# EL&JSTL

张海宁

贵州大学

*hnzhang1@gzu.edu.cn*

April 22, 2018

# Overview

# EL

# 什么是 EL

EL 是指 jsp Expression Language。EL 是 jsp2.0 引入的，其目的就是为了简化 jsp 页面中的的一些参数获取操作（using html like tags）。
使用 EL 可以方便的读取存储在以下对象中的数据：

- JavaBean
- jsp 的一些内置对象
  - request
  - session
  - application
  - ...

## 语法结构

${ expression }

# EL 的隐含对象

与 jsp 的内置对象（参考本 ppt 第35,36页）类似的概念，可以直接使用。EL 的隐含对象可以分为以下三类：

- 页面上下文对象
- 访问作用域范围的隐含对象
- 访问环境信息的隐含对象

# 页面上下文对象

```
1 ${pageContext.request }
2 <% request.getMethod(); %>
3 ${pageContext.request.method }
4
5 ${pageContext.response }
6 ${pageContext.out }
7 ${pageContext.session }
8 ${pageContext.page }
```

# 访问作用域范围的隐含对象

```
1  <%
2   pageContext.setAttribute("university", "pgGzu", 1);
3   request.setAttribute("university", "reqGzu");
4   session.setAttribute("university", "sessionGzu");
5   application.setAttribute("university", "appGzu");
6  %>
7   ${pageScope.university }
8   ${requestScope.university }
9   ${sessionScope.university }
10  ${applicationScope.university }
11  ${university }
```

**结合** $university，**来看作用域的查找顺序**:

page>request>session>application

# 访问环境信息的隐含对象

```
1  ${param.course }
2  <%request.getParameter("course"); %>
3  <!--获取复选框的值  -->
4  ${paramValues.multiCheckBox[0] }
5  ${paramValues.multiCheckBox[1] }
6
7  ${header.connection }
8  ${header["connection"] }
9  ${header["User-Agent"] }
10
11 ${cookie.JSESSIONID }
12 ${cookie.JSESSIONID.value }
```

# param 示例

## the form in test.jsp

```
1 <form action="showCookie.jsp" method="post">
2 课程: <select name="course">
3 <option value="AnalogCircuit" >模电</option>
4 ...
5 </select>
6 备注: <input name="note" type="text">
7 <button type="submit">submit</button>
8 </form>
```

Question: 怎么在 showCookie.jsp 页面获取数据？

# param 示例

**get the form parameter of test.jsp in showCookie.jsp**

```
1 The course is: ${param.course}.
```

Question: 怎么在 showCookie.jsp 页面获取 cookie 相关的数据？

# EL 的操作符

与 jsp 的内置对象类似的概念，可以直接使用。

| Operator | Description | Operator | Description |
|---|---|---|---|
| . | 访问 bean 里的属性或 map 里的记录 | | |
| [] | 访问 array 或 list 里的元素 | () | 改变优先级 |
| + - * / | + - * / | % | 取余 |
| ==(eq) | 判断是否相等 | !=(nq) | 判断是否不相等 |
| <(lt) | 判断是否小于 | >(gt) | 判断是否大于 |
| <=(le) | 判断是否小于或相等 | >=(ge) | 判断是否大于或相等 |
| &&(and) | 与 | \|\|(or) | 或 |
| !(not) | 非 | empty | 判断是否为空 |

Table: Basic Operators in EL

# JSTL

# 什么是 JSTL

JSTL 是JavaServer Pages Standard Tag Library (JSTL) 的简称。JSTL 提供了一系列封装好了的 jsp 标签，可以取代在 jsp 页面中嵌入 java 代码的做法。
JSTL 实际上由 5 个功能不同的标签库组成：

1. 核心标签库
2. 格式标签库
3. SQL 标签库
4. XML 标签库
5. 函数标签库

# 核心标签库

核心标签库主要用于完成 JSP 页面的常用功能，包括：

- 表达式标签
  - $< c:out >, < c:set >, < c:remove >, < c:catch >$
- URL 标签
  - $< c:import >, < c:redirect >, < c:url >, < c:param >$
- 流程控制标签
  - $< c:if >, < c:choose >, < c:when >, < c:otherwise >$
- 循环控制标签
  - $< c:forEach >, < c:forTokens >$

# 核心标签库中各标签的作用

| tag | Description |
| --- | --- |
| < c : out > | 将表达式的值输出到 jsp 页面 |
| < c : set > | 在指定的作用域内定义变量或为变量赋值 |
| < c : remove > | 从指定的作用域中移除指定的变量 |
| < c : catch > | 捕获程序的异常 |
| < c : import > | 导入文件到 web 页面中 |
| < c : redirect > | 将 request 请求重定向到其他 URL |
| < c : url > | 构造一个 URL |
| < c : param > | 为其他标签提供参数信息 |
| < c : if > | 简单的条件判断 |
| < c : choose >, < c : when >, < c : otherwise > | 选择，相当于 switch |
| < c : forEach > | 遍例数组或集合类中的成员 |
| < c : forTokens > | 迭代字符串中由分隔符分隔的成员 |

# JSTL 的配置

1. 下载 JSTL 的 jar 文件
   http://tomcat.apache.org/download-taglibs.cgi#Standard-1.2.5
2. 将下载好的 taglibs-standard-impl-1.2.5.jar 文件，放到项目下的 WEB-INF 文件夹下的 lib 文件夹中
3. 在想使用 JSTL 标签的 jsp 页面中，定义引用的标签库和访问前缀

```
1 <%@ taglib prefix = "c"
2     uri = "http://java.sun.com/jsp/jstl/core" %>
```

# 表达式标签 <c:out>

<c:out> 标签用于将表达式的值输出到 jsp 页面中去。类似于 jsp 的 <%=exp%> 或 el 的 ${exp}。

<c:out value="" [escapeXml="true|false"] [default="defaultValue"] />

- value
  用于指定变量或表达式，可以使用 el
- escapeXml
  可选属性，默认为 true，转换。用于指定是否转换 value 中的特殊字符。比如将 < 转换为 <
- default
  可选属性，默认为空字符串。用于指定当 value 的值为 null 时，将要显示的默认值。

<c:out value="<br>newline" /> 这里 <br> 原样输出

<c:out value="<br>newline" escapeXml="false"/>这里 <br> 会被解析为 html 标记

# 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：

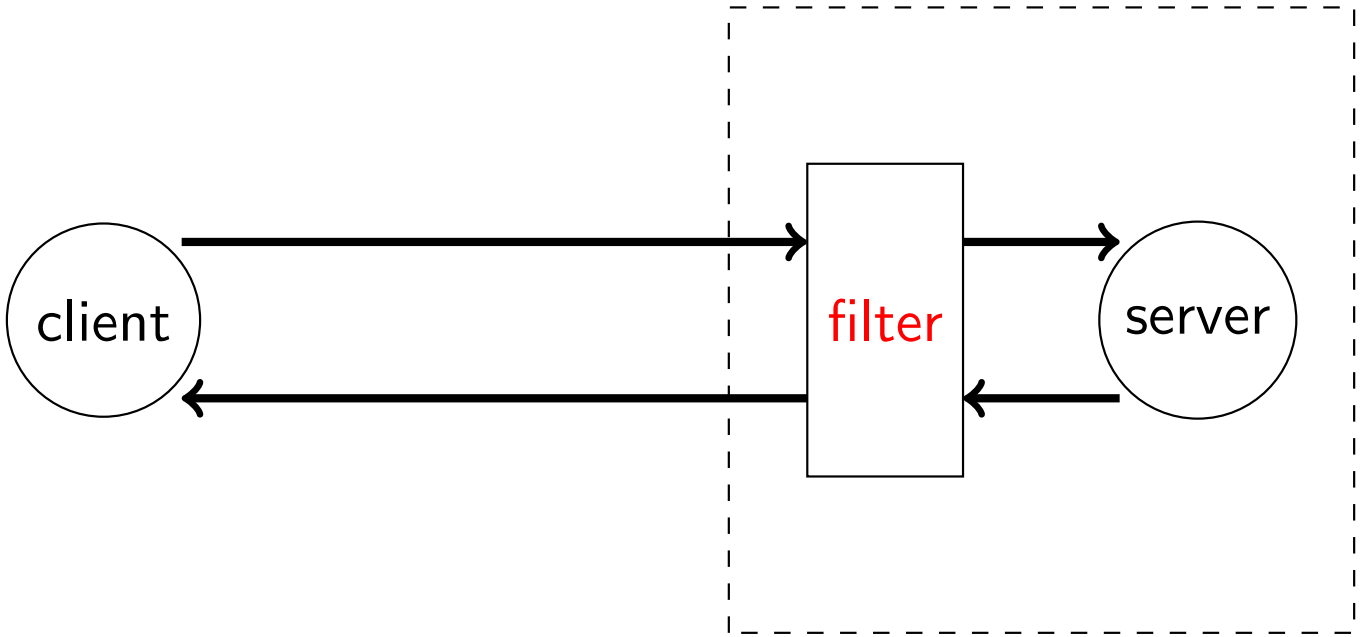


Figure: the position and function of a filter

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<br>HttpSessionActivationListener | HttpSeesionEvent |
| HttpSessionAttributeListener<br>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  <table>
4  <tr><td>
5  上传实验指导书:<input type="file" name="guideBook" >
6      </td></tr>
7  <tr><td>
8  <button type="submit">submit</button>
9      </td></tr>
10 </table>
11 </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需要是可序列化的。

## 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

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 <table>  <tr> <th>staffid</th> <th>nm</th>
7         <th>logDate</th> </tr>  <tr> <td>
8 <jsp:getProperty property="staffid" name="teacher"/>
9 </td><td>
10 <jsp:getProperty property="nm" name="teacher"/>
11 </td><td>
12 <jsp:getProperty property="logDate" name="teacher"/>
13 </td></tr></table>
```

# 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

# 内置对象

| Object | Description | Scope |
|---|---|---|
| request | 封装了客户端请求 | |
| response | 封装了服务器端的回应信息 | |
| out | 向 response 中写数据 | |
| session | 保存一次会话的信息 | 一次会话 |
| application | 保存应用程序中的公有数据 | 系统的全局变量 |
| config | | ServletConfig |
| pageContext | 获取 jsp 页面上下文，进而获取内置对象 | |
| page | 调用当前 jsp 页面中的变量或方法 | this 的同意词 |
| exception | 用来处理异常信息的 | |

Table: implicit objects of jsp

返回5

# 作用域

| Scope | Description |
|---|---|
| page | 当前的页面 |
| request | 当前的请求 |
| session | 当前的会话 |
| application | 当前网站 |

Table: scopes of jsp

返回5

## 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