**CUSTOM TAGS IN JSP**

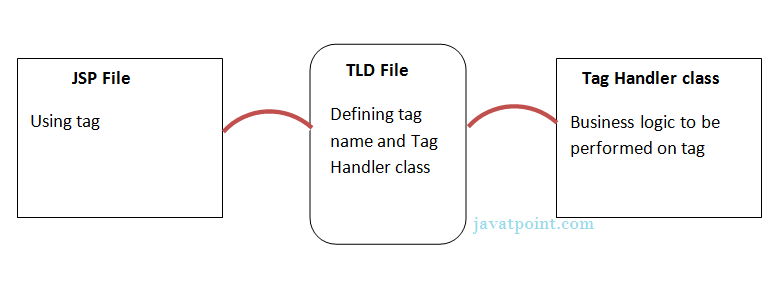
A custom tag is a user-defined JSP language element. When a JSP page containing a custom tag is translated into a servlet, the tag is converted to operations on an object called a tag handler. The Web container then invokes those operations when the JSP page's servlet is executed.

In this example, we are going to create a **custom tag that prints the current date and time**. We are performing action at the start of tag.

For creating any custom tag, we need to follow following steps:

1. **Create the Tag handler class** and perform action at the start or at the end of the tag.
2. **Create the Tag Library Descriptor (TLD) file** and define tags
3. **Create the JSP file that uses the Custom tag defined in the TLD file**

Understanding flow of custom tag in jsp



1) Create the Tag handler class

To create the Tag Handler, we are inheriting the **TagSupport class** and overriding its method **doStartTag()**.To write data for the jsp, we need to use the **JspWriter class**.

The **PageContext** class provides **getOut()** method that returns the instance of JspWriter class. TagSupport class provides instance of pageContext bydefault.



2) Create the TLD file

**Tag Library Descriptor** (TLD) file contains information of tag and Tag Hander classes. It must be contained inside the **WEB-INF** directory.

3) Create the JSP file

Let's use the tag in our jsp file. Here, we are specifying the path of tld file directly. But it is recommended to use the uri name instead of full path of tld file. We will learn about uri later.

It uses taglib directive to use the tags defined in the tld file.

Example:

A JSP tag extension lets you create new tags that you can insert directly into a JavaServer Page. The JSP 2.0 specification introduced the Simple Tag Handlers for writing these custom tags.

To write a custom tag, you can simply extend SimpleTagSupport class and override the doTag() method, where you can place your code to generate content for the tag.

Create "Hello" Tag

Consider you want to define a custom tag named <ex:Hello> and you want to use it in the following fashion without a body −

<ex:Hello />

To create a custom JSP tag, you must first create a Java class that acts as a tag handler. Let us now create the HelloTag class as follows −

packagecom.tutorialspoint;

importjavax.servlet.jsp.tagext.\*;

importjavax.servlet.jsp.\*;

import java.io.\*;

public class HelloTag extends SimpleTagSupport {

public void doTag() throws JspException, IOException {

JspWriter out = getJspContext().getOut();

out.println("Hello Custom Tag!");

}

}

The above code has simple coding where the doTag() method takes the current JspContext object using the getJspContext() method and uses it to send "Hello Custom Tag!" to the current JspWriterobject

Let us compile the above class and copy it in a directory available in the environment variable CLASSPATH. Finally, create the following tag library file: <Tomcat-Installation-Directory>webapps\ROOT\WEB-INF\custom.tld.

<taglib>

<tlib-version>1.0</tlib-version>

<jsp-version>2.0</jsp-version>

<short-name>Example TLD</short-name>

<tag>

<name>Hello</name>

<tag-class>com.tutorialspoint.HelloTag</tag-class>

<body-content>empty</body-content>

</tag>

</taglib>

Let us now use the above defined custom tag Hello in our JSP program as follows −

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

<html>

<head>

<title>A sample custom tag</title>

</head>

<body>

<ex:Hello/>

</body>

</html>

Call the above JSP and this should produce the following result −

Hello Custom Tag!

# Attributes in JSP Custom Tag

# There can be defined too many attributes for any custom tag. To define the attribute, you need to perform two tasks:

* Define the property in the TagHandler class with the attribute name and define the setter method
* define the attribute element inside the tag element in the TLD file

Let's understand the attribute by the tag given below:

1. **<m:cube** number="4"**></m:cube>**

Here **m** is the prefix, **cube** is the tag name and **number** is the attribute.

### Simple example of attribute in JSP Custom Tag

In this example, we are going to use the cube tag which return the cube of any given number. Here, we are defining the number attribute for the cube tag. We are using the three file here:

* index.jsp
* CubeNumber.java
* mytags.tld

**index.jsp**

**<**%@ taglib uri="WEB-INF/mytags.tld" prefix="m" %**>**

Cube of 4 is: **<m:cube** number="4"**></m:cube>**

**CubeNumber.java**

**package** com.javatpoint.taghandler;

**import** javax.servlet.jsp.JspException;

**import** javax.servlet.jsp.JspWriter;

**import** javax.servlet.jsp.tagext.TagSupport;

**public** **class** CubeNumber **extends** TagSupport{

**private** **int** number;

**public** **void** setNumber(**int** number) {

**this**.number = number;

}

**public** **int** doStartTag() **throws** JspException {

    JspWriter out=pageContext.getOut();

**try**{

        out.print(number\*number\*number);

    }**catch**(Exception e){e.printStackTrace();}

**return** SKIP\_BODY;

}

}

**mytags.tld**

**<?xml** version="1.0" encoding="ISO-8859-1" **?>**

<!DOCTYPE taglib

        PUBLIC "-//Sun Microsystems, Inc.//DTD JSP Tag Library 1.2//EN"

        "http://java.sun.com/j2ee/dtd/web-jsptaglibrary\_1\_2.dtd"**>**

**<taglib>**

**<tlib-version>**1.0**</tlib-version>**

**<jsp-version>**1.2**</jsp-version>**

**<short-name>**simple**</short-name>**

**<uri>**http://tomcat.apache.org/example-taglib**</uri>**

**<description>**A simple tab library for the examples**</description>**

**<tag>**

**<name>**cube**</name>**

**<tag-class>**com.javatpoint.taghandler.CubeNumber**</tag-class>**

**<attribute>**

**<name>**number**</name>**

**<required>**true**</required>**

**</attribute>**

**</tag>**

**</taglib>**

#### Output

1. Cube of 4 is: 64