第四章 EL和JSTL核心技术

4.1 EL表达式（熟悉）

4.1.1 基本概念

- EL（Expression Language）表达式提供了在JSP中简化表达式的方法，可以方便地访问各种数据并输出。

<%= %>

4.1.2 主要功能

当前页面有效 当前请求 当前会话 当前服务器

- 依次访问pageContext、request、session和application作用域对象存储的数据。
- 获取请求参数值。
- 访问Bean对象的属性。
- 访问集合中的数据。
- 输出简单的运算结果。

setAttribute, getAttribute

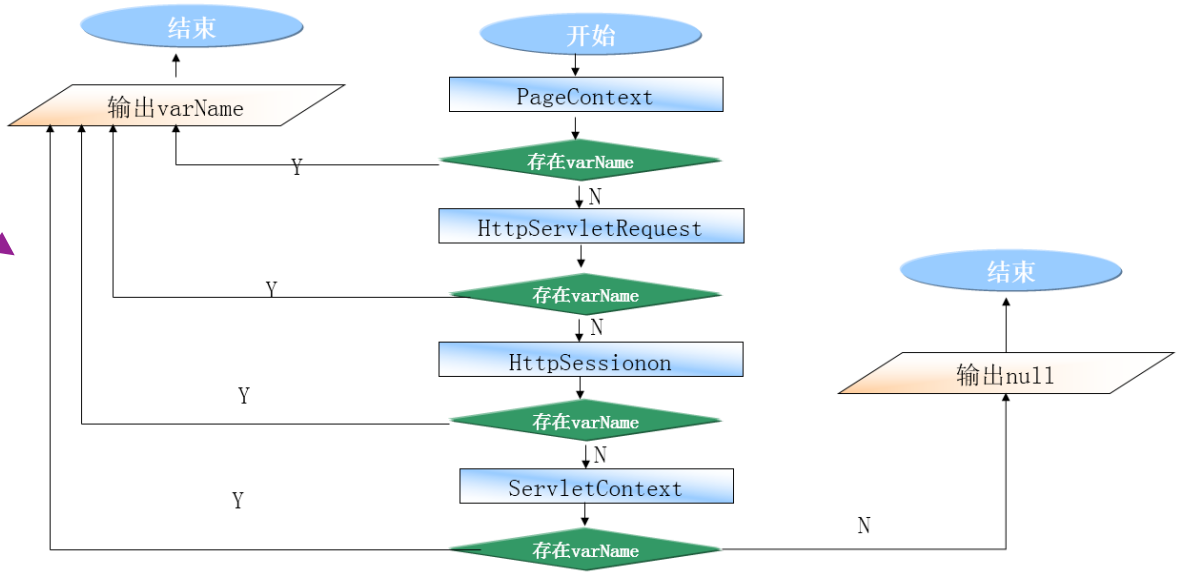
4.1.3 访问内置对象的数据

e.g.
name的数值为：${name}

（1）访问方式

- <%=request.getAttribute(“ varName”)%>
- 用EL实现: ${ varName }

（2）执行流程



4.1.4 访问请求参数的数据

- 在EL之前使用下列方式访问请求参数的数据
  request.getParameter(name);
  request.getParameterValues(name);
- 在EL中使用下列方式访问请求参数的数据
  param：接收的参数只有一个值。
  paramValues：接受的参数有多个值。

e.g.
姓名是：${param.name}
爱好是：${paramValues.hobby[0]}

```
<!-- 获取指定参数的数值 -->
${param.name}
<!-- 获取指定参数中指定下标的数值 -->
${paramValues.hobby[0]}
```

4.1.5 访问Bean对象的属性

（1）访问方式

- 方式一：${对象名.属性名}，例如：${user.name}
- 方式二：${对象名["属性名"]}，例如：${user["name"]}

（2）主要区别

- 当要存取的属性名中包含一些特殊字符，如：.或,等并非字母或数字的符号，就一定要使用[]而不是.的方式
- 使用[]的方式可以动态取值，具体方式如下：

```
<%
    request.setAttribute("prop","age");
%>
<!-- 相当于表达式中写一个变量 -->
${ user[prop] }
```

4.1.6 访问集合中的数据

```
<!-- student为ArrayList类型的对象 -->
${student[0].name}
```

4.1.7 常用的内置对象

类别	标识符	描述
JSP	pageContext	PageContext 处理当前页面
作用域	pageScope	同页面作用域属性名称和值有关的Map类
	requestScope	同请求作用域属性的名称和值有关的Map类
	sessionScope	同会话作用域属性的名称和值有关的Map类
	applicationScope	同应用程序作用域属性的名称和值有关的Map类
请求参数	param	根据名称存储请求参数的值的Map类
	paramValues	把请求参数的所有值作为一个String数组来存储的Map类
请求头	header	根据名称存储请求头主要值的Map类
	headerValues	把请求头的所有值作为一个String数组来存储的Map类
Cookie	cookie	根据名称存储请求附带的cookie的Map类
初始化参数	initParam	根据名称存储Web应用程序上下文初始化参数的Map类

4.1.8 常用的运算符

（1）常用的算术运算符

算术运算符	说 明	范 例	运算结果
+	加	${1+2}	3
-	减	${2-1}	1
*	乘	${2*3}	6
/ 或 div	除	${16/5}或${16div5}	3.2
% 或 mod	取余	${16%5}或${16mod5}	1

保留小数

（2）常用的关系运算符

关系运算符	说 明	范 例	运算结果
==或eq	等于	${1==2}或${1 eq 2}	false
!= 或ne	不等于	${2!=1}或${1 ne 2}	true
< 或lt	小于	${2<3}或${2 lt 3 }	true
> 或 gt	大于	${16>5}或${16 gt 5}	true
<= 或 le	小于等于	${16<=5}或${16 le 5}	false
>= 或 ge	大于等于	${16>=5}或${16 ge 5}	true

（3）常用的逻辑运算符

逻辑运算符	说 明	范 例	运算结果
&& 或 and	与运算	${true&&true}或${true and true}	true
|| 或or	或运算	${true\|\|false}或${true or false}	true
! 或not	非运算	${!true}或${not true }	false

（4）条件运算符

```
${条件表达式? 语句1 : 语句2}
```

（5）验证运算符

```
${empty 表达式}
```

返回布尔值判断表达式是否为"空"值，`null`值、无元素的集合或数组、长度为零的`String`被认为是空值。

4.2 JSTL标签（熟悉）

JSP标准标签库

4.2.1 基本概念

JSP标准标签库

- JSTL(JSP Standard Tag Library) 被称为JSP标准标签库。
- 开发人员可以利用这些标签取代JSP页面上的Java代码，从而提高程序的可读性，降低程序的维护难度。

4.2.2 使用方式

- 下载JSTL的jar包并添加到项目中，下载地址为：https://tomcat.apache.org/download-taglibs.cgi
- 在JSP页面中使用taglib指定引入jstl标签库，方式为：

```
<!-- prefix属性用于指定库前缀 -->
<!-- uri属性用于指定库的标识 -->
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

4.2.3 常用核心标签

core — c

(1) 输出标签

```
<c:out></c:out> 用来将指定内容输出的标签
```

```
<c:out value="Hello World"></c:out>   // Hello World
```

(2) 设置标签

```
<c:set></c:set> 用来设置属性范围值的标签
```

```
<c:set var="name" value="zhangfei" scope="page"></c:set>
<c:out value="${name}"></c:out>  // zhangfei
<jsp:useBean id="person" class="com.lagou.demo01.Person" scope="page"></jsp:useBean>
<c:set property="name" value="guanyu" target="${person}"></c:set>
<c:set property="age" value="35" target="${person}"></c:set>
<c:out value="${person.name}"></c:out>  // guanyu
<c:out value="${person.age}"></c:out>    // 35
```

(3) 删除标签

```
<c:remove></c:remove> 用来删除指定数据的标签
```

```
<%--设计一个属性值并打印--%>
<c:set var="name" value="liubei" scope="page"></c:set>
<c:out value="${name}"></c:out>  // liubei
<hr>
<%--删除这个属性值后在此打印--%>
<c:remove var="name" scope="page"></c:remove>
<c:out value="${name}" default="无名"></c:out>  // 无名
```

(4) 单条件判断标签

```
<c:set var="age" value="20" scope="page"></c:set>
<c:out value="${age}"></c:out>
<c:if test =“EL条件表达式”>
    满足条件执行
</c:if >
```

```
<c:if test="${age >= 18}">
    <c:out value="已经成年了！"></c:out>   // 已经成年了！
</c:if>
```

(5) 多条件判断标签

```
<c:choose >
    <c:when test =“EL表达式”>
        满足条件执行
    </c:when>
    …
    <c:otherwise>
        不满足上述when条件时执行
    </c:otherwise>
</c:choose >
```

```
<%--设置一个变量 代表考试的成绩 并指定数值--%>
<c:set var="score" value="60" scope="page"></c:set>
<c:out value="${score}"></c:out>
<hr>

<%--进行多条件判断和处理--%>  // 60分万岁，多一分浪费！
<c:choose>
  <c:when test="${score > 60}">
    <c:out value="成绩不错，继续加油哦！"></c:out>
  </c:when>
  <c:when test="${score == 60}">
    <c:out value="60分万岁，多一分浪费！"></c:out>
  </c:when>
  <c:otherwise>
    <c:out value="革命尚未成，同志仍需努力！"></c:out>
  </c:otherwise>
</c:choose>
```

(6) 循环标签

```
<c:forEach var="循环变量" items="集合">
    ...
</c:forEach>
```

```
指定起始和结尾位置 从下标1开始 到3结束 包含1和3<br>
<c:forEach var="ts" items="${sArr}" begin="1" end="3">
   <c:out value="${ts}"></c:out>
</c:forEach>
```

4.2.4 常用函数标签

```
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
```

4.2.5 常用格式化标签

```
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
```

4.2.6 自定义标签

- 如果上面几个标签不能满足需求，程序员也可以自定义标签，步骤如下：
- 编写标签类继承SimpleTagSupport类或TagSupport类并重写doTag方法或doStartTag方法。

```
public class HelloTag extends SimpleTagSupport {

    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public void doTag() throws JspException, IOException {
        JspWriter out = this.getJspContext().getOut();
        out.println("自定义标签的参数为：" + name);
    }
}
```

- 定义标签库文件（tld标签库文件）并配置标签说明文件到到WEB-INF下：

```
<tag>
    <name>helloTag</name>
    <tag-class>com.lagou.demo02.HelloTag</tag-class>
    <body-content>empty</body-content>
    <attribute>
        <name>name</name>
        <required>true</required>
    </attribute>
</tag>
```

- 在JSP中添加taglib指令引入标签库使用：

```
<%@ taglib prefix="hello" uri="http://lagou.com" %>
```