JSP Custom Tags Quiz

1. **Rewrite the JSP page in the first demo (forEach) to use scripting instead of the JSTL forEach.**

|  |
| --- |
| <**table**>  <**c:forEach var="student" items="${**students**}"**>  <**tr**>  <**td**>**${**student.name**}**</**td**>  <**td**>**${**student.age**}**</**td**>  </**tr**>  </**c:forEach**> </**table**> |

🡪

|  |
| --- |
| <**table**>  **<%** Student[] students = (Student[]) request.getAttribute(**"students"**);  **for** (**int** i = 0; i < students.**length**; i++) {  **%>** <**tr**>  <**td**>**<%=**students[i].getName()**%>** </**td**>  <**td**>**<%=**students[i].getAge()**%>** </**td**>  </**tr**>  **<%** }  **%>** </**table**> |

1. **How do custom tags relate to JSTL?**

* JSTL is a standard library of custom tags.
* Provides new tags for JSPs that reduce scripting
* The JSTL library provides 5 kinds of tags, each having a different (standard) prefix. We will “import” a library by placing a “taglib” directive at the top of the jsp page

1. **What is the role of the URI in the TLD and the taglib directive?**

* It relates to the tablib directive.
* For example, in the TLD we have the URL as below

<**uri**>/WEB-INF/tlds/TldDemo</**uri**>

* At the top of the jsp page, we will import this lib

<%@ **taglib prefix**='**mytag**' **uri**='**/WEB-INF/tlds/TldDemo**' %>

1. **What is a tag handler class?**

* This class extends the tag support framework class:

extends SimpleTagSupport

* This class handle calls to the custom tags

1. **What is the role of attribute setters in a tag handler class?**

* The container calls these methods to set values from the tag attributes.
* It uses JavaBean property naming convention

1. **What is the role of the doTag() method in a tag handler class?**

* This is called from JSP servlet to render custom tag
* The container calls doTag() method when the JSP invokes the tag using the name declared in the TLD
* So the doTag() method does the actual work: format and print out data based on the attributes of the tag.

1. **What does the operation getJspContext().getOut().write(“Hi Bob”) do when called in a doTag() method?**

**public void** doTag() **throws** JspException, IOException {  
 JspWriter out = getJspContext().getOut();  
  
 getJspContext().getOut().write(**"Hi Bob"**);  
  
}

It writes a string “Hi Bob” into the web browser like:

<html>

<body>

**Hi Bob**

</body>

</html>

1. **What does the operation getJspBody().invoke(null) do when called in a doTag() method?**

* Example:

**public void** doTag() **throws** JspException, IOException {  
   
 getJspBody().invoke(**null**);  
  
 }

* Result:

java.lang.NullPointerException

**Description** The server encountered an unexpected condition that prevented it from fulfilling the request.

**Exception**

org.apache.jasper.JasperException: An exception occurred processing JSP page [/index.jsp] at line [19]

16: <title>JSTL and Custom Tags Lecture</title>

17: </head>

18: <body>

19: <mytag:Head theColor='red' words='hello'/>

20: </body>

21: </html>

Stacktrace:

org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:617)

org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:514)

org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:386)

org.apache.jasper.servlet.JspServlet.service(JspServlet.java:330)

javax.servlet.http.HttpServlet.service(HttpServlet.java:742)

org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)

**Root Cause**

java.lang.NullPointerException

mum.cs472.Head.doTag(Head.java:17)

org.apache.jsp.index\_jsp.\_jspx\_meth\_mytag\_005fHead\_005f0(index\_jsp.java:162)

org.apache.jsp.index\_jsp.\_jspService(index\_jsp.java:125)

org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)

javax.servlet.http.HttpServlet.service(HttpServlet.java:742)

org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:476)

org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:386)

org.apache.jasper.servlet.JspServlet.service(JspServlet.java:330)

javax.servlet.http.HttpServlet.service(HttpServlet.java:742)

org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)

**Note** The full stack trace of the root cause is available in the server logs.