**实验十 表达式语言**

一、实验目的

1. 了解表达式语言的功能；

2. 掌握表达式语言的使用。

二、实验原理

表达式语言（EL）是JSP 2.0新增的功能。它是一种简洁的数据访问语言，通过它可以在JSP页面中方便地访问应用程序数据，无需使用JSP表达式（<%=和%>）。EL的目标是使动态网页的设计、开发和维护更加容易。

EL的特点是访问数据可以使用点（.）运算符或方括号（[]）运算符；可以访问隐含对象；有自己的运算符构造表达式；可以使用Java语言编写的函数等。

三、实验内容与步骤

**1、完成operator.jsp，实现基本的表达式语言的算术运算符的使用，其中包括加(+)、减(-)、乘(\*)、除(/ 或 div)以及取余(% 或 mod)。（15分）**

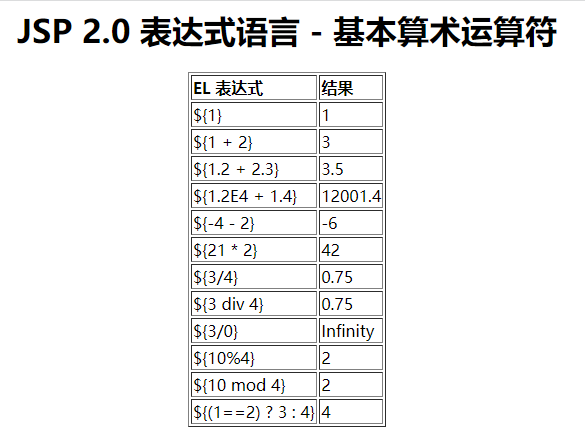


图10-1 operator.jsp的运行效果

**（1）代码**

<%@ page contentType="text/html;charset=gb2312"%>

<html>

<head>

<title>JSP 2.0 Expression Language - Basic Arithmetic</title>

</head>

<body>

**<h1 align="center">JSP 2.0 表达式语言 - 基本算术运算符</h1>【1分】**

**<table border="1" align="center">【2分】**

**<thead>**

**<td><b>EL 表达式</b></td>**

**<td><b>结果</b></td>**

**</thead>【1分】**

**<tr>**

**<td>\${1}</td>**

**<td>${1}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${1 + 2}</td>**

**<td>${1 + 2}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${1.2 + 2.3}</td>**

**<td>${1.2 + 2.3}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${1.2E4 + 1.4}</td>**

**<td>${1.2E4 + 1.4}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${-4 - 2}</td>**

**<td>${-4 - 2}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${21 \* 2}</td>**

**<td>${21 \* 2}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${3/4}</td>**

**<td>${3/4}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${3 div 4}</td>**

**<td>${3 div 4}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${3/0}</td>**

**<td>${3/0}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${10%4}</td>**

**<td>${10%4}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${10 mod 4}</td>**

**<td>${10 mod 4}</td>**

**</tr>【1分】**

**<tr>**

**<td>\${(1==2) ? 3 : 4}</td>**

**<td>${(1==2) ? 3 : 4}</td>**

**</tr>【1分】**

</table>

</body>

</html>

**（2）运行截图**

**2、访问作用域变量（45分）**

编写名为EmployeeBean的JavaBean，其中包括3个属性：eno表示雇员号、ename表示雇员名和ecompany表示雇员公司名。

【步骤1】按照JavaBeans规范完成EmployeeBean.java，实现Employee实体类的属性和方法。**（10分）**

**（1）代码**

package com.bean;

public class EmployeeBean {

**private String eno = "";【1分】**

**private String ename = "";【1分】**

**private String ecompany = "";【1分】**

**public EmployeeBean() {【2分】**

}

public void setEno(String eno) {

this.eno = eno;

}

public void setEname(String ename) {

this.ename = ename;

}

public void setEcompany(String ecompany) {

this.ecompany = ecompany;

}

public String getEno() {

return eno;

}

public String getEcompany() {

return ecompany;

}

public String getEname() {

return ename;

}

}

【步骤2】编写JSP页面，通过表单输入雇员信息，将请求转发给EmployeeServlet。

**（2）代码（10分）**

<%@ page contentType="text/html;charset=gb2312"%>

<html>

<body>

请输入雇员信息:

**<form action="employee.do" method="post">【2分】**

<table>

<tr>

<td>雇员号:</td>

<td**><input type="text" name="eno"></td>【1分】**

</tr>

<tr>

<td>雇员名:</td>

<td>**<input type="text" name="ename"></td>【1分】**

</tr>

<tr>

<td>公司名:</td>

**<td><input type="text" name="ecompany"></td>【1分】**

</tr>

</table>

<input type="submit" value="提交">

</form>

</body>

</html>

【步骤3】完成EmployeeServlet.java，实现获取客户信息，存储在请求作用域中并转发到displayEmployee.jsp页面。**（15分）**

**（3）代码**

package com.servlet;

import javax.servlet.RequestDispatcher;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.bean.EmployeeBean;

public class EmployeeServlet extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

**String eno = request.getParameter("eno");【1分】**

**String ename = request.getParameter("ename");【1分】**

**String ecompany = request.getParameter("ecompany");【1分】**

**EmployeeBean employee = new EmployeeBean();【1分】**

**employee.setEno(eno); 【1分】**

**employee.setEname(ename); 【1分】**

**employee.setEcompany(ecompany); 【1分】**

**request.setAttribute("employee", employee); 【1分】**

**RequestDispatcher view = request.getRequestDispatcher("/displayEmployee.jsp");【1分】**

**view.forward(request, response); 【1分】**

}

}

【步骤4】完成displayEmployee.jsp页面，实现使用EL表达式显示用户的信息。**（10分）**

**（4）代码**

<%@ page contentType="text/html;charset=gb2312"%>

<html>

<body>

雇员的信息如下：

<br>

<ul>

**<li>雇员号:${employee.eno}【3分】**

**<li>雇员名:${employee.ename}【3分】**

**<li>公司名:${employee.ecompany}【3分】**

</ul>

</body>

</html>

**3、完成implicit.jsp页面，使用EL隐含对象实现表单，用于输入foo参数值并提交至本页面，显示参数foo的值、初始URL中的主机和端口Host、浏览器可接受的MIME类型Accept以及浏览器类型user-agent。（25分）**

**代码：**

<%@ page contentType="text/html;charset=gb2312"%>

<html>

<head>

<title>EL implicit objects</title>

</head>

<body>

<h1>JSP 2.0 表达式语言-隐含对象</h1>

<hr>

<blockquote>

<b>输入foo参数值</b>

**<form action="implicit.jsp" method="GET">【2分】**

**foo= <input type="text" name="foo" value="${param["foo"]}"> 【4分】**

<input type="submit">

</form>

<br>

<code>

<table border="1">

<thead>

<td><b>EL 表达式</b></td>

<td><b>结果</b></td>

</thead>

<tr>

<td>\${param.foo}</td>

**<td>${param.foo}&nbsp;</td>【2分】**

</tr>

<tr>

<td>\${param["foo"]}</td>

**<td>${param["foo"]}&nbsp;</td>【3分】**

</tr>

<tr>

<td>\${header["host"]}</td>

**<td>${header["host"]}</td>【3分】**

</tr>

<tr>

<td>\${header["accept"]}</td>

**<td>${header["accept"]}</td>【3分】**

</tr>

<tr>

<td>\${header["user-agent"]}</td>

**<td>${header["user-agent"]}</td>【3分】**

</tr>

</table>

</code>

</blockquote>

</body>

</html>

**4、完成collections.jsp页面，实现访问由CollectServlet.java传递过来的集合对象的元素。（15分）**

【步骤1】完成CollectServlet.java，实现存储国家和城市信息，并存储在请求作用域中转发至collections.jsp页面。**（10分）**

1. **代码**

package com.demo;

import java.util.\*;

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

@WebServlet("/CollectServlet")

public class CollectServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

**ArrayList<String> country = new ArrayList<>();【1分】**

**country.add("China");【1分】**

**country.add("England");【1分】**

**country.add("Russia");【1分】**

**HashMap<String, String> capital = new HashMap<String, String>();【1分】**

**capital.put("China", "北京");【1分】**

**capital.put("England", "伦敦");【1分】**

**capital.put("Russia", "莫斯科");【1分】**

**request.setAttribute("country", country); 【1分】**

**request.setAttribute("capital", capital); 【1分】**

**RequestDispatcher rd = request.getRequestDispatcher("/collections.jsp");【1分】**

rd.forward(request, response);

}

}

【步骤2】完成collections.jsp页面，实现显示国家首都和城市名称。**（5分）**

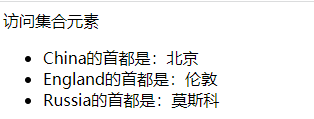


图10-2 collections.jsp的运行效果

1. **代码**

<%@ page contentType="text/html;charset=UTF-8"%>

<html>

<head>

<title>访问集合元素</title>

</head>

<body>

<p>访问集合元素</p>

<ul>

**<li>${country[0]}的首都是：${capital["China"]}【2分】**

**<li>${country[1]}的首都是：${capital["England"]}【2分】**

**<li>${country[2]}的首都是：${capital.Russia}【1分】**

</ul>

</body>

</html>