Creating Custom Tags in JSP

Nima Memarzadeh

Module 10 Assignment - Creating Custom Tags – Paper

07/17/2025

JSP is a highly extensible platform for building server side web applications. One of JSP's most important features is support for custom tags. Custom tags are user-defined action tags that encapsulate processing code to make jsp pages easier to read, write and reuse. Custom tags allow for a clean separation between view and logic to follow good architectural patterns such as MVC(Model-View-Controller).

Custom tags are the alternative to using scriptlets to embed java code in a JSP file. Scriptlets are a maintenance nightmare. Custom tags allow for moving reusable processing logic to java classes. JSP pages are kept readable (for front-end developers) and maintainable (by back-end developers).

Advantages of Custom Tags

Custom tags provide many architectural and productivity benefits:

1. **Separation of Concerns:** Custom tags encapsulate Java logic in tag handler classes instead of embedding it in the JSP. This clearly separates the presentation layer (UI design) from the business logic. It leads to more maintainable code and supports large-scale development by multiple teams (GeeksforGeeks, 2025).
2. **Reusability:** Custom tags are self-contained and can be reused across different JSP pages and even different applications. It allows developers to follow the DRY (Don’t Repeat Yourself) principle and eliminates code redundancy (Flylib, 2002).
3. **Readability and Maintainability:** Custom tags make JSPs more readable and maintainable by moving the Java logic out of the view layer. The code of the JSP remains simple and is understandable even for non-backend developers. The tag <user: name /> is much more readable than <%= user.getName() %>, for instance (SitePoint).

4. **Attribute-based Flexibility:** Custom tags support attributes to pass data. Based on these attributes, a single custom tag can have dynamic behavior. Custom tags can change their behavior based on how and where they are used. It makes them more flexible and modular.
5. **Tag Body Manipulation:** Some advanced custom tags can even access or manipulate their body content. It makes custom tags great for things like loops, formatting, conditional display, and so on (Flylib, 2002).

Disadvantages of Custom Tags

Custom tags are not always the ideal solution. The following are some caveats in using custom tags:

1. **Development Effort:** Creating a custom tag involves writing a handler class, a TLD (Tag Library Descriptor) file, and registering the tag with directives in a JSP. This may be considered an overhead for small projects.
2. **Learning Curve:** Developers must be familiar with the JSP tag APIs, the TLD XML syntax, and how the JSP engine processes and invokes tags (including the tag lifecycle). This learning curve can be steep for beginners (BeginnersBook, 2022).
3. **Expression Language Limitations:** Although attributes of a custom tag can accept runtime expressions (rtexprvalue), working with complex Java logic inside a custom tag can be cumbersome and verbose compared to using plain Java code (Flylib, 2002).

Requirements for Developing a Custom Tag

The following are the requirements to create a custom tag:

1. **Tag Handler Class:** This is the Java class that defines what the tag does. It usually extends SimpleTagSupport for convenience and overrides the doTag() method.

```
public class WelcomeTag extends SimpleTagSupport {
    public void doTag() throws JspException, IOException {
        getJspContext().getOut().print("Welcome to Custom Tags!");
    }
}
```

2. **Tag Library Descriptor (TLD) File:** This XML file associates your tag name to the Java class. It also specifies the attributes and other details.

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>2.0</jsp-version>
  <tag>
    <name>welcome</name>
    <tag-class>mytags.WelcomeTag</tag-class>
    <body-content>empty</body-content>
  </tag>
</taglib>
```

3. **Taglib Directive in JSP**: Use this to load the tag library in your JSP file and assign a prefix:

```
<%@ taglib uri="WEB-INF/welcome.tld" prefix="custom" %>
<custom:welcome />
```

4. **Optional: Attributes and Body Support**: Tags can accept attributes using setAttribute() methods and process tag content using JspFragment.

```
private String message;
public void setMessage(String msg) { this.message = msg; }
public void doTag() throws JspException, IOException {
    getJspContext().getOut().write(message);
}
```

5. **Lifecycle Methods**: Depending on the type of tag, you may use methods like doStartTag(), doEndTag(), or just doTag().

Personal Opinion

In my opinion, custom tags are the most beautiful solution to the problem of maintaining large JSP-based applications, although the initial learning curve is not trivial. Once learned, however, it's hard to go back to a non-modular approach.

On the other hand, custom tags are not the right tool for every job. In particular, if you're just prototyping something and you're not going to have a significant amount of JSP code to begin with, JSTL or EL will usually get you far enough. But in a real professional Java EE development situation where long-term maintenance is a concern, custom tags are indispensable. I particularly like the SimpleTagSupport approach to writing custom tags introduced in JSP 2.0. It provides a powerful yet easy-to-use implementation technique. Personally, if I were writing web applications that I wanted to be beautiful and scalable, I would always use custom tags rather than scriptlets.

Conclusion

Custom tags in JSP allow developers to build modular, reusable, and maintainable components that abstract the underlying logic from the view, improving the overall architecture and readability of web applications. While their setup is more complex compared to traditional JSP approaches, the benefits custom tags provide in terms of separation of concerns and clean JSPs are significant. Backed by handler classes, TLD files, and flexible attribute mechanisms, they can encapsulate advanced processing. Mastering custom tags is crucial for any developer creating scalable Java web applications.

References

BeginnersBook. (2022). *JSP Custom Tags with Example – JSP Tutorial*. Retrieved from https://beginnersbook.com/2014/01/jsp-custom-tags-with-example-jsp-tutorial/

Flylib. (2002). *Writing a Custom JSP Tag Library*. Retrieved from https://flylib.com/books/en/2.96.1.87/1/

GeeksforGeeks. (2025). *Custom Tags in JSP*. Retrieved from https://www.geeksforgeeks.org/java/custom-tags-in-jsp/

Grant, A. (2004, October 27). *JSP 2.0: Simple Tags Explained*. SitePoint. https://www.sitepoint.com/jsp-2-simple-tags/