MVC Servlet JavaBean

张海宁

贵州大学

hnzhang1@gzu.edu.cn

April 23, 2018

Overview

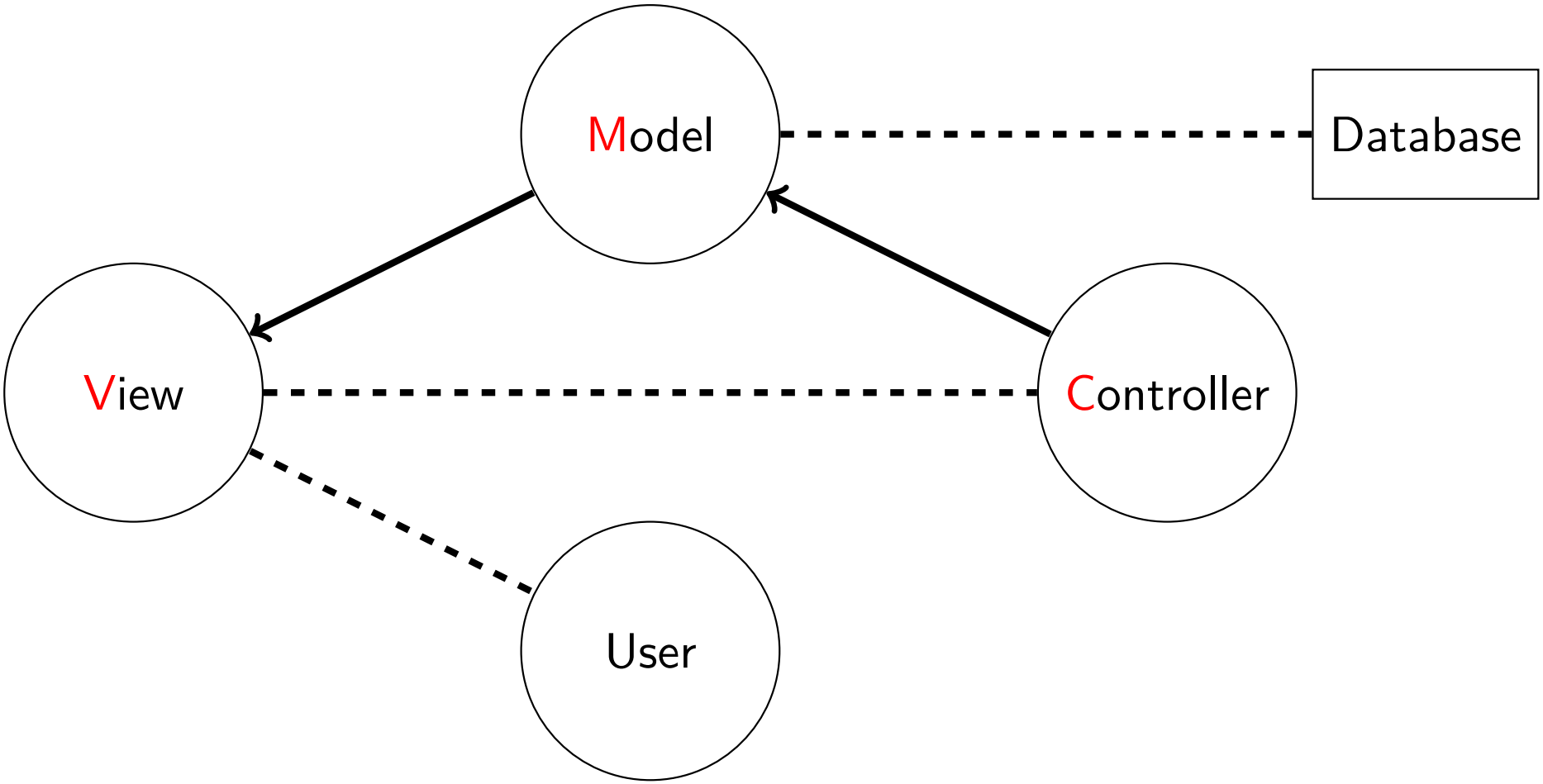
MVC

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

什么是 servlet

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。

servlet 代码片断

```
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    }
12  }
```

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.

```
1  @WebServlet("/Reg")
2  public class Reg extends HttpServlet {
3  private static final long serialVersionUID = 1L;
4     public Reg() {
5           super();
6        }
7    protected void doGet(HttpServletRequest req,
8        HttpServletResponse resp)
9        throws ServletException, IOException {
10   }
11    protected void doPost(HttpServletRequest req,
12       HttpServletResponse resp)
13       throws ServletException, IOException {
14    }
15 }
```

servlet 使用示例

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

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

完成 Reg 里的 doPost() 方法

```
1  protected void doPost(... , ...)... {
2  // 设置request传递过来值的编码，并获取传递值
3  request.setCharacterEncoding("utf-8");
4  String id = request.getParameter("staffid");
5  String name = request.getParameter("nm");
6  //get current date and time
7  LabDate ld = new LabDate();
8  String time = ld.getDtTm();
9  //write to database
10 String[] fields= {"id","name","logDate"};
11 String[] values= new String[3];
12 values[0]=id; values[1]=name; values[2]=time;
13 Db db = new Db();
14 int i = db.writeDb("teachers", fields, values);
15 db.getClose();
16 }
```

完成 Reg 里的 doPost() 方法-II

```
1  protected void doPost(... , ...)... {
2  // 设置request传递过来值的编码，并获取传递值
3  //get current date and time
4  //write to database
5  int i = db.writeDb("teachers", fields, values);
6  db.getClose();
7  String rz="";
8  if(i==1) {
9     rz = "Done! Will return  in 3 seconds.";
10 }else {
11    rz = "Sth wrong! Will return  in 3 seconds.";
12 }
13 response.getWriter().print(rz);
14 response.setHeader("refresh","3,URL=teacher.jsp");
15 }
```

servlet 过滤器

先观察下图：

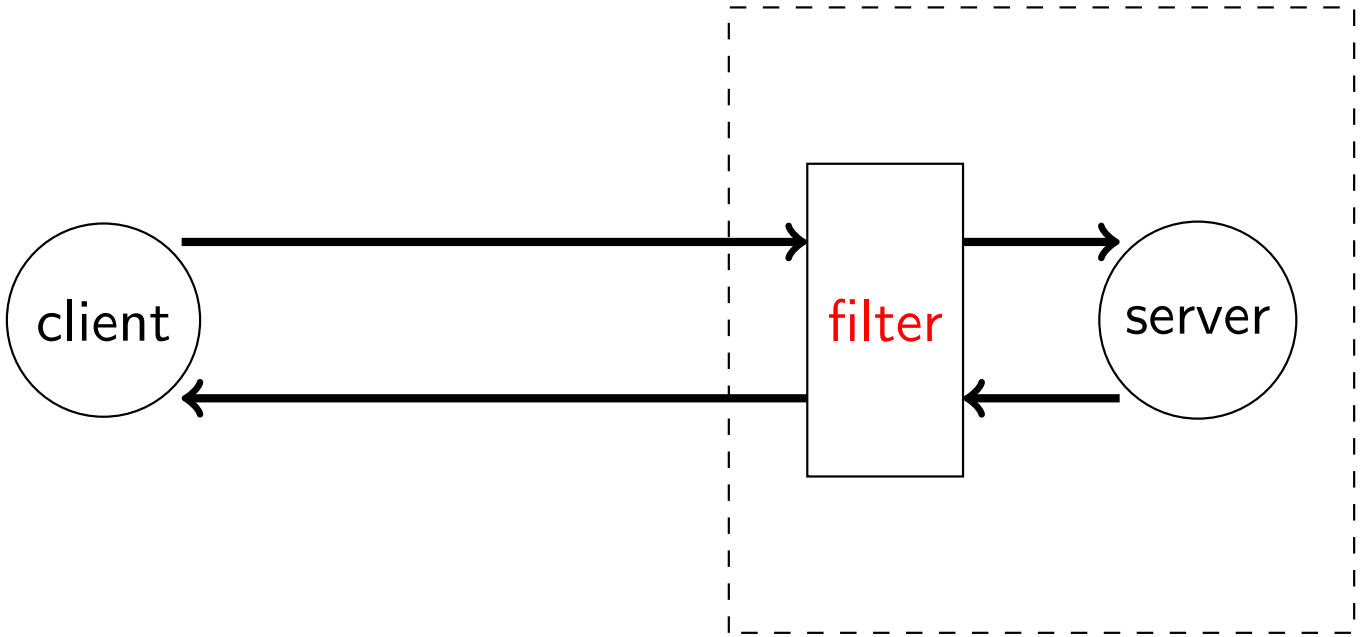


Figure: the position and function of a filter

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：

```
1  @WebFilter(filterName = "/FilterReg",
2          urlPatterns = "/Reg")
3  public class FilterReg implements Filter {
4     public void doFilter(ServletRequest request,
5     ServletResponse response, FilterChain chain)  {
6     HttpServletRequest req =
7          (HttpServletRequest) request;
8     String id = request.getParameter("staffid");
9     System.out.println("staffid is: "+id);
10    // pass the request along the filter chain
11    chain.doFilter(request, response);
12    }
13 }
```

chain.doFilter()

Pay attention to chain.doFilter().

```
1  public void doFilter(ServletRequest request,
2      ServletResponse response, FilterChain chain){
3  HttpServletRequest req =
4    (HttpServletRequest) request;
5  String id = request.getParameter("staffid");
6  System.out.println("staffid is: "+id);
7
8  // pass the request along the filter chain
9  chain.doFilter(request, response);
10
11 HttpServletResponse resp =
12     (HttpServletResponse) response;
13 resp.getWriter().print("add is done. Stay here.");
14 //resp.setHeader("refresh", "3,URL=index.jsp");
15 }
```

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
ServletContexAttributeListener	ServletContextAttributeEvent
HttpSessionListener HttpSessionActivationListener	HttpSeesionEvent
HttpSessionAttributeListener HttpSessionBindingListener	HttpSessionBindingEvent
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）：

```
1 <form action="ServletClassArrange"
2     enctype="multipart/form-data" method="post">
3
4 上传实验指导书:<input type="file" name="guideBook" >
5
6 <button type="submit">submit</button>
7
8 </form>
```

使用 servlet 处理文件上传 II

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

```
1  @WebServlet("/ServletClassArrange")
2  @MultipartConfig
3  public class ServletClassArrange
4      extends HttpServlet {
5   protected void doPost(......){
6    request.setCharacterEncoding("utf-8");
7    Part pt = request.getPart("guideBook");
8    String fName = pt.getSubmittedFileName();
9    System.out.println("the file is:"+fName);
10   System.out.println(
11    this.getServletContext().getRealPath("/")+fName);
12   pt.write("/Users/hainingzhang/Downloads/"+fName);
13   System.out.println("success uploaded!");
14  }
15 }
```

javabean

什么是 javabean

如果一个类满足以下条件：

1. All properties private (use getters/setters)
2. A public no-argument constructor
3. Implements Serializable

则就称这个类是一个 javabean。

serializable

如果想将一个 java 对象保存到文件中，那么这个对象必须是可序列化的(serializable)，也就是说这个对象的类是需要实现 Serializable 接口的。

Teacher 类实现序列化

```
1  import java.io.Serializable;
2
3  public class Teacher implements Serializable{
4      private static final long serialVersionUID = 1L;
5  }
```

为什么要使用 javabean

使用 javabean 可以使 html 代码和 java 代码尽可能的分离，将 java 代码单独封装成为一个处理某种业务逻辑的类，然后在 jsp 页面中调用此类。以简化 jsp 页面和提高 java 程序代码的重用性。

Teacher

定义 Teacher **类，该类为封装教师对象的** javabean。

```
1  //Teacher.java
2  package bean;
3  import java.io.Serializable;
4  public class Teacher implements Serializable{
5   private static final long serialVersionUID = 1L;
6   private String staffid,nm,logDate;
7   public Teacher() {}
8   public void setstaffid(String id){this.staffid=id;}
9   public void setnm(String name) {this.nm=name;}
10  public void setlogDate(String logDate) {
11    this.logDate=logDate;}
12  public String getstaffid() { return staffid; }
13  public String getnm() { return nm;  }
14  public String getlogDate() { return logDate;  }
15 }
```

与 javabean 相关的几个 jsp 标签

1. jsp:useBean
   实例化一个对象
2. jsp:setProperty
   为一个属性赋值
3. jsp:getProperty
   获取一个属性值

```
1  //teacher_add.jsp
2  <jsp:useBean id="teacher" class="bean.Teacher">
3    </jsp:useBean>
4  <jsp:setProperty name="teacher" property="nm"
5    value="贵州大学" />
6  <jsp:getProperty property="nm" name="teacher"/>
```

对 javabean 属性赋值与获取 javabean 属性 I

在 jsp 页面中，对 javabean 属性赋值与获取 javabean 属性。

```
1  //teacher_add.jsp
2 <% request.setCharacterEncoding("utf-8"); %>
3 <jsp:useBean id="teacher" class="bean.Teacher">
4   </jsp:useBean>
5 <jsp:setProperty name="teacher" property="*" />
6
7 <jsp:getProperty property="staffid" name="teacher"/>
8
9 <jsp:getProperty property="nm" name="teacher"/>
10
11 <jsp:getProperty property="logDate" name="teacher"/>
```

这种给 javabean 赋值的方式（第 5 行），要求 javabean 里的属性名字和 jsp 表单中的名字完全一致。

对 javabean 属性赋值与获取 javabean 属性 II

在 jsp 页面中，对 javabean 属性赋值与获取 javabean 属性。

```
1  //showCookie.jsp
2  <jsp:useBean id="student" class="bean.Student">
3     </jsp:useBean>
4
5  <jsp:setProperty name="student" property="name"
6     param="course"/>
7
8  Student name is :<jsp:getProperty name="student"
9     property="name"/>
```

作业

1. 使用 session 记录登陆状态
2. 练习 servlet 和 javabean 的使用
3. 学习 jsp+servlet+javabean 联合使用
   `https://github.com/danielniko/SimpleJspServletDB`

The End

Appendix

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
   放在 ppt 的仓库里

ServletRequest 和 HttpServletRequest

public interface HttpServletRequest
extends ServletRequest
Extends the ServletRequest interface to provide request information for HTTP servlets.