JSP其他主题

本章内容：共2小节，10个知识点

- 第1节：内置对象
- 第2节：指令与动作

本章目标

- 掌握JSP中内置对象的用法；
- 熟练使用page、include指令；
- 熟练使用include动作、JavaBean有关动作；

第1节【内置对象】

- 知识点1：request与response内置对象
- 知识点2： out、page内置对象
- 知识点3： pageContext、session、application内置对象
- 知识点4： exception内置对象
- 知识点5： config内置对象

知识点1【request与response内置对象】

- 内置对象指的是服务器已经创建好的对象，可以直接使用；
- JSP翻译生成的Java文件中，提供服务的方法是_jspService，JSP中的内容都将被翻译到该方法中，该方法的参数如下所示：

```
public void _jspService(final javax.servlet.http.HttpServletRequest request, final
javax.servlet.http.HttpServletResponse response)  throws java.io.IOException, javax.servlet.ServletException {
```

- 其中request,response就是两个内置对象，分别是HttpServletRequest和HttpServletResponse类型，可以在JSP中直接使用这两个接口中的方法；
- 例如：

```
<%
    String[] addrs=request.getAttribute(“city”);
    response.addCookie(new  Cookie(“code,”1”));
%>
```

知识点2【out、page内置对象】

- 在JSP的_jspService方法中，有out，page对象，如下所示：

```
javax.servlet.jsp.JspWriter out = null;
final java.lang.Object page = this;
out = pageContext.getOut();
```

- 可见，out的类型是JspWriter，page即当前类对象；
- out可以用来输出内容到客户端，但是程序员一般不会使用，因为直接使用<%=%>即可以实现输出；
- page也很少使用，与this相同；

知识点3【pageContext、session、application内置对象】-1

- 仔细看_jspService方法可以发现，有一个超级无敌的对象pageContext，其他多数内置对象都是通过它获得的：

```
final javax.servlet.jsp.PageContext pageContext;
javax.servlet.http.HttpSession session = null;
final javax.servlet.ServletContext application;
final javax.servlet.ServletConfig config;
javax.servlet.jsp.JspWriter out = null;
final java.lang.Object page = this;
javax.servlet.jsp.JspWriter _jspx_out = null;
javax.servlet.jsp.PageContext _jspx_page_context = null;
try {
  response.setContentType("text/html;charset=utf-8");
  pageContext = _jspxFactory.getPageContext(this, request, response,null, true, 8192, true);
  _jspx_page_context = pageContext;
  application = pageContext.getServletContext();
  config = pageContext.getServletConfig();
  session = pageContext.getSession();
  out = pageContext.getOut();
  _jspx_out = out;
```

知识点3【pageContext、session、application内置对象】-2

- pageContext对象是JSP中一个非常重要的对象，是javax.servlet.jsp.PageContext类型的对象，指的是页面的上下文，封装了其他的内置对象，同时代表的是四大作用域【页面、请求、会话、上下文】中的页面作用域，也可以在页面上下文范围添加属性，PageContext中与属性相关方法如下：

方法声明	方法描述
void setAttribute(java.lang.String name, java.lang.Object o)	将任意类型对象设置为属性，指定一个名字；
java.lang.Object getAttribute(java.lang.String name)	通过属性的名字，获取属性的值；
void removeAttribute(java.lang.String name)	通过属性的名字，删除属性；

- 实际应用中，如果自定义标签，pageContext对象会使用较多；除此之外，使用较少；

知识点3【pageContext、session、application内置对象】-3

- session是JSP中的另一个内置对象，是HttpSession类型的对象，可以在JSP中调用HttpSession接口中的任何方法；【默认情况下，session内置对象存在；可以通过指令设置不存在，后续学习】
- 在_jspService方法中的代码如下：

```
javax.servlet.http.HttpSession session = null;
session = pageContext.getSession();
```

- application是JSP中的另一个内置对象，是ServletContext类型的对象，可以在JSP中调用ServlletContext接口中的任何方法；
- 在_jspService方法中的代码如下：

```
final javax.servlet.ServletContext application;
application = pageContext.getServletContext();
```

- JSP中使用方法如下所示：

```
<%=session.getAttribute(“username”)%>
<%=application.getInitParameter(“appName”)%>
```

知识点4【exception内置对象】

- 内置对象exception比较特殊，默认情况下不存在；只有当JSP中使用指令指定该页面作为错误页面使用时才会翻译生成该内置对象。
- test.jsp中有如下声明【指令相关内容后续学习】：

```
<%@page isErrorPage="true" %>
```

- 则test.jsp翻译生成的.java文件中的_jspService方法中，就生成了内置对象exception，是Throwable类型的对象

```
java.lang.Throwable exception = org.apache.jasper.runtime.JspRuntimeLibrary.getThrowable(request);
  if (exception != null) {
    response.setStatus(javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
  }
```

知识点5【config内置对象】

- config是JSP中另一个内置对象，是ServletConfig类型的对象，在_jspService方法中，该对象的声明赋值情况如下：

```
final javax.servlet.ServletConfig config;
config = pageContext.getServletConfig();
```

- 在JSP中可以直接使用config对象调用ServletConfig接口中任意方法，例如，可以在web.xml中对JSP配置初始化参数，与前面学习的Servlet初始化参数相同的含义：

```
<servlet>
  <servlet-name>Test</servlet-name>
  <jsp-file>/test.jsp</jsp-file>
  <init-param>
  <param-name>title</param-name>
  <param-value>Test JSP</param-value>
  </init-param>
</servlet>

<servlet-mapping>
   <servlet-name>Test</servlet-name>
   <url-pattern>/test.do</url-pattern>
</servlet-mapping>
```

JSP的本质就是Servlet，所以可以像配置Servlet一样配置JSP

使用test.do访问test.jsp时，就可以使用到初始化参数。

```
<%=config.getInitParameter("title")%>
```

本节总结提问【内置对象】

- JSP中有哪九个内置对象?
- 哪几个内置对象比较“特殊”?

本节总结【内置对象】

- 共有九个内置对象，分别是page,request,session,application,out,response,pageContext,config,exception；
- 其中pageContext对象封装了所有其他的内置对象；
- session默认存在，可以通过指令配置为不存在，后续学习；
- exception默认不存在，当页面被指定为错误页面时才会存在；

第2节【指令与动作】

- 知识点1： JSP page指令标签的作用与特性
- 知识点2： JSP include、taglib指令标签的使用
- 知识点3： JSP include动作标签的使用
- 知识点4： 动态include与静态include的差异
- 知识点5： JSP其他动作标签简述

知识点1【JSP page指令标签的作用与特性】-1

- JSP可以通过指令元素而影响容器翻译生成Java类的整体结构；
- 指令的语法为：<%@ directive {attr="value"}* %>；
- 其中，directive为指令名，attr指该指令对应的属性名，一个指令可能有多个属性；
- JSP中常用的指令有三个：page、include、taglib，本章主要学习其中的两个：page指令、include指令。taglib指令将在JSP标签章节学习；
- page指令是最为复杂的一个指令，共有13个属性；
- page指令作用于整个JSP页面，可以将指令放在JSP页面任何一个位置；

主要学习每个属性的意义以及起到的作用。

知识点1【JSP page指令标签的作用与特性】-2

- page指令的import属性：用来引入JSP文件需要使用的类；
- 如下代码所示：

```
<%@page import="java.util.*,java.io.*" %>
<%@page import="com.etc.vo.*" %>
```

- 上述代码可以在JSP文件中使用，引入JSP需要使用的Java类；
- 可以使用逗号同时引入多个包，也可以在一个JSP文件中多次使用import；
- **值得注意的是**，import是page指令中唯一一个可以在一个JSP文件中多次出现的属性，其他属性在一个JSP文件中只能出现一次；

知识点1【JSP page指令标签的作用与特性】-3

- page指令的pageEncoding属性：用来设置JSP文件的页面编码格式；
- 如下代码所示：

  <%@page pageEncoding=“utf-8"%>
- 上述代码设置当前JSP的页面编码格式是utf-8。
- page指令的session属性：用来设置JSP页面是否生成session对象。该属性默认值为true，可以设置成false。
- 如下代码所示：

  <%@page session="false"%>
- session属性值设置为false后，该JSP翻译生成的类中将没有内置对象session，该JSP不参与会话。

知识点1【JSP page指令标签的作用与特性】-4

- page 指令的errorPage属性：设置JSP页面的错误页面。当JSP中发生异常或错误却没有被处理时，容器将请求转发到错误页面；
- 如下代码所示：

```
<%@page errorPage="error.jsp"%>
This is my JSP page. <br>
<%=100/0%><br>
```

- 显然，访问该页面将发生数学异常，而且并没有对异常进行处理，那么将跳转到错误页面error.jsp

知识点1【JSP page指令标签的作用与特性】-5

- isErrorPage属性默认值是false，可以设置为true。在JSP的错误页面中，将isErrorPage设置为true，则该页面翻译生成的Java类中，将生成exception内置对象。在error.jsp中加入代码：

  <%@page isErrorPage="true"%>

- 上述代码将error.jsp页面设置为错误页面，所以，在error.jsp翻译生成的Java类中的_jspService方法中将生成exception内置对象

- 注意：即使一个页面没有设置isErrorPage=“true”，也可以作为错误页面使用，区别在是否有exception内置对象产生。

知识点2【JSP include指令、taglib指令标签的使用】-1

- include指令是JSP中另外一个常用指令，用来**静态包含**其他页面；
- 所谓静态包含，指的是在**翻译期间**，把包含的页面也翻译到当前页面的Java文件中，也就是Java源文件即实现“二合一”；
- 例如，在main.jsp中编写如下代码：

  <%@include file="copyright.jsp"%>

- 过程：翻译main.jsp时，会把copyright.jsp文件翻译后插入到main.jsp翻译生成的.java文件中的相应位置；

知识点2【JSP include指令、taglib指令标签的使用】-2

- JSP API允许用户自定义标签，一个自定义标签库就是自定义标签的集合。
- Taglib指令引入一个自定义标签集合的定义，包括库路径、自定义标签。
- Taglib指令的语法：
- <%@ taglib uri="uri" prefix="prefixOfTag" %>
- uri属性确定标签库的位置，prefix属性指定标签库的前缀。
- 等价的XML语法：
- <jsp:directive.taglib uri="uri" prefix="prefixOfTag" />

知识点3【JSP include动作标签的使用】-1

- JSP规范中定义了一系列的标准动作。Web容器按照规范进行了实现，可以解析并执行标准动作；
- 标准动作使用标准的XML语法。

<jsp:action_name attribute1="value1" attribute2="value2">

</jsp:action_name>

- 其中action_name表示标准动作的名字，attribute1和attribute2是标准动作的若干个属性；

知识点3【JSP include动作标签的使用】-2

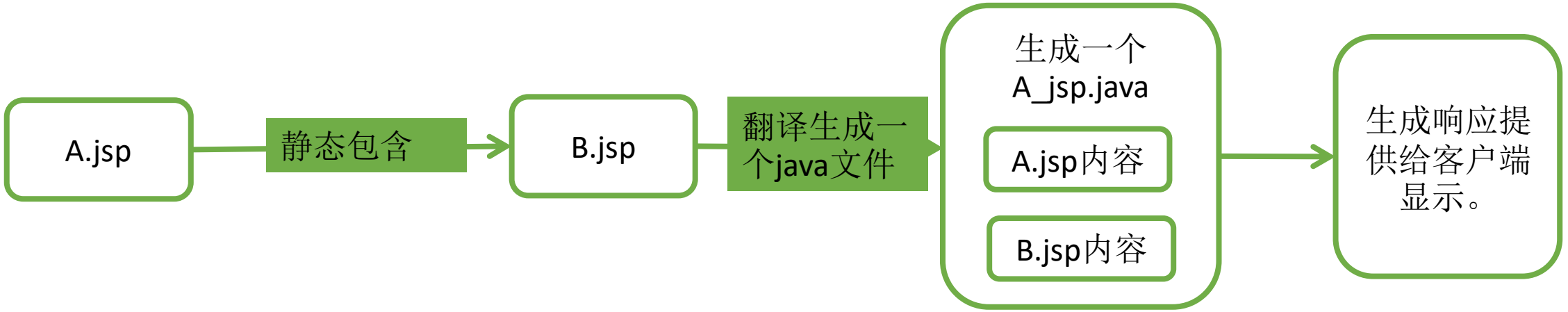
- include动作：在JSP页面中，进行动态包含，如下代码所示：

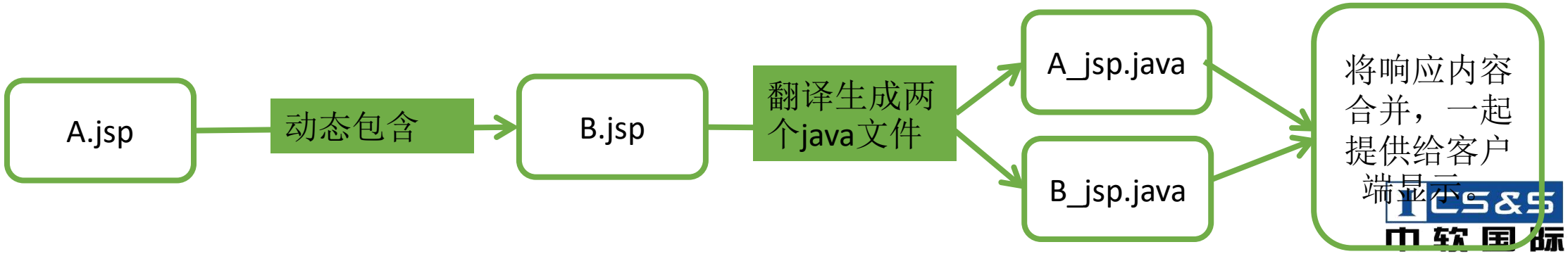
```
<jsp:include page=“top.jsp">
</jsp:include>
```

- <jsp:include>是动态包含，即在**运行期**访问被包含的页面，并将响应结果同包含页面的响应结果合并，生成最终响应。类似在Servlet中调用RequestDispatcher的include方法进行包含。

知识点4【动态include与静态include的差异】

- include标准动作与include指令都是实现包含其他页面的功能;
- include标准动作的属性是page，实现动态包含，发生在请求阶段;
- include指令的属性是file，实现静态包含，发生在翻译阶段。





知识点5【JSP其他动作标签简述】-1

- 除了include动作标签外，还有其他的动作标签，使用较少，进行简单了解；
- forward动作：在JSP页面中进行请求转发，如下代码所示：

```
<jsp:forward page=“loginsuccess.jsp">
</jsp:forward>
```

- 上述代码将把请求转发到loginsuccess.jsp页面，类似在Servlet中调用RequestDispatcher的forward方法进行请求转发。

知识点5【JSP其他动作标签简述】-2

- param动作：往往作为子动作使用，为forward和include动作传递参数，如下代码所示：

```
<jsp:forward page="copyright.jsp">
<jsp:param name="author" value="etc"/>
 </jsp:forward>
<jsp:include page="copyright.jsp">
  <jsp:param name="author" value="etc"/>
</jsp:include>
```

- 上述代码使用param为forward和include动作传递参数，参数将被作为请求参数传递。
- 使用标准动作时，一定注意正确结束标准动作，如<jsp:include>是标准动作的开始，一定要对应结束标记，如</jsp:include>。如果标准动作没有动作体，就可以直接结束，如<jsp:forward page="copyright.jsp">
- <jsp:include page="copyright.jsp"/>。

知识点5【JSP其他动作标签简述】-3

- JSP中还提供了3个与JavaBean有关的动作；
- JavaBean是用Java语言描述的软件组件模型，实际上是一个Java SE的类，这些类遵循一定的编码规范：
  - 必须是public类；
  - 必须有一个无参的public的构造方法；
  - 返回属性的方法为getXxxx()格式；
  - 设置属性的方法为setXxx(形式参数)格式；

知识点5【JSP其他动作标签简述】-4

- useBean动作：<jsp:useBean id="" class="" scope="">
- useBean标准动作用来使用JavaBean对象，JavaBean对象是某一范围（用scope指定）的属性；
- Java Bean对象名字用id指定，类型用class指定。如果对应范围没有该属性，则调用class指定类的无参构造方法，创建一个该类的对象，并将该对象存储为scope内的一个属性，属性名为id；
- 其中scope有四种：page、request、session、application，分别为PageContext范围、HttpServletRequest范围、HttpSession范围、ServletContext范围。如果不指定scope的值，默认为page范围。

知识点5【JSP其他动作标签简述】-5

- setProperty 动作：<jsp:setProperty name="" property="" param|value=""/>
- setProperty标准动作可以用来对JavaBean对象的属性赋值，替代调用setXxxx方法；
- setProperty的name属性表示JavaBean对象的id值，property表示需要调用的setXxx方法中的Xxx部分，将首字母变小写。比如需要调用setCustname方法，则property即为Custname首字母变小写，即custname；
- 如果setXxx方法的参数是某一个请求参数的值，则使用param属性指定请求参数名字即可；
- 如果setXxx方法的参数是一个常量，则使用value属性指定即可。
- 同时，setProperty标准动作可以对一些常见数据类型直接转换，如字符串与Integer的转换就可以自动进行；

知识点5【JSP其他动作标签简述】-6

- getProperty动作：<jsp:getProperty name="" property=""/>
- getProperty标准动作用来调用JavaBean对象的getXxx方法，将其返回值在当前位置输出。
- name是JavaBean对象的id值，property的值是getXxx方法中的Xxx部分，首字母变小写。假设需要调用getAddress方法显示其返回值，那么property的值就是Address的首字母变小写，即address。

本节总结提问【指令与动作】

- page指令有哪些常用属性?
- include指令和include动作有什么区别?

本节总结【指令与动作】

- page指令中常用的属性有import、session等；
- include指令是静态包含，include动作是动态包含；

本章总结

- 本章主要学习了JSP中的九个内置对象，每个内置对象的具体类型；
- 学习了常用的page指令、include指令、taglib指令；
- 学习了常用的include动作；
- 掌握静态包含和动态包含的区别；

本章作业

- 作业1:
- 题目：自行编写JSP，验证9个内置对象的作用，验证静态包含和动态包含的区别。
- 难度：低