**作业报告5**

1、开发一个带属性的标签类MathTag，该标签类有一个double型属性x，标签功能是求x的平方根，用math.jsp访问该标签，输出10、100、200的平方根值。

**1.1运行结果**



**1.2问题及解决方案**

<demo:mathtag x="10"/>写入数字时一定要加“”，否则会报错.

**1.3源代码**

1. **math.jsp**

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

<%@ taglib prefix=*"demo"* uri=*"/WEB-INF/taglib.tld"* %>

<html><head><title>求数的平方根</title></head>

<body>

<table border=*"1"*>

<tr align=*"center"*><td>x</td><td>sqrt(x)</td></tr>

<demo:mathtag x=*"10"*/>

<demo:mathtag x=*"100"*/>

<demo:mathtag x=*"200"*/>

</table>

</body></html>

**(2)MathTag.java**

**public** **class** MathTag **extends** SimpleTagSupport{

**double** x=0;

**public** **void** setX(**double** x){

**this**.x = x;

}

**public** **void** doTag() **throws** JspException, IOException{

JspWriter out = getJspContext().getOut();

**try**{

**if**(x<0){

out.print("<tr><td>"+x+"</td><td>error</td><tr>");

}**else**{

out.print("<tr><td>"+x+"</td><td>"+Math.*sqrt*(x)+"</td><tr>");

}

}**catch**(Exception e){

**throw** **new** JspException("Error in MathTag.doTag()");

}

}

}

**(3)taglib.tld中的标签**

<tag>

<name>mathtag</name>

<tag-class>tag.MathTag</tag-class>

<body-content>empty</body-content>

<attribute>

<name>x</name>

<required>true</required>

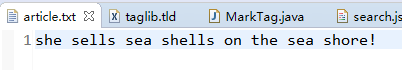
<rtexprvalue>true</rtexprvalue>

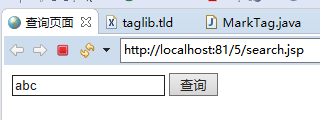
</attribute>

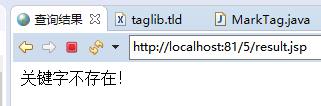
</tag>

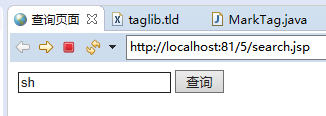
2、用自定义带标签体的标签类MarkTag实现如下功能：在一个查询页面search.jsp输入关键字查询存在某文章（article.txt）中是否存在，如果存在则在查询结果result.jsp显示该文章，并对所有关键字标红显示。

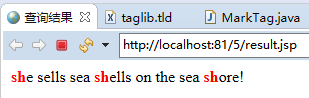
**2.1运行结果**











**2.2问题及解决方案**

1. 用标签读取文章时，要用绝对路径读取
2. 设置带标签体的标签时，由于在标签体中用到了EL表达式，taglib.tld中必须要设置<rtexprvalue>true</rtexprvalue>

**2.3源代码**

1. **result.jsp**

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

<%@ taglib prefix=*"demo"* uri=*"/WEB-INF/taglib.tld"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>查询结果</title>

</head>

<body>

<demo:mark search=*"*${search}*"*>${text}</demo:mark>

</body>

</html>

**(2)MarkTag.java**

**public** **class** MarkTag **extends** SimpleTagSupport{

**private** String search = **null**;

**public** **void** setSearch(String search){

**this**.search = search;

}

**public** **void** doTag() **throws** JspException, IOException{

String output="";

//读取文章

BufferedReader bw=**new** BufferedReader(**new** FileReader("D:/Web应用/5/WebContent/WEB-INF/article.txt"));//要填绝对路径

String s;

StringBuffer s1 = **new** StringBuffer();

**while**((s=bw.readLine())!=**null**) {

s1.append(s);

}bw.close();

String text = **new** String(s1);

**int** len = search.length();

**int** oldIndex=0, newIndex=0,flag=0;

**while**((newIndex = text.indexOf(search,oldIndex))>=0){

**if** (newIndex<oldIndex){

**break**;

}flag=1;

output+=text.substring(oldIndex,newIndex)+"<b><font color='red'>"+search+"</font></b>";

oldIndex = newIndex + len;

}

**if**(flag==0) {

output="关键字不存在！";

}**else** {

output+=text.substring(oldIndex);

}

getJspContext().setAttribute("text",output);

getJspBody().invoke(**null**);

}

}

**(3)taglib.tld中的标签**

<tag><name>mark</name>

<tag-class>tag.MarkTag</tag-class>

<body-content>scriptless</body-content>

<attribute>

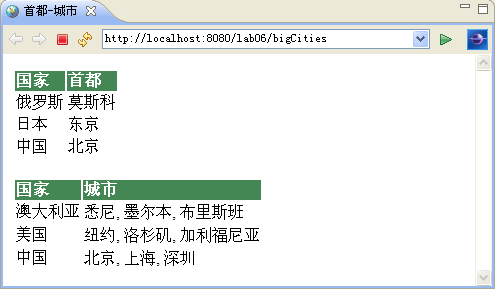
<name>search</name>

<required>true</required>

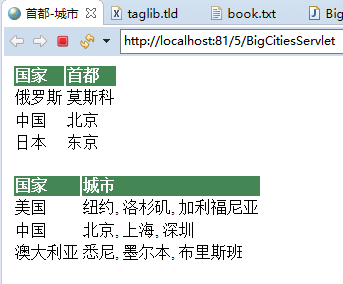
<rtexprvalue>true</rtexprvalue>

</attribute></tag>

3、编写BigCitiesServlet类和bigCities.jsp页面实现<c:forEach>标签对Map对象迭代。在BigCitiesServlet类中创建一个Map<String,String>对象capitals，键为国家名称，值为首都名称，添加几个对象（不少于三个国家和首都）。另外创建一个Map<String,String[]>对象bigCities，键为国家名称，值为String数组包含该国家的几个大城市。在doGet()中使用RequestDispatcher对象将请求转发到bigCities.jsp页面，在bigCities.jsp页面中显示这些国家的首都和城市，效果如下图所示：



**3.1运行结果**



**3.2问题及解决方案**

无

**3.3源代码**

**(1)bigCities.jsp**

<%@ 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>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>首都-城市</title>

</head>

<body>

<table>

<tr style="background:*#448755*;color:*white*;font-weight:*bold*">

<td>国家</td>

<td>首都</td>

</tr>

<c:forEach items=*"*${requestScope.capitals}*"* var=*"mapItem"*>

<tr>

<td>${mapItem.key}</td>

<td>${mapItem.value}</td>

</tr>

</c:forEach>

</table>

<br/>

<table>

<tr style="background:*#448755*;color:*white*;font-weight:*bold*">

<td>国家</td>

<td>城市</td>

</tr>

<c:forEach items=*"*${requestScope.bigCities}*"* var=*"mapItem"*>

<tr>

<td>${mapItem.key}</td>

<td>

<c:forEach items=*"*${mapItem.value}*"* var=*"city"* varStatus=*"status"*>

${city}<c:if test=*"*${!status.last}*"*>,</c:if>

</c:forEach>

</td>

</tr>

</c:forEach>

</table>

</body>

</html>

1. **BigCitiesServlet.java**

@WebServlet("/BigCitiesServlet")

**public** **class** BigCitiesServlet **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**public** BigCitiesServlet() {

**super**();

}

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

Map<String, String> capitals = **new** HashMap<String, String>();

capitals.put("俄罗斯", "莫斯科");

capitals.put("日本", "东京");

capitals.put("中国", "北京");

request.setAttribute("capitals", capitals);

Map<String, String[]> bigCities = **new** HashMap<String, String[]>();

bigCities.put("澳大利亚", **new** String[] { "悉尼", "墨尔本", "布里斯班" });

bigCities.put("美国", **new** String[] { "纽约", "洛杉矶", "加利福尼亚" });

bigCities.put("中国", **new** String[] { "北京", "上海", "深圳" });

request.setAttribute("capitals", capitals);

request.setAttribute("bigCities", bigCities);

RequestDispatcher rd = request.getRequestDispatcher("/bigCities.jsp");

rd.forward(request, response);

}

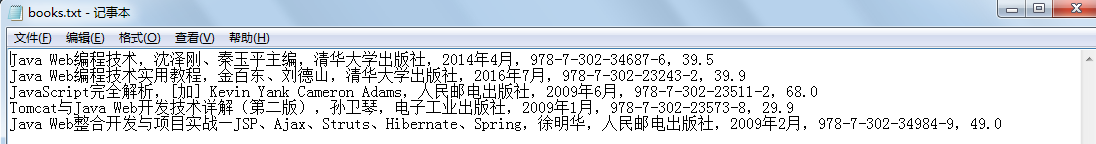
**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)**throws** ServletException, IOException {

doGet(request, response);

}

}

4、用JSP标签和MVC模式设计实现如下功能：输入书名、作者或ISBN号可以模糊查询出图书信息，以列表形式显示，其中图书信息存在book.txt文件或数据库中，包括书名、作者、出版社、出版时间、ISBN号、价格等，具体信息如下：



附图书的信息：

Java Web编程技术，沈泽刚、秦玉平主编，清华大学出版社，2014年4月，978-7-302-34687-6，39.5

Java Web编程技术实用教程，金百东、刘德山，清华大学出版社，2016年7月，978-7-302-23243-2，39.9

JavaScript完全解析，[加] Kevin Yank Cameron Adams，人民邮电出版社，2009年6月，978-7-302-23511-2，68.0

Tomcat与Java Web开发技术详解（第二版），孙卫琴，电子工业出版社，2009年1月，978-7-302-23573-8，29.9

Java Web整合开发与项目实战—JSP、Ajax、Struts、Hibernate、Spring，徐明华，人民邮电出版社，2009年2月，978-7-302-34984-9，49.0

注：本题的报告按附件的实验报告要求来写。

**4.1需求分析**

**1、功能描述**

输入书名、作者或ISBN号可以模糊查询出图书信息。

**2、业务处理流程及要求**

（1）输入的信息为空时，跳转回查询页面；不为空时，点击查询按钮查询信息；

（2）跳转到查询结果页面,若查到则显示查到的图书信息，若没查到则显示“检索失败！”；

（3）在查询页面点击超链接可以查看所有图书信息。

**3、输入信息**

书名、作者或ISBN号。

**4、输出信息**

要查询的图书信息。

**4.2开发步骤**

分析功能，有查询页面和查询结果页面，再增加一个显示所有图书信息的超链接。

**(1)编写Book.java:**创建Book对象，包含书名、作者、出版社、出版时间、ISBN号、价格6个属性。

**(2)查询页面：**可用第2题的查询页面做修改。表单中设置2行内容。第1行含有一个输入框，接收要查询的信息；含有一个按钮，点击查询按钮提交信息。第2行有个超链接，点击超链接可跳转至图书列表查看所有图书信息。

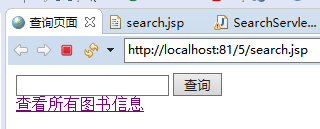
**(3)BooksServlet：**点击超链接后跳转至BooksServlet。读取book.txt的信息存到Book对象的List集合中，将对象存到请求作用域中转发到books.jsp，以列表形式显示所有图书信息。

**(4)SearchServlet：**可直接用第2题的SearchServlet.java.提交查询信息后跳转至查询信息处理页面。若信息为空则跳转回查询页面，否则跳转至查询结果页面。

**(5)查询结果页面：**可用第2题的查询结果页面做修改。用带标签体的用户自定义标签类BookTag实现检索图书信息并输出到页面的功能。

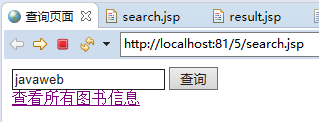
**4.3运行结果**

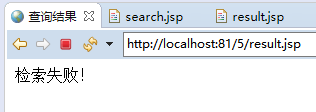
1. **查询页面**



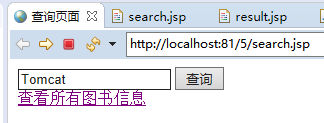
**2)显示所有图书信息页面**

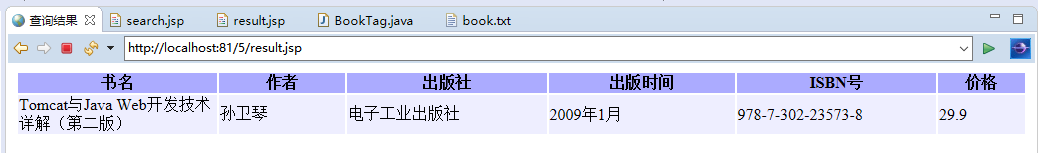
**3)查询图书信息**



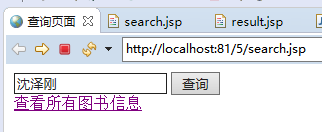


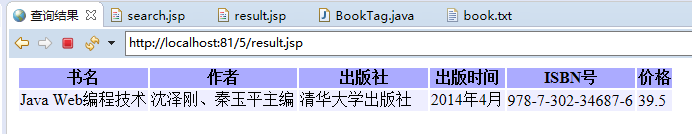
输入书名查询：



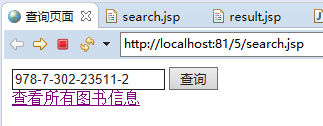


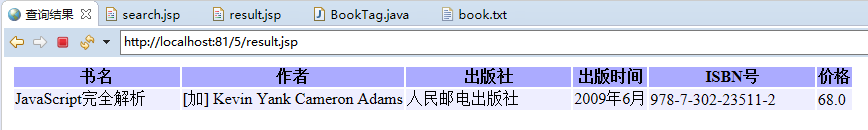
输入作者查询：





输入ISBN号查询：





**4.4问题及解决方案**

问题：检索图书信息时，输入中文会出现URL传中文参数乱码问题。

原因：传参时得到的是UTF-8形式的URL，服务器端通过request.getParameter()解码查询参数(通常是iso-8859-1)就会得到乱码。

解决方案：request.setCharacterEncoding("UTF-8");//加上这句

String search=request.getParameter("search");//此时search解码用的是UTF-8形式的URL

**4.5实验收获**

用JSP标签和MVC模式设计让web功能更加丰富强大，使工程视图看起来条理清晰，易于分类查找。

<c:forEach>标签对List对象迭代使大量图书信息的输出变得方便简单。

**4.6源代码**

**(1)BookTag.java**

**public** **class** BookTag **extends** SimpleTagSupport{

**private** String search = **null**;

**public** **void** setSearch(String search){

**this**.search = search;

}

**public** **void** doTag() **throws** JspException, IOException{

String output="";Book book;

//读取图书信息

BufferedReader bw=**new** BufferedReader(**new** FileReader("D:/Web应用/5/WebContent/WEB-INF/book.txt"));//要填绝对路径

String s="";

**int** flag=0;

**while**((s=bw.readLine())!=**null**) {

**if**(s.indexOf(search,0)>=0) {

flag=1;

**break**;

}

}bw.close();

**if**(flag==0) {

output="检索失败！";

}**else** {

String[] sArr=s.split("，");

**double** price=Double.*parseDouble*(sArr[5]);

book=**new** Book(sArr[0],sArr[1],sArr[2],sArr[3],sArr[4],price);

output="<table><tr style=\"background:#ababff\">"+

"<th width=\"20%\">书名</th><th>作者</th><th width=\"20%\">出版社</th>"+

"<th>出版时间</th><th width=\"20%\">ISBN号</th><th>价格</th></tr>"+

"<tr style=\"background:#eeeeff\">" +

"<td>"+book.getTitle()+"</td>" +

"<td>"+book.getAuthor()+"</td>" +

"<td>"+book.getPublisher()+"</td>" +

"<td>"+book.getDate()+"</td>" +

"<td>"+book.getIsbn()+"</td>" +

"<td>"+book.getPrice()+"</td></tr></table>";

}

getJspContext().setAttribute("info",output);

getJspBody().invoke(**null**);

}

}

**(2)BooksServlet.java**

@WebServlet("/BooksServlet")

**public** **class** BooksServlet **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

@Override

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

List<Book> books = **new** ArrayList<Book>();

BufferedReader bw=**new** BufferedReader(**new** FileReader("D:/Web应用/5/WebContent/WEB-INF/book.txt"));

String s1="";

StringBuffer s=**new** StringBuffer();

**while**((s1=bw.readLine())!=**null**) {

s.append(s1+"，");

}bw.close();

String str=**new** String(s);

String[] sArr=str.split("，");

**for**(**int** i=0;i<sArr.length;i+=6) {

**double** price=Double.*parseDouble*(sArr[i+5]);

books.add(**new** Book(sArr[i],sArr[i+1],sArr[i+2],sArr[i+3],sArr[i+4],price));

}

request.setAttribute("books",books); RequestDispatcher rd = request.getRequestDispatcher("/books.jsp");

rd.forward(request, response);

}

}