In **JSP 2.0**, **tag files** are a simple and powerful way to create **custom tags** without needing to write Java classes or use the more complex JSP tag handler API. A **tag file** is essentially a JSP file with a .tag (or .tagf) extension and can be reused across JSPs to encapsulate reusable view logic.

**✅ Key Concepts of Tag Files**

1. **Location**:
   * /WEB-INF/tags/ (or its subdirectories)
   * Not directly accessible by the browser
2. **File Extension**:
   * .tag – for regular tags
   * .tagx – for XML-compliant tag files (for JSP documents)

**📁 Folder Structure**

project/

├── WebContent/

│ ├── index.jsp

│ └── WEB-INF/

│ └── tags/

│ └── hello.tag

**🧪 Example 1: Simple Custom Tag with No Attributes**

**hello.tag**

<%@ tag description="Simple Hello Tag" pageEncoding="UTF-8"%>

Hello from custom tag!

**index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<h2>Using Custom Tag</h2>

<my:hello />

</body>

</html>

**🧪 Example 2: Custom Tag with Attributes**

**welcome.tag**

<%@ tag description="Welcomes a user" pageEncoding="UTF-8"%>

<%@ attribute name="name" required="true" %>

Hello, <strong>${name}</strong>! Welcome to JSP custom tags.

**index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:welcome name="Manjiri"/>

</body>

</html>

**🧪 Example 3: Custom Tag with Body Content**

**box.tag**

<%@ tag description="Box container" pageEncoding="UTF-8"%>

<%@ tag body-content="scriptless" %>

<div style="border: 1px solid black; padding: 10px; width: 300px;">

<jsp:doBody />

</div>

**index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:box>

This content is inside the custom box tag.

</my:box>

</body>

</html>

**🔧 Features Supported in Tag Files**

| **Feature** | **Supported** |
| --- | --- |
| Attributes | ✅ |
| Body content | ✅ |
| Nested tags | ✅ |
| JSP scripting (limited) | ✅ |
| Reusability | ✅ |

**📝 Best Practices**

* Keep tag files in /WEB-INF/tags to avoid direct access.
* Use meaningful attribute names.
* Prefer tag files over Java-based custom tags for UI logic.

Here are **advanced examples** of JSP 2.0 tag files using **iteration**, **conditions**, and **nested tags**, all using .tag files.

**✅ Example 1: Iteration with a Custom Tag**

**📄 repeat.tag**

<%@ tag description="Repeats content N times" pageEncoding="UTF-8"%>

<%@ attribute name="count" required="true" type="int" %>

<%@ tag body-content="scriptless" %>

<%

for (int i = 0; i < count; i++) {

%>

<jsp:doBody />

<%

}

%>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:repeat count="3">

<p>This is repeated.</p>

</my:repeat>

</body>

</html>

**✅ Example 2: Conditional Display (like if)**

**📄 if.tag**

<%@ tag description="Simple IF condition" pageEncoding="UTF-8"%>

<%@ attribute name="test" required="true" type="boolean" %>

<%@ tag body-content="scriptless" %>

<%

if (test) {

%>

<jsp:doBody />

<%

}

%>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:if test="<%= 10 > 5 %>">

<p>Condition is true, so this is displayed.</p>

</my:if>

</body>

</html>

**✅ Example 3: Nested Tags (Parent-Child Relationship)**

**📄 parent.tag**

<%@ tag description="Parent tag" pageEncoding="UTF-8"%>

<%@ tag body-content="scriptless" %>

<div style="border: 2px solid blue; padding: 10px;">

<h3>Parent Tag:</h3>

<jsp:doBody />

</div>

**📄 child.tag**

<%@ tag description="Child tag" pageEncoding="UTF-8"%>

<p style="margin-left: 20px;">-- Child content here --</p>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:parent>

<my:child />

<my:child />

</my:parent>

</body>

</html>

**📌 Summary of Features Used**

| **Concept** | **Tag File** | **Highlight** |
| --- | --- | --- |
| Iteration | repeat.tag | Loops over count times |
| Condition | if.tag | Evaluates a boolean condition |
| Nesting | parent.tag | Used together with child.tag |

The error:

**The operator != is undefined for the argument type(s) int, null**

means you're comparing an int (a primitive type) with a null — which is not allowed in Java or in JSP EL or scriptlets. Primitives like int, double, etc., **cannot be null**, and **cannot be directly compared to null**.

**✅ Fix 1: Use Wrapper Class (Integer) Instead of int**

If you're accepting an attribute in a tag file, and it may be optional or null, change the type from int to java.lang.Integer.

**❌ Incorrect (primitive type):**

<%@ attribute name="count" required="false" type="int" %>

**✅ Correct (wrapper class):**

<%@ attribute name="count" required="false" type="java.lang.Integer" %>

Then inside the tag file:

<%

if (count != null && count > 0) {

for (int i = 0; i < count; i++) {

%>

<jsp:doBody />

<%

}

}

%>

**✅ Fix 2: Ensure count Is Always Set (Required)**

If you're **always** going to pass count, then just make it required and avoid null checking:

<%@ attribute name="count" required="true" type="int" %>

Then use it safely:

<%

for (int i = 0; i < count; i++) {

%>

<jsp:doBody />

<%

}

%>

Customizing tag behavior using **attributes** is one of the key powers of **JSP 2.0 tag files**. You can pass values into your custom tag files using the <%@ attribute %> directive, allowing you to create dynamic, flexible UI components.

**✅ Syntax for Declaring Attributes**

<%@ attribute name="attrName" required="true|false" type="java.lang.String|int|boolean|..." %>

**🧪 Example 1: Greeting with Attribute**

**📄 greet.tag**

<%@ tag description="Greets a user" pageEncoding="UTF-8"%>

<%@ attribute name="name" required="true" type="java.lang.String" %>

<p>Hello, <strong>${name}</strong>!</p>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:greet name="Manjiri" />

</body>

</html>

**🧪 Example 2: Styling with Multiple Attributes**

**📄 alertBox.tag**

<%@ tag description="Alert box" pageEncoding="UTF-8"%>

<%@ attribute name="message" required="true" type="java.lang.String" %>

<%@ attribute name="color" required="false" type="java.lang.String" %>

<div style="border:1px solid black; padding:10px; background-color:<%= color != null ? color : "lightyellow" %>">

<strong>Alert:</strong> ${message}

</div>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:alertBox message="Invalid login attempt!" color="lightpink" />

<my:alertBox message="Welcome back!" />

</body>

</html>

**🧪 Example 3: Dynamic Output Based on Boolean Attribute**

**📄 status.tag**

<%@ tag description="Shows status" pageEncoding="UTF-8"%>

<%@ attribute name="active" required="true" type="boolean" %>

<p>Status:

<span style="color:<%= active ? "green" : "red" %>">

<%= active ? "Active" : "Inactive" %>

</span>

</p>

**📄 index.jsp**

<%@ taglib prefix="my" tagdir="/WEB-INF/tags" %>

<html>

<body>

<my:status active="true" />

<my:status active="false" />

</body>

</html>

**✅ Using type Correctly in Attributes**

| **Type in attribute** | **Java Equivalent** | **Use Case** |
| --- | --- | --- |
| java.lang.String | Strings | Names, labels, messages |
| int / Integer | Numbers | Counts, numeric values |
| boolean / Boolean | Conditions | Flags |
| java.util.List | Collections | Looping, dropdowns, etc. |

**🛠 Best Practices**

* Always use required="true" if the tag won't work properly without the attribute.
* Use wrapper types (Integer, Boolean) for optional attributes that may be null.
* Combine attributes and body content (<jsp:doBody />) for maximum flexibility.