Can not find the tag library descriptor for "http://java.sun.com/jsp/jstl/core"

Ask Question

asked 6 years, 5 months ago
viewed 119,580 times
active 15 days ago

FEATURED ON META
Union Meta Zoo #1: Why another network?
Announcing the arrival of Valued Associate #679: Cesar Manara
Data science time! April 2019 and today with experience
The Ask Question Wizard is Live!

I have included this at the very top of my JSP page:

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

I already placed the JSTL JAR file in the `WEB-INF/lib` directory. But still, the JSP can't resolve the taglib. I get the below error:

> Can not find the tag library descriptor for "http://java.sun.com/jsp/jstl/core"

I am using Eclipse Juno and the project structure is shown below:

- WebContent
  - META-INF
  - WEB-INF
    - lib
      - jstl.jar
      - jsp.jar
    - web.xml
  - index.jsp

share improve this question — edited Jul 23 '14 at 7:37 BenuC — asked Nov 8 '12 at 9:10 user462206 — 4,087 · 13 · 48 · 104 · 154

add the jar folder to your classpath – Abubakkar Nov 8 '12 at 9:12

Are you talking about eclipse not being able to resolve the taglib, or tomcat? – nikochiao Nov 8 '12 at 9:14

add a comment

13 Answers

active oldest votes

> Can not find the tag library descriptor for "http://java.sun.com/jsp/jstl/core"

Based on one of your previous questions you're using Tomcat 7. In that case you need JSTL 1.1 instead of JSTL 1.2 which is for Tomcat 8. Tomcat 7 does not ship with the required libraries like `jstl.jar`, `jsp.jar`. The file name `jstl.jar` is typical for JSTL 1.0 and 1.1. This version requires a `standard.jar` along in `/WEB-INF/lib`, which contains the necessary TLD files. However, in your particular case the `standard.jar` is clearly missing in `/WEB-INF/lib`, and that's exactly the reason why the taglib URI couldn't be resolved.

To solve this you must remove the wrong JAR file, download jstl-1.2.jar and drop it in its entirely in `/WEB-INF/lib`. That's all. You do not need to extract it nor to fiddle in properties *Build Path*.

Don't forget to remove that loose `c.tld` file too. It absolutely doesn't belong there. This is indeed trying to serve loose poor tutorials or answers which are based on JSTL 1.0 which caused by a major misunderstanding and misconfiguration. There is never a need to have a loose JSTL TLD file in the classpath, also not in previous JSTL versions.

In case you're using Maven, use the following dependency instead:

```
<dependency>
    <groupId>javax.servlet/groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
```

You should also make sure that your `web.xml` is declared conform at least Servlet 2.4 and thus not as Servlet 2.2 or older. Otherwise EL expressions inside JSTL tags would in turn fail to work. Pick the highest version matching your target container and make sure that you don't have a `c.tld` lurking somewhere in your `web.xml`. Here's a Servlet 3.0 (Tomcat 7) compatible example:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    version="3.0">

    <!-- Config here. -->

</web-app>
```

See also:

- Our JSTL wiki page (you can reach there by hovering the mouse on jstl and clicking *info* link)
- How to install JSTL? The absolute url: http://java.sun.com/jsp/jstl/core cannot be resolved

share improve this answer — edited May 23 '17 at 12:26 Community ♦ — answered Nov 9 '12 at 12:02 BalusC — 1 · 4 · 9 · 305 · 2,705 · 2,726

add a comment

The URI depends on the version of JSTL you are using. For Version 1.0 use:

```
http://java.sun.com/jstl/core
```

and for 1.1 (and later), you need to use:

```
http://java.sun.com/jsp/jstl/core
```

share improve this answer — answered Nov 8 '12 at 9:13 nikochiao — 15.2k · 11 · 59 · 90

Helped me. Thank you. – Abdul Jun 21 '16 at 7:23

add a comment

I had same problem and despite having jstl

```
<dependency>
    <groupId>javax.servlet/groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
```

I had to add 'standard' as well:

```
<dependency>
    <groupId>taglibs/groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
</dependency>
```

Also, as mentioned in previous post:

- for version 1.0 use: http://java.sun.com/jstl/core
- for 1.1 (and later) use: http://java.sun.com/jsp/jstl/core

share improve this answer — answered Apr 18 '17 at 19:05 Hamed Bulaienko — 4,162 · 17 · 43 · 64

add a comment

If you are using maven your pom's `<dependencies>` tag should look like:

```
<dependencies>
    <dependency>
        <groupId>javax.servlet/groupId>
        <artifactId>jstl</artifactId>
        <version>1.2</version>
    </dependency>
    <dependency>
        <groupId>javax.servlet.jsp.jstl/groupId>
        <artifactId>javax.servlet.jsp.jstl-api</artifactId>
        <version>1.2.1</version>
    </dependency>
    <dependency>
        <groupId>taglibs/groupId>
        <artifactId>standard</artifactId>
        <version>1.1.2</version>
    </dependency>
</dependencies>
```

It works for me.

share improve this answer — edited Dec 10 '16 at 14:55 user — 2,977 · 11 · 12 · 44 — answered Dec 10 '16 at 13:52 Anand Eugene — 1,1 · 1 · 2 · 4

I have been looking at this issue for years off and on, with all the correct setup which is documented everywhere. Adding maybe as eleven above from mvnrepository.com/artifact/org.apache.taglibs/ — resolved it. – Richard Bradley Smith Jan 18 at 18:32

add a comment

Downloading jstl-1.2.jar instead of jstl.jar solves the problem

share improve this answer — answered Jun 5 '18 at 5:09 vishna singh — 16 · 1 · 1 · 4

add a comment

you need to configure this in web.xml as well.Please refer below code.

```
<taglib>
    <taglib-uri>http://java.sun.com/jsp/jstl/core</taglib-uri>
    <taglib-location>WEB-INF/lib/c.tld</taglib-location>
</taglib>
```

Please let me know if you still face any issue.

share improve this answer — answered Nov 8 '12 at 9:50 Gaurav — 1,475 · 2 · 10 · 44

Whether you need to do this depends on the JSP version in use. This is only necessary for version smaller than 2.3. Tomcat since JSP 2.3 since 8.5. In normal circumstances, this should not be needed. – nikochiao Nov 8 '12 at 10:03

Thanks @nikochiao. I was not aware of this. – Gaurav Nov 8 '12 at 11:49

This bad myth is only true when you used the wrong JSTL version for the JSP version. You should instead rather have checked and provide the right JSTL version for the JSP version used. You should absolutely not extract the TLD file from the JAR and clutter the classpath and web.xml with it. See also stackoverflow.com/a/4928316 – BalusC Nov 8 '12 at 11:55

add a comment

Well, I have read this post and seems it not good enough to fix this issue.

Spring based application on the tomcat 8 will not work.

Here is the solution

Step 1 = Add the following dependency in your pom.xml

```
<!-- JSTL Support -->

    <dependency>
        <groupId>javax.servlet/groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.1.0</version>
    </dependency>
<dependency>
    <groupId>javax.servlet.jsp</groupId>
    <artifactId>javax.servlet.jsp-api</artifactId>
    <version>2.3.3</version>
</dependency>
<dependency>
    <groupId>javax.servlet/groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
```

Step 2 = Add the following two statement in your JSP page, ensure that you are using `isELignored="false"` attribute in <%@ page %> tag

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" isELignored="false"
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
```

Step 3 = Remove all the other configuration that you have done till now in web.xml or anywhere else :)

Step 4 = Clean the Tomcat and restart Tomcat.

Side Note = actually, JSTL will work with only 2.x Servlet specifications on Tomcat 8.

```
[http://www.br/blog.com/2018/jsp/jstl-with-spring-8-on-tomcat-8.html]
```

share improve this answer — edited Feb 21 '19 at 15:26 Reeno — 3,138 · 9 · 32 · 48 — asked Feb 21 '19 at 15:06 user1788142 — 77 · 4 · 1

Servlet and JSP APIs don't belong in webapp. They are already provided by Tomcat itself – BalusC Feb 21 '19 at 17:43 ✓

add a comment

To make it work:

Add jstl and **standard** jar files to your library.

Hope it helps. :)

share improve this answer — answered May 4 '17 at 8:24 Gaval Chandra — 2,288 · 14 · 21

add the maven dependencies:

```
<dependency>
    <groupId>jstl</groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
<dependency>
    <groupId>taglibs</groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
</dependency>
```

share improve this answer — answered Jul 11 '17 at 12:17 Glen M — 434 · 2 · 10 · 23

add a comment

Simply download **javax.servlet.jsp.jstl.jar** and add to your build path and `WEB-INF/lib` if you simply developing dynamic web application

When you develop dynamic web application using maven then add **javax.servlet.jsp.jstl** dependency in pom file.

Thanks

nirmal\rajarapeev

share improve this answer — edited Nov 30 '17 at 5:06 Chinghathak — 18k · 10 · 44 · 85 — asked Nov 30 '17 at 4:50 Rajapeev Kumar — 18 · 1 · 4

add a comment

Make sure that both the below jars are present in the build path:

1. standard-1.1.2.jar
2. jstl-api-1.2.jar

Maven Dependencies are:

```
<dependency>
    <groupId>javax.servlet.jsp.jstl</groupId>
    <artifactId>jstl-api</artifactId>
    <version>1.2</version>
</dependency>
<dependency>
    <groupId>taglibs</groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
</dependency>
```

share improve this answer — answered Feb 16 '18 at 8:25 Anshul Sood — 75 · 3

This works as well:

```
<dependency>
    <groupId>javax.servlet/groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>

<dependency>
    <groupId>taglibs/groupId>
    <artifactId>standard</artifactId>
    <version>1.1.2</version>
</dependency>
```

so used jstl 1.2 instead of standard.jar together with jstl-api 1.2

share improve this answer — answered Mar 6 at 15:35 Tomas Antos — 138 · 1 · 5

add a comment

What worked for me (I use Java 8 and Tomcat 9 in Eclipse 2019):

pom.xml

```
<!-- jstl -->
    <dependency>
        <groupId>javax.servlet/groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.1.0</version>
    </dependency>
    <dependency>
        <groupId>javax.servlet.jsp.jstl</groupId>
        <artifactId>javax.servlet.jsp.jstl-api</artifactId>
        <version>1.2.1</version>
    </dependency>
    <dependency>
        <groupId>javax.servlet/groupId>
        <artifactId>standard</artifactId>
        <version>1.1.2</version>
    </dependency>
```

jsp file

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
```

share improve this answer — answered Apr 9 at 0:16 Johanna — 1 · 1

add a comment

Your Answer

Sign up or log in
Sign up using Google
Sign up using Facebook
Sign up using Email and Password
Post a Question
Name
Email
Required, but never shown
Post Your Answer
By clicking "Post Your Answer", you agree to our terms of service, privacy policy and cookie policy

Not the answer you're looking for? Browse other questions tagged jsp, eclipse, tomcat, jstl or ask your own question.

Senior Analyst, Devops & Automation
Charlotte, Australia
amazon-web-services ansible
Senior Cloud Architect
Docklands, Australia
cloud devops
Senior Analyst, Microservice Engineer
Docklands, Australia
aws
View all 5 job openings!
QR Technologies
Abbotsford, Australia
Software Engineer: IoT Platform Development
Abbotsford, Australia AWS ASTON
aws
View all job openings!
96 people chatting
Arsenal
Jesse_Pinkman
Lil Peep
Jake
Zoe
Zoe
Zoe