=student.getSprovince()%></td>
                <td><%=student.getStuition()%></td>
                <td><a href="/studentServlet?method=deleteStu&sid=
<%=student.getSid()%>">删除</a></td>
```

```
                <td><a href="/studentServlet?method=findOneStu&sid=
<%=student.getSid()%>">修改</a></td>
            </tr>

            <%
                }
            %>

        </tbody>
    </table>

</body>
</html>
```

updateStu.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>修改学生信息</title>
</head>
<body>

    <%
        //首先需要从request中获取选中的学生信息
        student student =(student) request.getAttribute("student");
    %>
    <%--修改之后需要进行提交，用from表单--%>
    <form action="/studentServlet" method="post">
        <input type="hidden" name="method" value="updateStu">
        id  :<input type="hidden" name="sid" value="<%=student.getSid()%>" ><br><br>
        姓名: <input type="text" name="sname"  value="<%=student.getSname()%>" ><br><br>
        年龄: <input type="text" name="sage"  value="<%=student.getSage()%>" ><br><br>
        省份: <input type="text" name="sprovince"  value="<%=student.getSprovince()%>" >
<br><br>
        学费: <input type="text" name="stuition"  value="<%=student.getStuition()%>" >
<br><br>
        <input type="submit" value="提交">
    </form>
</body>
</html>
```

MVC案例

架构分析

DAO层

一个Servlet处理多个请求

(模糊)查询