JSP Custom Tags Quiz

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

   Answers:

```
<% Student students="${students}"
for(int i = 0; i < students.length; i++) { %>
<%= students[i].name  + ”,  “ + students[i].age = %>
<br>
<% }  %>
```

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

Answers: *JSTL is standard library of JSP actions which provides new tags for JSPs to reduce scripting. Beyond JSTL functionality, Custom Tag allows developers to create their own actions. Custom tag provides an easy mechanism to dynamically generate markup for common processing tasks. There are four key steps: a) defining a tag including attributes and body; b) write a Tag Library Descriptor (TLD) that the container will read; c) write a tag handler class that implements the tag functionality; d) use the tag on a JSP page and link it to the tag descriptor*

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

Answers: *The taglib directive declares that your JSP page uses a set of custom tags, identifies the location of the library. The URL in TDL(XML file) used to defines a public URI that uniquely identifies this version of the tag library. Similarity URL in taglib directive provides a means for identifying the custom tags in our JSP page. The URL attribute value resolves to a location the container understands.*

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

Answers: *Custom tags are distributed in a tag library, which defines a set of related custom tags and contains objects that implement the tags. Thus, the object that implements a custom tag is called a tag handler. Thus , when the JSP engine encounters a tag in a JSP page, a new tag handler is initialized.*

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

Answers: *For each tag attribute, we must define a set method in the tag handler that conforms to the JavaBeans architecture conventions. This tag handler contains the methods for setting value(initialize) each attributes of a custom tag.*

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

Answers: *doTag is method invoked from JSP servlet to render custom tag. The setPageContext() and setParent() methods of the Tag interface are invoked to set up the environment context for the tag handler The doTag is defined within our tag handler class to initialize the tag handler. Finally, we can determine whether the tag body should be evaluated by returning one of the value constants from tag handler class.*

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

Answers: *The handler for a basic tag without a body must implement the doTag method of the SimpleTag interface. The doTag method is invoked when the end element of the tag is encountered. In this regard, write("Hi Bob") is an implementation of doTag ,here, which prints :Hi Bob" inside the placeholder of the custom tag within JSP file.*

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

Answers: *If a tag handler needs to evaluate the body, it gets the body using the getJspBody method of SimpleTag and then evaluates the body using the* ***invoke*** *method. The body of the tag is encapsulated in a JSP fragment. If the test is true, the handler retrieves the fragment using the getJspBody method. The invoke method directs all output to a supplied writer or, if the writer is* ***null****, to the JspWriter returned by the getOut method of the JspContext associated with the tag handler.*