

MVC Servlet JavaBean

张海宁

贵州大学

hnzhang1@gzu.edu.cn

April 14, 2018

Overview

1 MVC

2 servlet

3 java bean

4 Appendix

MVC

什么是 MVC

MVC 代表Model-View-Controller（模型-视图-控制器）模式。这种模式应用于应用程序（包括但不限于 web）的分层开发。

在 web 开发中，这种模型将 web 应用分成了三个部分：

- Model

web 应用核心功能

- ① 业务逻辑 (Java Bean)
- ② 数据库操作 (JDBC)

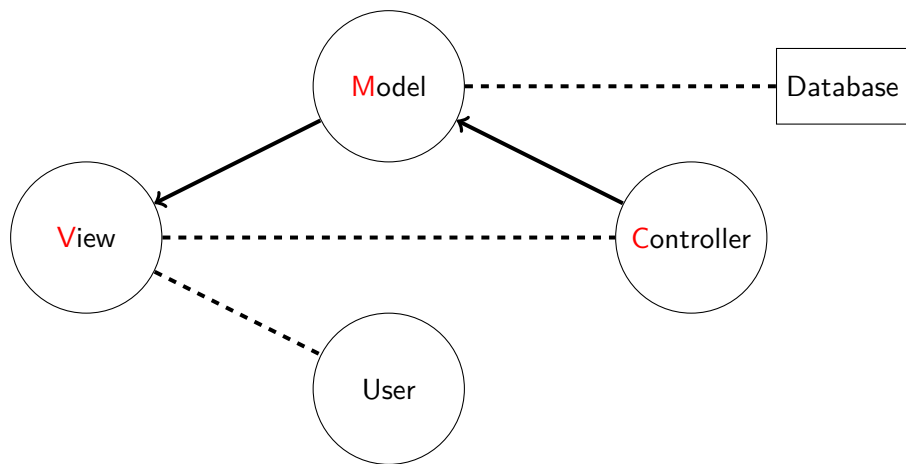
- View

主要指与用户的交互界面，通过jsp 或 html来构建，负责接收用户输入并转交给控制器。

- Controller

负责接收用户的请求，并转发到模型去处理。(servlet)

MVC 模型



MVC 的具体实现

在不引入框架的情况下:

M/V/C	technology
V	jsp,html
C	servlet
M	java bean

Table: MVC 的无框架实现

servlet

servlet 是什么

servlet 是运行在 web 服务器端的 java 程序。

- ① 封装了对 HTTP 请求的处理
- ② 运行需要 servlet 容器支持（比如 tomcat、weblogic 等）。

servlet 先于 jsp 产生，但是其代码组织形式是将 html 代码混杂在 java 代码中，给 web 程序设计带来了很多不便，比如负责网页设计的美工人员也需要学习 java 知识，才能进行页面的设计，在程序设计中，servlet 产生的动态网页又需要在代码中编写大量输出 html 标签的语句。基于以上原因，Sun 公司提出了 jsp 技术。jsp 是一种在 servlet 规范之上的动态网页技术，jsp 文件在第一次被请求时，会被编译成 servlet 文件，再通过容器调用 servlet 进行处理。

为什么使用 servlet

由于以下两个特点，

- ① servlet 由 java 语言编写，因此其可获得 java 语言的所有功能支持
- ② servlet 对 web 应用进行了封装，可对 http 请求进行相应的处理

再加上 jsp 承担了原来 servlet 中的负责页面显示的部分，因此现在的 servlet 就比较适合用来处理 web 系统的业务逻辑。

```
1 while(enu.hasMoreElements()){
2   String attr = enu.nextElement();
3   if(attr.equals("user")){
4     user = (String)session.getAttribute("user");
5     out.print("你好, "+user+"。");
6     out.print("<form action=\"logout.jsp\"
7               method=\"post\">");
8     out.print("<input type=\"submit\"
9               value=\"logout\"></form>");
10    break;
11  }
```

servlet 代码结构

在 eclipse 中，右键点击项目，选择新建，servlet，即可建立一个 servlet.

```
@WebServlet("/Reg")
public class Reg extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public Reg() {
        super();
    }

    protected void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {
    }

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

以注册为例，展示 servlet 的使用方法。原来的注册功能是通过 teacher_add.jsp 页面来实现的。

接下来将采用 Reg 这个 servlet 来实现相同的功能，主要步骤为：

- ① 更改 form 的 action 地址为 Reg 这个 servlet
- ② 完善 Reg 这个 servlet 里的 doPost() 方法，以处理表单通过 post 提交过来的数据

更改 form 的 action

```
<form action="Reg" method="post">
```

完成 Reg 里的 doPost() 方法

```
protected void doPost(... , ...) ... {  
    // 设置 request 传递过来值的编码，并获取传递值  
    request.setCharacterEncoding("utf-8");  
    String id = request.getParameter("staffid");  
    String name = request.getParameter("nm");  
    //get current date and time  
    LabDate ld = new LabDate();  
    String time = ld.getDtTm();  
    //write to database  
    String[] fields= {"id","name","logDate"};  
    String[] values= new String[3];  
    values[0]=id; values[1]=name; values[2]=time;  
    Db db = new Db();  
    int i = db.writeDb("teachers", fields, values);  
    db.getClose();  
}
```

完成 Reg 里的 doPost() 方法-II

```
protected void doPost (... , ...) ... {  
    // 设置 request 传递过来值的编码, 并获取传递值  
    // get current date and time  
    // write to database  
    int i = db.writeDb("teachers", fields, values);  
    db.getClose();  
    String rz="";  
    if(i==1) {  
        rz = "Done! Will return in 3 seconds.";  
    } else {  
        rz = "Sth wrong! Will return in 3 seconds.";  
    }  
    response.getWriter().print(rz);  
    response.setHeader("refresh", "3,URL=teacher.jsp");  
}
```

先观察下图：

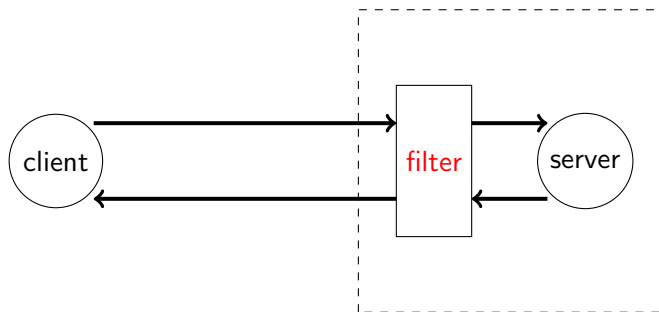


Figure: the position and function of a filter

可以看出，filter 是介于 client 和 server 端的一个“关卡”，其负责“过滤”信息流（这个过滤器可以有多个，形成过滤器链）。

创建一个 filter

创建一个 filter，禁止特定 id 的用户注册 (`chain.doFilter()`)。

创建过程与创建 servlet 类似。

FilterReg.java 的主要代码如下所示：

```
@WebFilter(filterName = "/FilterReg",
           urlPatterns = "/Reg")
public class FilterReg implements Filter {
    public void doFilter(ServletRequest request,
                        ServletResponse response, FilterChain chain) {
        HttpServletRequest req =
            (HttpServletRequest) request;
        String id = request.getParameter("staffid");
        System.out.println("staffid is: "+id);
        // pass the request along the filter chain
        chain.doFilter(request, response);
    }
}
```

chain.doFilter()

Pay attention to `chain.doFilter()`.

```
public void doFilter(ServletRequest request ,
    ServletResponse response , FilterChain chain) {
    // place your code here
    HttpServletRequest req = (HttpServletRequest) request;
    String id = request.getParameter("staffid");
    System.out.println("staffid is: "+id);

    // pass the request along the filter chain
    chain.doFilter(request , response);

    HttpServletResponse resp =
        (HttpServletResponse) response;
    resp.getWriter().print("add is done. Stay here.");
    //resp.setHeader("refresh" , "3,URL=index.jsp");
}
```

监听器

监听是对**特定事件**进行关注。

所谓监听器就是对特定事件的发生做出反应，比如你点击了什么按钮等等。

Listener 接口	Event 类
ServletContextListener	ServletContextEvent
ServletContextAttributeListener	ServletContextAttributeEvent
HttpSessionListener	HttpSeesionEvent
HttpSessionActivationListener	
HttpSessionAttributeListener	HttpSessionBindingEvent
HttpSessionBindingListener	
ServletRequestListener	ServletRequestEvent
ServletRequestAttributeListener	ServletRequestAttributeEvent

Table: Listener 接口与 Event 类

使用 servlet 处理文件上传 I

form 的主要代码为 (class_apply.jsp):

```
<form action="ServletClassArrange"
    enctype="multipart/form-data" method="post">
<table>
<tr><td>
    上传实验指导书: <input type="file" name="guideBook" >
    </td></tr>
<tr><td>
<button type="submit">submit</button>
    </td></tr>
</table>
</form>
```

使用 servlet 处理文件上传 II

Servlet 的主要代码为 (ServletClassArrange.java):

```
@WebServlet("/ServletClassArrange")
@MultipartConfig
public class ServletClassArrange extends HttpServlet {
    protected void doPost(.....){
        request.setCharacterEncoding("utf-8");
        Part pt = request.getPart("guideBook");
        String fName = pt.getSubmittedFileName();
        System.out.println("the file is:"+fName);
        System.out.println(
            this.getServletContext().getRealPath("/") + fName);
        pt.write("/Users/hainingzhang/Downloads/" + fName);
        System.out.println("success uploaded!");
    }
}
```

java bean

The End

Appendix

ppt、项目源代码及实验指导书的地址

- ① ppt
<https://github.com/gmsft/ppt/tree/master/javaweb>
- ② lab-java 项目
<https://github.com/gmsft/javaweb>
- ③ 实验指导书
暂无，拟放在 ppt 的仓库里