

Understanding the New Features of TLD Caching in JSPs

The use of TLD caching speeds performance at application startup and during JSP page translation.

The following sections provide additional information:

Contents

1. [TLD Caching and Well-Known Tag Library Locations](#)
2. [TLD Cache Features and Files](#)
3. [Turning off the TLD Caching](#)
4. [Example: Packaging Multiple Tag Libraries and TLD Files in a JAR File](#)
5. [Resources](#)
6. [Feedback](#)

TLD Caching and Well-Known Tag Library Locations

As an extension of standard JSP "well-known URI" functionality described in the JSP specification, the OC4J JSP container supports the use of one or more directories, known as *well-known tag library locations*, where you can place tag library JAR files to be shared across multiple Web applications.

Beginning with the OC4J 9.0.4 implementation, there is also a persistent caching feature for TLD files, with a global cache for TLD files in any well-known tag library locations, as well as an application-level cache for any application that uses TLD caching.

Here we will specifically cover the following:

[Mechanism to enable/disable the TLD Caching](#)
[Important points to take care of](#)

Mechanisms to enable/disable the TLD Caching

Let us see in detail how the TLD Caching can be enabled or disabled in OC4J 9.0.4:

[Enabling the TLD Caching](#)
[Disabling the TLD Caching](#)

Enabling TLD Caching

TLD caching is enabled or disabled through the `jsp-cache-tlds` attribute of the `<orion-web-app>` element, at a global level through this attribute in the `global-web-application.xml` file, or at an application level through this attribute in the application `orion-web.xml` file.

By default, TLD caching is enabled at a global level through the default setting `jsp-cache-tlds="true"` in `global-web-application.xml`. This is also the default setting in the `orion-web.xml` file of each application.

If TLD caching is enabled, you can specify one or more well-known tag library locations using a semicolon-delimited list of directory paths in the `jsp-taglib-locations` attribute of the `<orion-web-app>` element in `global-web-application.xml`. See "[OC4J Configuration Parameters for JSP](#)" for additional information about this attribute.

Disabling TLD Caching

As mentioned above, by default, TLD caching is enabled at a global level through the default setting `jsp-cache-tlds="true"` in `global-web-application.xml`. This is also the default setting in the `orion-web.xml` file of each application, but you can disable TLD caching for any particular application with a setting of `jsp-cache-tlds="false"` in `orion-web.xml`. This overrides the global setting.

Alternatively, you can disable TLD caching at a global level with a "false" setting in `global-web-application.xml`, then optionally enable TLD caching for any particular application with a "true" setting in `orion-web.xml`.

If TLD caching is disabled, the well-known tag library location is limited to a single directory, using functionality that existed prior to the availability of TLD caching. In this case, the well-known location is determined by the `well_known_taglib_loc` JSP configuration parameter. See "[JSP Configuration Parameters](#)" for additional information about this parameter.

In an Oracle Application Server environment, the default well-known location is ORACLE_HOME/j2ee/home/jsp/lib/taglib (assuming ORACLE_HOME is defined).

Important Points to take care of

- By default, orion-web.xml inherits its jsp-cache-tlds setting from global-web-application.xml.
- Use the jsp-taglib-locations attribute only in global-web-application.xml, not in orion-web.xml.
- For any application to pick up files in the well-known location or locations, the directory or directories that are specified in jsp-taglib-locations or the directory that is specified in well_known_taglib_loc must be added to the path attribute setting of the <library> element in the OC4J global application.xml file in the configuration files directory (j2ee/home/config by default in OC4J standalone). See the [Oracle Application Server Containers for J2EE User's Guide](#) for information about application.xml.
- If a TLD file is present both in the well-known location and under the /WEB-INF directory of an application, the /WEB-INF copy takes precedence and is used.
- If TLD files with the same URI value are present in or under the /WEB-INF directory and also in a JAR file in the /WEB-INF/lib directory, the decision of which one to use is indeterminate. Avoid this situation.

TLD Cache Features and Files

For any application that uses TLD caching, whether it is enabled at the global level or at the application level, there are two levels of caching, and two aspects of caching at each level. Let us see these caching levels in detail:

[Caching Levels](#)

[Caching aspects at each caching level](#)

[Important Notes](#)

Caching Levels

There is a global cache for TLD files that are in JAR files in any well-known tag library locations.

- There is an application-level cache for TLD files under the application /WEB-INF directory.

At the application level, tag library JAR files, which include TLD files, must be in the /WEB-INF/lib directory. Individual TLD files can be directly in /WEB-INF or in any subdirectory, but preferably not in /WEB-INF/lib or /WEB-INF/classes.

Caching aspects at each caching level

- There is a file containing resource information for the relevant location-- the well-known location for the global cache, or /WEB-INF or /WEB-INF/lib for the application-level cache. Because of this feature, JAR files do not have to be scanned more than once. The file contains two types of entries:
 - There is a list of all resources (tag library JAR files) that includes a timestamp for each resource so that any change to any resource can be detected. There is also an indication ("true" or "false") of whether each resource includes a TLD file.
 - There is a list of TLD files, where each entry consists of a TLD name, TLD URI value if present, and tag library listeners if present. (See "[Tag Library Event Listeners](#)".)
- There is a serialized DOM representation of each TLD file. Because of this feature, TLD files do not have to be parsed more than once.

Global Cache

The global cache is always located in a directory called `tidcache`, parallel to the configuration directory. The `tidcache` directory contains the following:

- There is a file, `_GlobalTldCache`, that contains resource information, as described above, for any well-known locations.
- There are DOM representations of the TLD files that are in well-known locations. For each TLD file that is in a JAR file in a well-known location, the DOM representation is in a subdirectory according to the name of the JAR file, with a file name according to the name of the TLD file. For

example, if email.tld is found in ojsputil.jar in a well-known location, then its DOM representation would be in the following file (file name email in directory ojsputil_jar):

ORACLE_HOME/j2ee/home/jsp/lib/taglib/persistence/ojsputil_jar/email

This is for an Oracle Application Server environment, where ORACLE_HOME is defined. In OC4J standalone, the j2ee directory is relative to where OC4J is installed.

Application-level Cache

The application-level cache is in the directory indicated by the jsp-cache-directory setting in either global-web-application.xml or orion-web.xml. (See "[OC4J Configuration Parameters for JSP](#)" for information about jsp-cache-directory.) This directory contains the following:

=> There is a file, _TldCache, that contains resource information, as described above, for TLD files under the /WEB-INF directory--either in JAR files in /WEB-INF/lib, or individually in /WEB-INF or any subdirectory, but preferably not /WEB-INF/lib or /WEB-INF/classes.

=> There are DOM representations of the TLD files under /WEB-INF. For TLD files that are in JAR files in the /WEB-INF/lib directory, the DOM representations go into subdirectories under the directory indicated by jsp-cache-directory, in the same type of scheme as described for the global cache. For individual TLD files under /WEB-INF, the DOM representations go directly in the jsp-cache-directory location.

Important Notes

- TLD changes at the global level are reflected only after OC4J is restarted.
- TLD changes at the application level are reflected immediately in an OC4J standalone environment, but only after the application is restarted in an Oracle Application Server environment.
- You can increase the OC4J verbosity level to see information regarding construction of TLD caches and regarding any TLD URIs that are duplicated. Level 4 provides some information; level 5 provides additional information. You can use Oracle Enterprise Manager to set the verbosity level. The default level is 3.

Turning off the TLD Caching

The use of TLD caching speeds performance at application startup and during JSP page translation. However, you might typically turn it off under either of the following circumstances:

- Your application does not use tag libraries.

or:

- You have pretranslated the JSP pages and none of the TLD files use `<listener>` elements for tag library event listeners. (See "[Tag Library Event Listeners](#)".)

Example: Multiple Tag Libraries and TLD Files in a JAR File

This section assumes an example JAR file and presents an example of tag library packaging. This is a situation where multiple tag libraries are packaged in a single JAR file. The JAR file includes tag handler classes, tag-library-validator classes, and TLD files for multiple libraries. The following shows the contents and structure of the example JAR:

```
examples/BasicTagParent.class
examples/ExampleLoopTag.class
examples/BasicTagChild.class
examples/BasicTagTLV.class
examples/TagElemFilter.class
examples/XMLViewTagTLV.class
examples/TagFilter.class
examples/XMLViewTag.class
META-INF/xmlview.tld
META-INF/exampletag.tld
META-INF/basic.tld
META-INF/MANIFEST.MF
```

We will specifically cover the following:

[Key TLD File Entries for Multiple-Library Example](#)
[Key web.xml File Entries for Multiple-Library Example](#)

Key TLD File Entries for Multiple-Library Example

This section illustrates the <uri> elements of the TLD files.

The basic.tld file includes the following:

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>1.2</jsp-version>
  <short-name>basic</short-name>
  <uri>http://xmlns.oracle.com/j2ee/jsp/tld/demos/basic.tld</uri>
  ...
</taglib>
```

The exampletag.tld file includes the following:

```
<taglib xmlns="http://java.sun.com/JSP/TagLibraryDescriptor">
  <tlib-version>1.0</tlib-version>
  <jsp-version>1.2</jsp-version>
  <short-name>example</short-name>
  <uri>http://xmlns.oracle.com/j2ee/jsp/tld/demos/exampletag.tld</uri>
  ...
</taglib>
```

The xmlview.tld file includes the following:

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>1.2</jsp-version>
  <short-name>demo</short-name>
  <uri>http://xmlns.oracle.com/j2ee/jsp/tld/demos/xmlview.tld</uri>
  ...
</taglib>
```

Key web.xml File Entries for Multiple-Library Example

This section shows the <taglib> elements of the web.xml deployment descriptor. These map the full URI values, as seen in the <uri> elements of the TLD files in the previous section, to shortcut URI values used in the JSP pages that access these libraries.

```
...
<taglib>
  <taglib-uri>/oraloop</taglib-uri>
  <taglib-location>http://xmlns.oracle.com/j2ee/jsp/tld/demos/exampletag.tld
  </taglib-location>
</taglib>
<taglib>
  <taglib-uri>/orabasic</taglib-uri>
```

```

<taglib-location>http://xmlns.oracle.com/j2ee/jsp/tld/demos/basic.tld
</taglib-location>
</taglib>
<taglib>
<taglib-uri>/oraxmlview</taglib-uri>
<taglib-location>http://xmlns.oracle.com/j2ee/jsp/tld/demos/xmlview.tld
</taglib-location>
</taglib>
...

```

JSP Page taglib Directives for Multiple-Library Example

This section shows the appropriate taglib directives, which reference the shortcut URI values defined in the web.xml elements listed in the preceding section.

The page basic1.jsp includes the following directive:

```
<%@ taglib prefix="basic" uri="/orabasic" %>
```

The page exampletag.jsp includes the following directive:

```
<%@ taglib prefix="example" uri="/oraloop" %>
```

The page xmlview.jsp includes the following directive:

```
<%@ taglib prefix="demo" uri="/oraxmlview" %>
```

Resources

This resource section is part of a tutorial series titled '[Understanding the New Features of TLD Caching in JSPs](#)'. Following are links to resources that can help you understand and apply the concepts and techniques presented in the tutorials.

Resource	URL
JSP Tag Libraries	http://download.oracle.com/docs/cd/B12314_01/web.904/b10320/taglibs.htm
JSP Tag Library and Utilities Reference 10g	http://download.oracle.com/docs/cd/B12314_01/web.904/b10319/toc.htm

Java Server Pages Developer' s Guide 10g	http://download.oracle.com/docs/cd/B12314_01/web.904/b10320/toc.htm
OTN Sample Code	http://otn.oracle.com/sample_code/content.html

Feedback

If you have questions or comments about this tutorial, you can:

- Post a message in the [OTN Sample Code discussion forum](#). OTN developers and other experts monitor the forum.
- Send email to the author. <mailto:Shefali.Bansal@oracle.com>

If you have suggestions or ideas for future tutorials, you can:

- Post a message in the [OTN Member Feedback forum](#).
- Send email to <mailto:Tom.Haunert@oracle.com>