**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）**

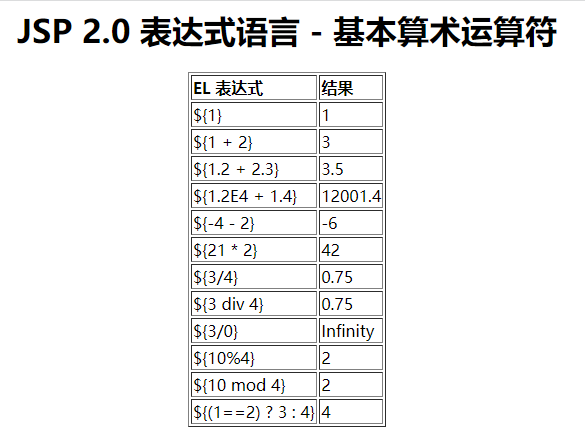


图10-1 operator.jsp的运行效果

**（1）代码（10分）**

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

pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>operator</title>

</head>

<body>

<h1>JSP表达式语言-基本算术运算符</h1>

<table border="1px">

<tr>

<th>EL表达式</th>

<th>结果</th>

</tr>

<tr>

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

<td>${1}</td>

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

</tr>

<tr>

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

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

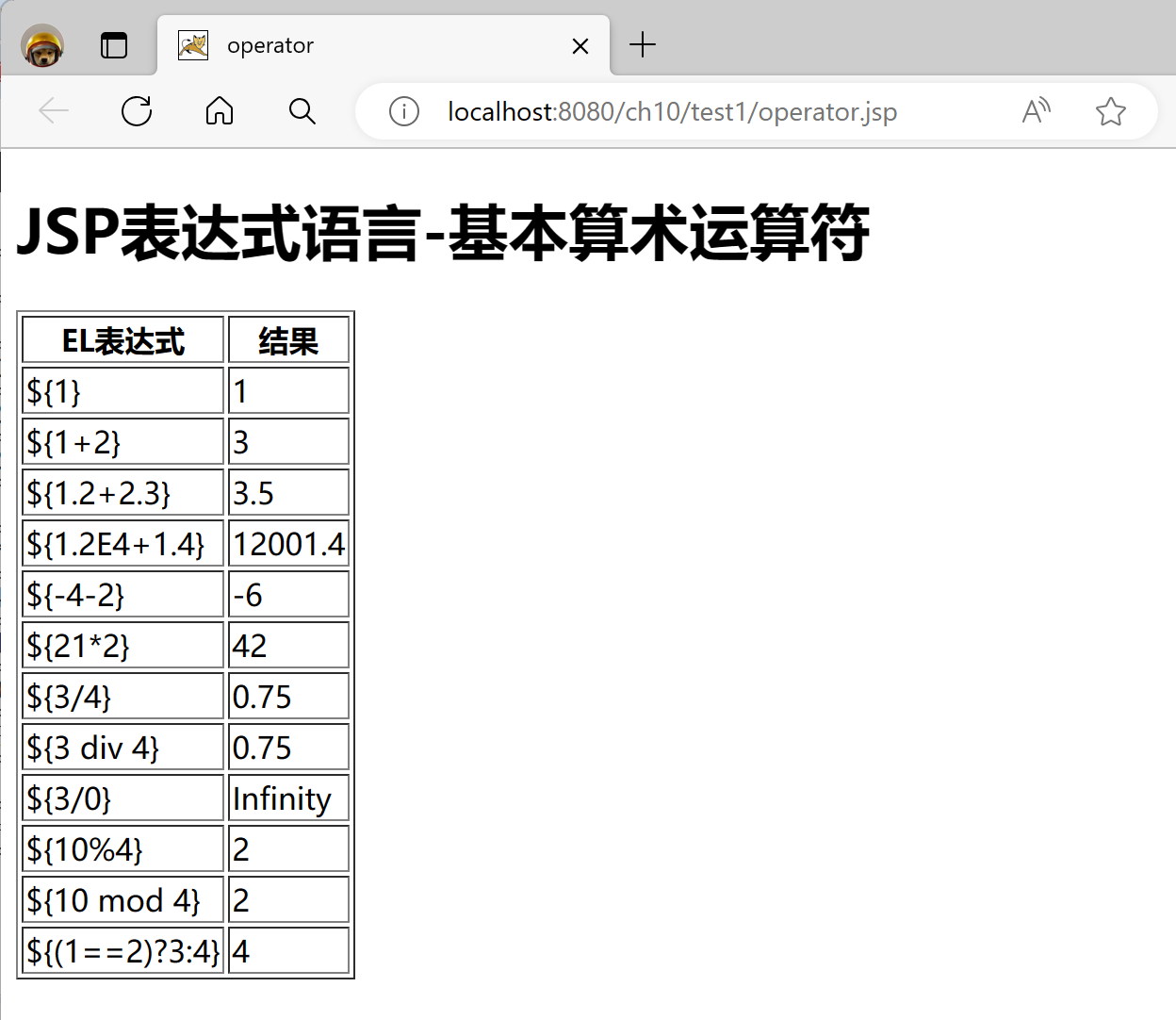
</tr>

</table>

</body>

</html>

**（2）运行截图（5分）**



**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.zyh;

public class EmployeeBean {

private String eno;

private String ename;

private String ecompany;

public EmployeeBean() {}

public EmployeeBean(String eno, String ename, String ecompany) {

super();

this.eno = eno;

this.ename = ename;

this.ecompany = ecompany;

}

public String getEno() {

return eno;

}

public void setEno(String eno) {

this.eno = eno;

}

public String getEname() {

return ename;

}

public void setEname(String ename) {

this.ename = ename;

}

public String getEcompany() {

return ecompany;

}

public void setEcompany(String ecompany) {

this.ecompany = ecompany;

}

}

【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 language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>employeeForm</title>

</head>

<body>

<form action="../EmployeeServlet" method="post">

雇员号:<input name="eno" /><br />

雇员名:<input name="ename" /><br />

雇员公司名:<input name="ecompany" /><br />

<input type="submit" />

</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.zyh;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/EmployeeServlet")

public class EmployeeServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

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

String eno = request.getParameter("eno");

String ename = request.getParameter("ename");

String ecompany = request.getParameter("ecompany");

EmployeeBean employee = new EmployeeBean(eno,ename,ecompany);

request.setAttribute("employee", employee);

request.getRequestDispatcher("test2/displayEmployee.jsp").forward(request, response);

}

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

doGet(request, response);

}

}

【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 language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>displayEmployee</title>

</head>

<body>

雇员号:${employee.eno}<br />

雇员名:${employee.ename}<br />

雇员公司名:${employee.ecompany}

</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 language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>implicit</title>

</head>

<body>

<form action="implicit.jsp" method="get">

foo=<input name="foo" value='${param["foo"]}' />

<input type="submit" /><br />

</form>

显示参数foo的值:${param.foo}<br />

初始URL中的主机和端口Host:${header.host}<br />

浏览器可接受的MIME类型Accept:${header.accept}<br />

浏览器类型user-agent:${header["user-agent"]}<br />

</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.zyh;

import java.io.IOException;

import java.util.HashMap;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/CollectServlet")

public class CollectServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

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

HashMap<String, String> collection = new HashMap<String, String>();

collection.put("China", "北京");

collection.put("England", "伦敦");

collection.put("Russia", "莫斯科");

request.setAttribute("collection", collection);

request.getRequestDispatcher("/test4/collections.jsp").forward(request, response);

}

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

doGet(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）**

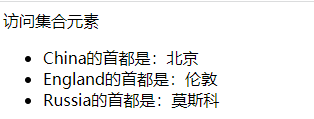


图10-2 collections.jsp的运行效果

1. **代码**

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

pageEncoding="UTF-8"%>

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

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>collections</title>

</head>

<body>

访问集合元素

<ul>

<c:forEach items="${collection}" var="country">

<li>${country.key }的首都是:${country.value }</li>

</c:forEach>

</ul>

</body>

</html>