Class Loading Conflicts in JVM

This guide shows how to analyze and avoid potential problems caused by class loading conflicts. The content is structured in the following sections, the first one gives a little introduction to the Classloader model in Weblogic [1], following the installation of Classloader Analysis Tool (CAT) [1] is shown and then two examples that shows the use of CAT on a real application is presented.

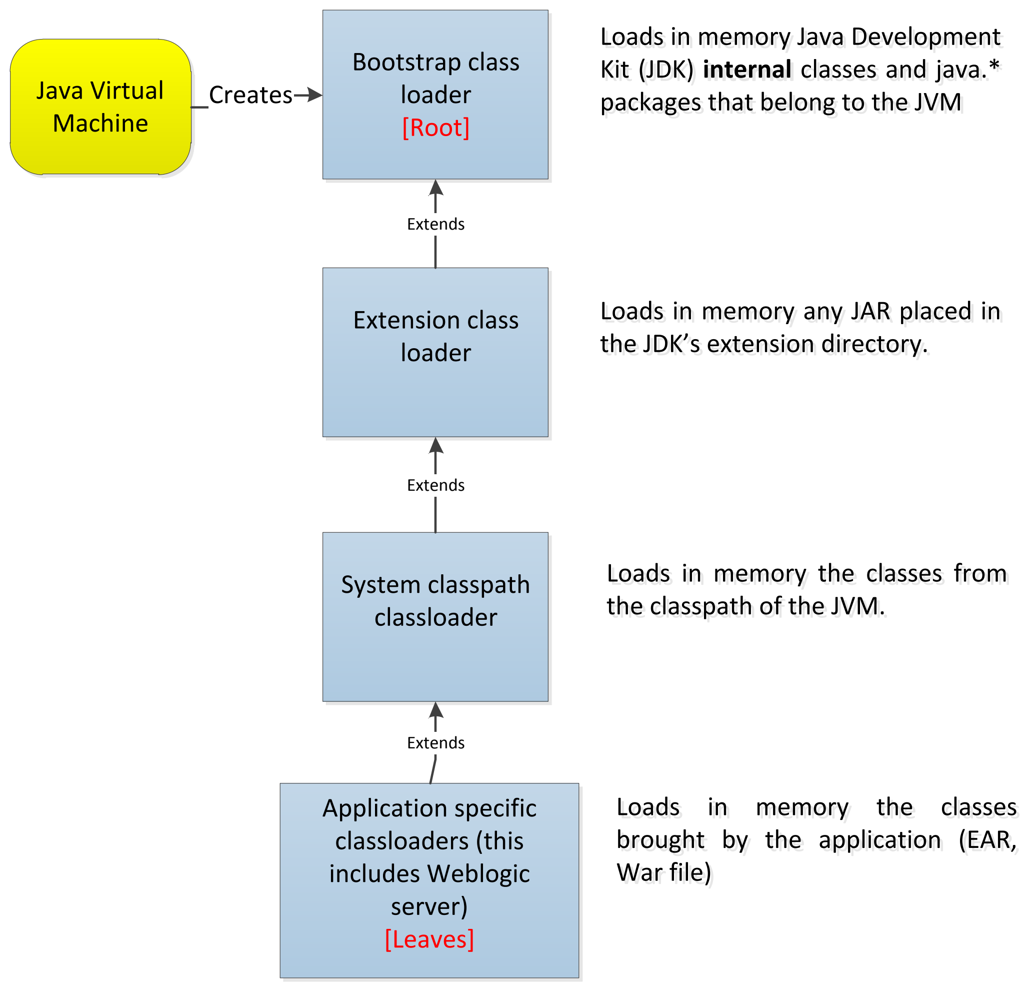
Class loading in Weblogic

Summarizing the class loading process in just some lines is hard so in this guide the focus is talking about hierarchies and the delegation model.

Concept	Definition
Hierarchies	The classloader in an application server such as Weblogic is based on the model defined by the JVM, which means a hierarchical model that on Weblogic is organized as a tree with these levels: bootstrap class loader, extension class loader, system classpath classloader, application specific classloaders (this includes Weblogic server) [1]. In the previous tree, bootstrap class loader is the root and application specific classloaders are the leaves [1].
Delegation model	A common question that arises when Java application servers are used is why is my application using the wrong class? This is because the delegation model, which states “*The classloader implementation first checks its cache to see if the requested class has already been loaded. This class verification improves performance in that its cached memory copy is used instead of repeated loading of a class from disk. If the class is not found in its cache, the current classloader asks its parent for the class. Only if the parent cannot load the class does the classloader attempt to load the class.* ***If a class exists in both the parent and child classloaders, the parent version is loaded”*** [1]

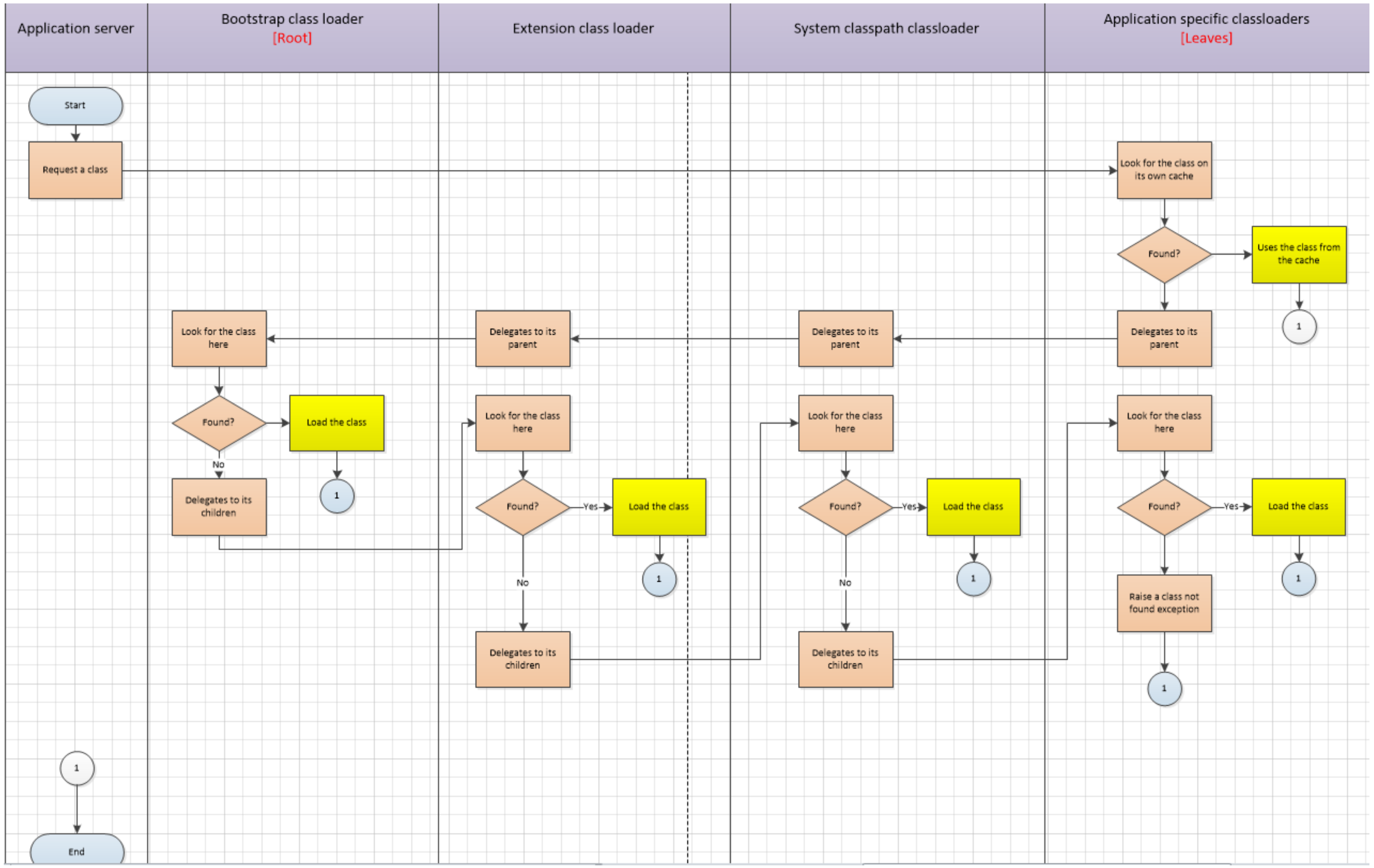
The following picture depicts this hierarchy of classloaders.

Java Classloader Hierarchy


Java Virtual Machine
Creates
Bootstrap class loader
[Root]
Loads in memory Java Development Kit (JDK) **internal** classes and java.* packages that belong to the JVM
Extends
Extension class loader
Loads in memory any JAR placed in the JDK's extension directory.
Extends
System classpath classloader
Loads in memory the classes from the classpath of the JVM.
Extends
Application specific classloaders (this includes Weblogic server)
[Leaves]
Loads in memory the classes brought by the application (EAR, War file)

The hierarchy of classloaders in Java

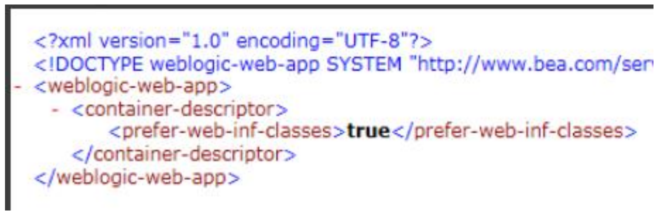
In the previous figure let us say one application specific classloader wants to load a class so the following diagram depicts this process

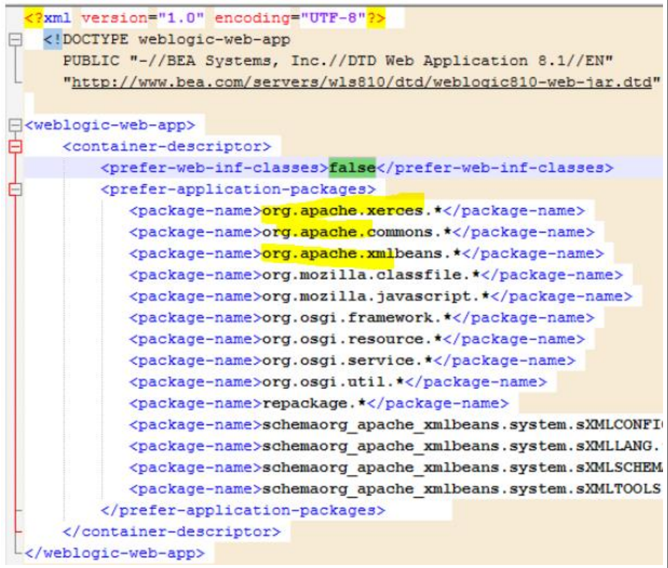


In the previous diagram is possible to see that because of the delegation process even if the application specific classloader has the requested class, this will be loaded by a superior classloaders in the hierarchy if the class exists on one of the superior levels otherwise the class is loaded by the application specific classloader.

How could we subvert the previous process?

Of course, there are ways to subvert this process to allow us using our own libraries as is described in the following table.

Way to subvert the process	Definition	Advantages	Disadvantages
prefer-web-inf-classes Element	"If true, classes located in the WEB-INF directory of a web-app will be loaded in preference to classes loaded in the application or system classloader" [1]	• This is the easiest way to subvert the delegation model to use your own classes.  `<?xml version="1.0" encoding="UTF-8"?>` `<!DOCTYPE weblogic-web-app SYSTEM "http://www.bea.com/ser` `- <weblogic-web-app>` `- <container-descriptor>` `<prefer-web-inf-classes>true</prefer-web-inf-classes>` `</container-descriptor>` `</weblogic-web-app>`	• With this you can include some undesirable classes that are part of the library because a library can include many packages and classes so you should know the library thoroughly before subverting the delegation model in this way. • This is prone to be affected by new bugs introduced on latest version of JVMs as can be seen in these two bugs [3] and [4]. • Sometimes can be unpredictable as can be seen in the second example shown below on this document.
Using a Filtering ClassLoader	This is mechanism to use third party libraries telling the class loader which packages are going to be loaded by the application classloader rather than the system classloader [1].	• The implementation could be a little difficult because each needed package should be mentioned specifically. • The risk of including specific undesirable classes is minimized because it specifies packages instead of the whole jar. • It looks more stable to face possible bugs since even is recommended as a work around for the bug described in [4]	• The implementation can be tricky because we have to provide details about each package we want our application to use.

		• It has been used here to fix a problem that is described in the second example shown below on this document. 	

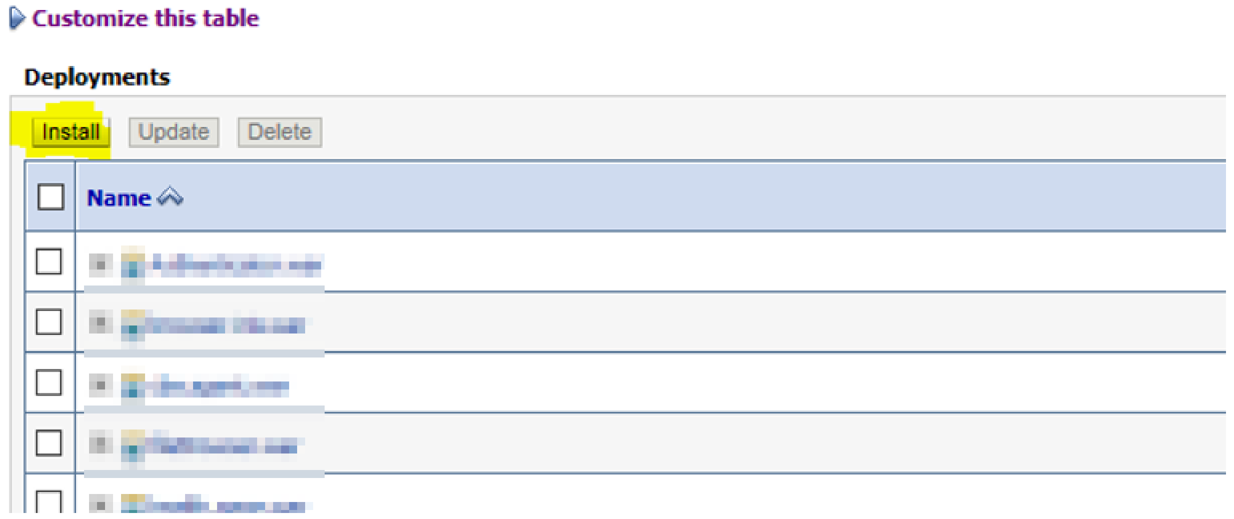
```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE weblogic-web-app
    PUBLIC "-//BEA Systems, Inc.//DTD Web Application 8.1//EN"
    "http://www.bea.com/servers/wls810/dtd/weblogic810-web-jar.dtd"

<weblogic-web-app>
    <container-descriptor>
        <prefer-web-inf-classes>false</prefer-web-inf-classes>
        <prefer-application-packages>
           <package-name>org.apache.xerces.*</package-name>
           <package-name>org.apache.commons.*</package-name>
           <package-name>org.apache.xmlbeans.*</package-name>
           <package-name>org.mozilla.classfile.*</package-name>
           <package-name>org.mozilla.javascript.*</package-name>
           <package-name>org.osgi.framework.*</package-name>
           <package-name>org.osgi.resource.*</package-name>
           <package-name>org.osgi.service.*</package-name>
           <package-name>org.osgi.util.*</package-name>
           <package-name>repackage.*</package-name>
           <package-name>schemaorg_apache_xmlbeans.system.sXMLCONFI
           <package-name>schemaorg_apache_xmlbeans.system.sXMLLANG.
           <package-name>schemaorg_apache_xmlbeans.system.sXMLSCHEM
           <package-name>schemaorg_apache_xmlbeans.system.sXMLTOOLS
        </prefer-application-packages>
    </container-descriptor>
</weblogic-web-app>
```

Installing Classloader Analysis Tool (CAT)

This are the steps to install CAT, which is a file called **wls-cat.war** located on $WL_HOME/server/lib/wls-cat.war

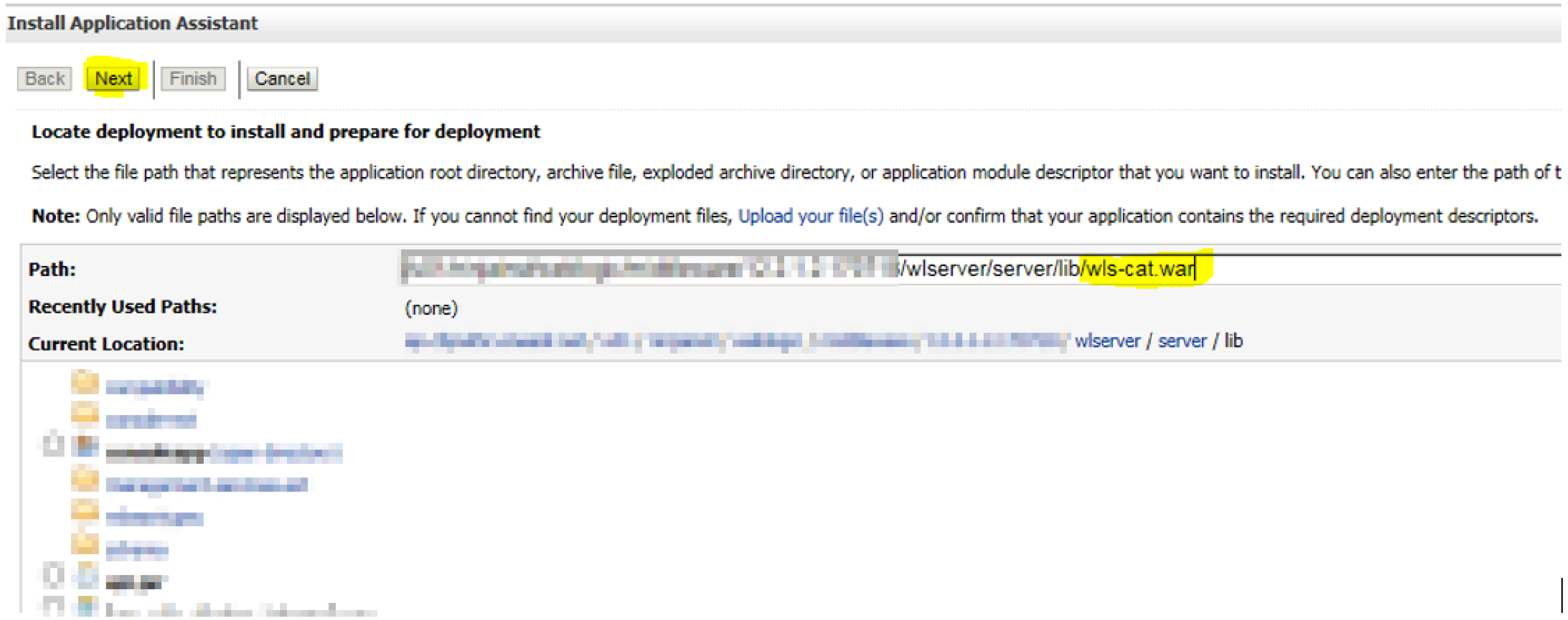
1. After login into the console with an admin user, lock the console to edit and click on Install



Customize this table

Deployments

2. Look for the application $WL_HOME/server/lib/wls-cat.war and click on Next.



Install Application Assistant

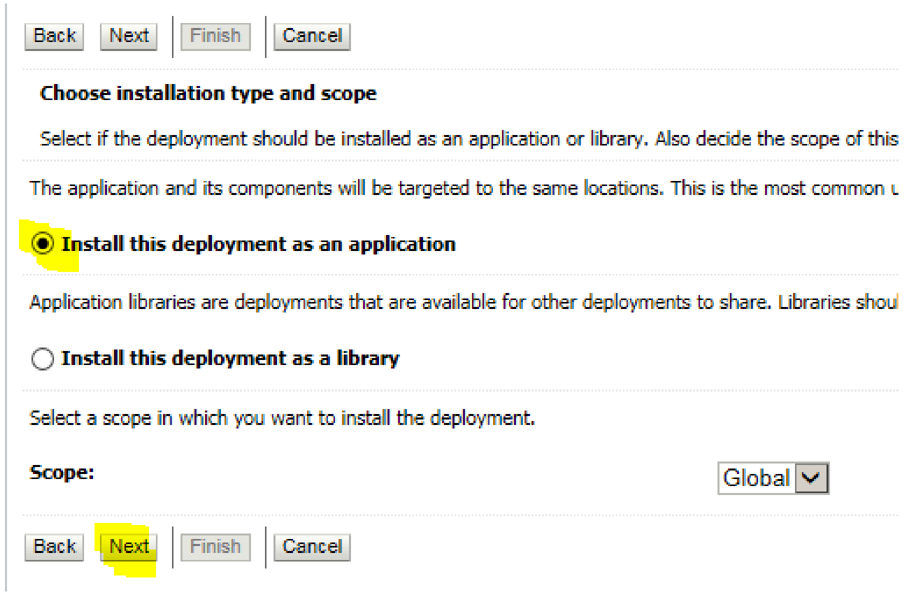
Locate deployment to install and prepare for deployment

Select the file path that represents the application root directory, archive file, exploded archive directory, or application module descriptor that you want to install. You can also enter the path of t

Note: Only valid file paths are displayed below. If you cannot find your deployment files, Upload your file(s) and/or confirm that your application contains the required deployment descriptors.

Path:	/wlserver/server/lib/wls-cat.war
Recently Used Paths:	(none)
Current Location:	wlserver / server / lib

3. Choose the remarked radio button and click on Next



Choose installation type and scope

Select if the deployment should be installed as an application or library. Also decide the scope of this

The application and its components will be targeted to the same locations. This is the most common u

[✓ **Install this deployment as an application**]

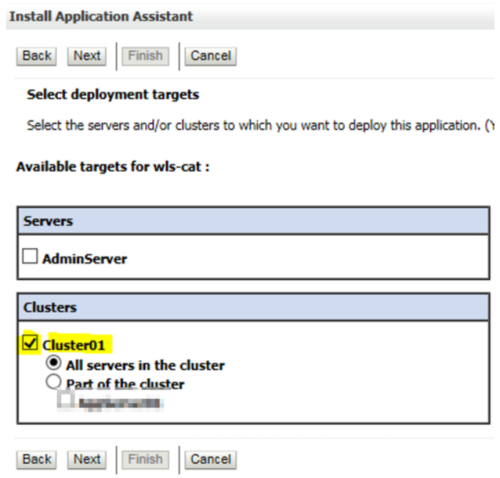
Application libraries are deployments that are available for other deployments to share. Libraries shoul

Install this deployment as a library

Select a scope in which you want to install the deployment.

Scope: Global

4. Choose the cluster and click on Next


Install Application Assistant

Select deployment targets

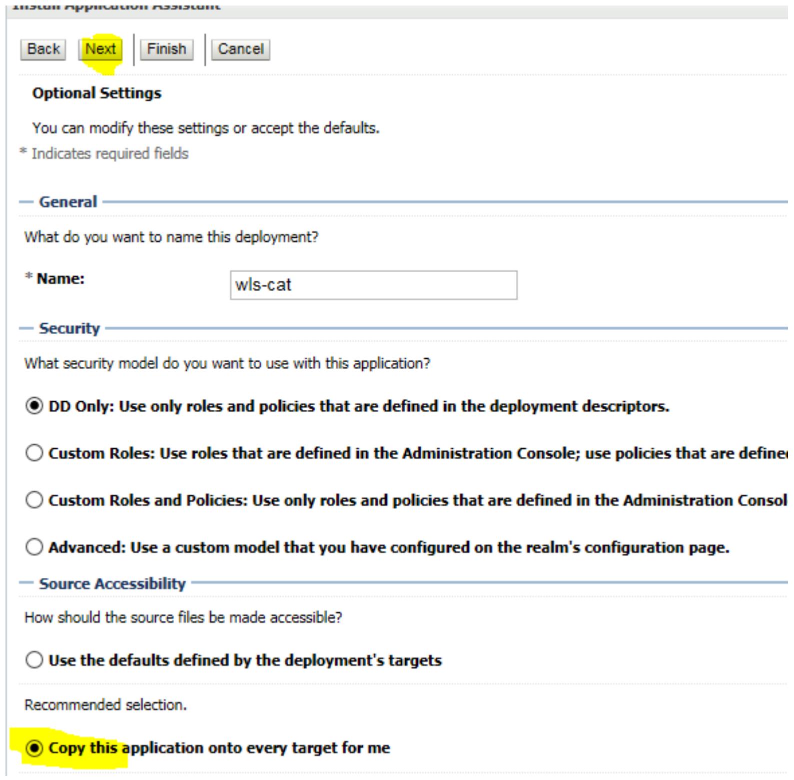
Select the servers and/or clusters to which you want to deploy this application. (Y

Available targets for wls-cat :

Servers
AdminServer

Clusters
[✓ Cluster01]
[✓ All servers in the cluster]
Part of the cluster
[redacted]

5. Choose the remarked option and click on Next



Optional Settings

You can modify these settings or accept the defaults.

* Indicates required fields

General

What do you want to name this deployment?

* **Name:** wls-cat

Security

What security model do you want to use with this application?

● **DD Only: Use only roles and policies that are defined in the deployment descriptors.**

○ **Custom Roles: Use roles that are defined in the Administration Console; use policies that are define**

○ **Custom Roles and Policies: Use only roles and policies that are defined in the Administration Consol**

○ **Advanced: Use a custom model that you have configured on the realm's configuration page.**

Source Accessibility

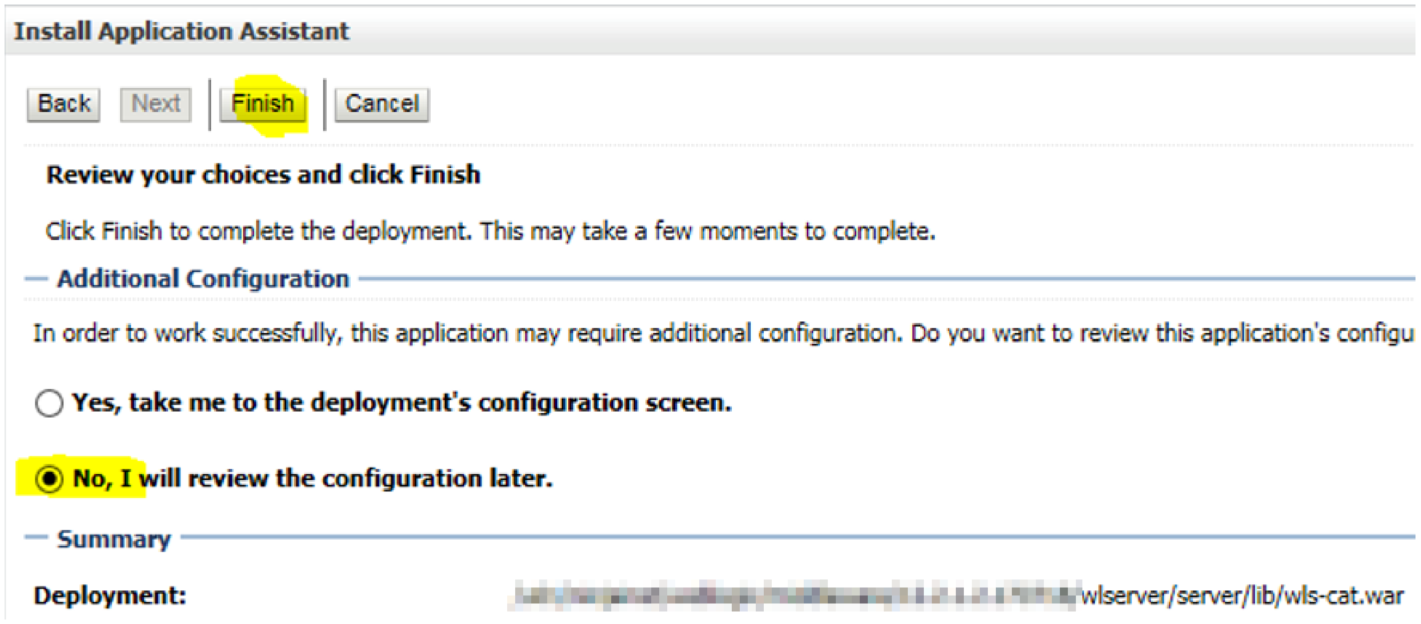
How should the source files be made accessible?

○ **Use the defaults defined by the deployment's targets**

Recommended selection.

● **Copy this application onto every target for me**

6. Select the remarked option and click on Finish



Install Application Assistant

Back | Next | Finish | Cancel

Review your choices and click Finish

Click Finish to complete the deployment. This may take a few moments to complete.

Additional Configuration

In order to work successfully, this application may require additional configuration. Do you want to review this application's configu

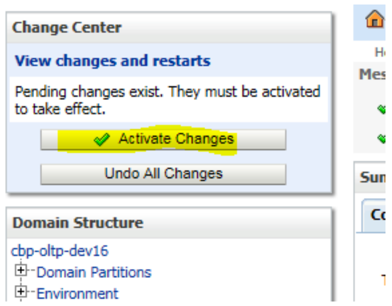
○ **Yes, take me to the deployment's configuration screen.**

● **No, I will review the configuration later.**

Summary

Deployment: [redacted]wlserver/server/lib/wls-cat.war

7. Click on Activate Changes



Change Center

View changes and restarts

Pending changes exist. They must be activated to take effect.

Activate Changes

Undo All Changes

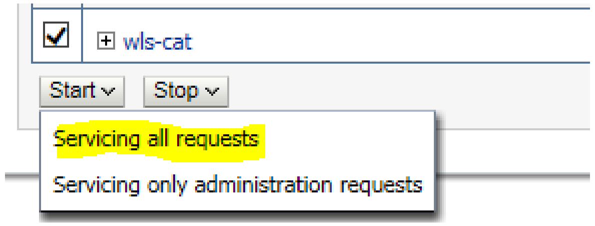
Domain Structure

cbp-oltp-dev16

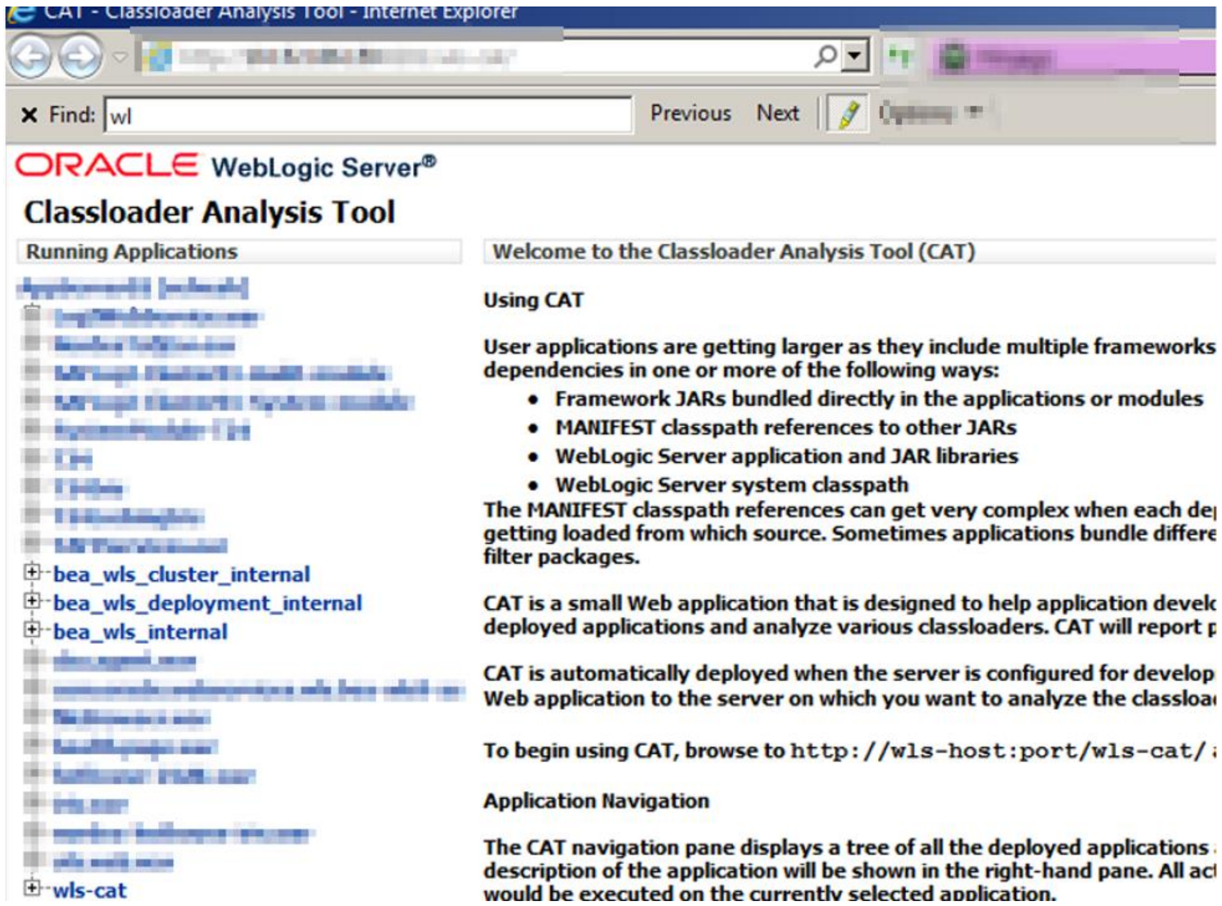
Domain Partitions

Environment

8. Go to Deployments > Control and start the application



9. Test the application using the listen address and the port assigned to each managed server inside the cluster. The application will request a user and you should user the **Weblogic** user, after login this is the main page of CAT.



Classloader Analysis Tool

Running Applications

[redacted]
bea_wls_cluster_internal
bea_wls_deployment_internal
bea_wls_internal
[redacted]
wls-cat

Welcome to the Classloader Analysis Tool (CAT)

Using CAT

User applications are getting larger as they include multiple frameworks
dependencies in one or more of the following ways:

- Framework JARs bundled directly in the applications or modules
- MANIFEST classpath references to other JARs
- WebLogic Server application and JAR libraries
- WebLogic Server system classpath

The MANIFEST classpath references can get very complex when each de
getting loaded from which source. Sometimes applications bundle differe
filter packages.

CAT is a small Web application that is designed to help application devel
deployed applications and analyze various classloaders. CAT will report p

CAT is automatically deployed when the server is configured for develop
Web application to the server on which you want to analyze the classloa

To begin using CAT, browse to `http://wls-host:port/wls-cat/`

Application Navigation

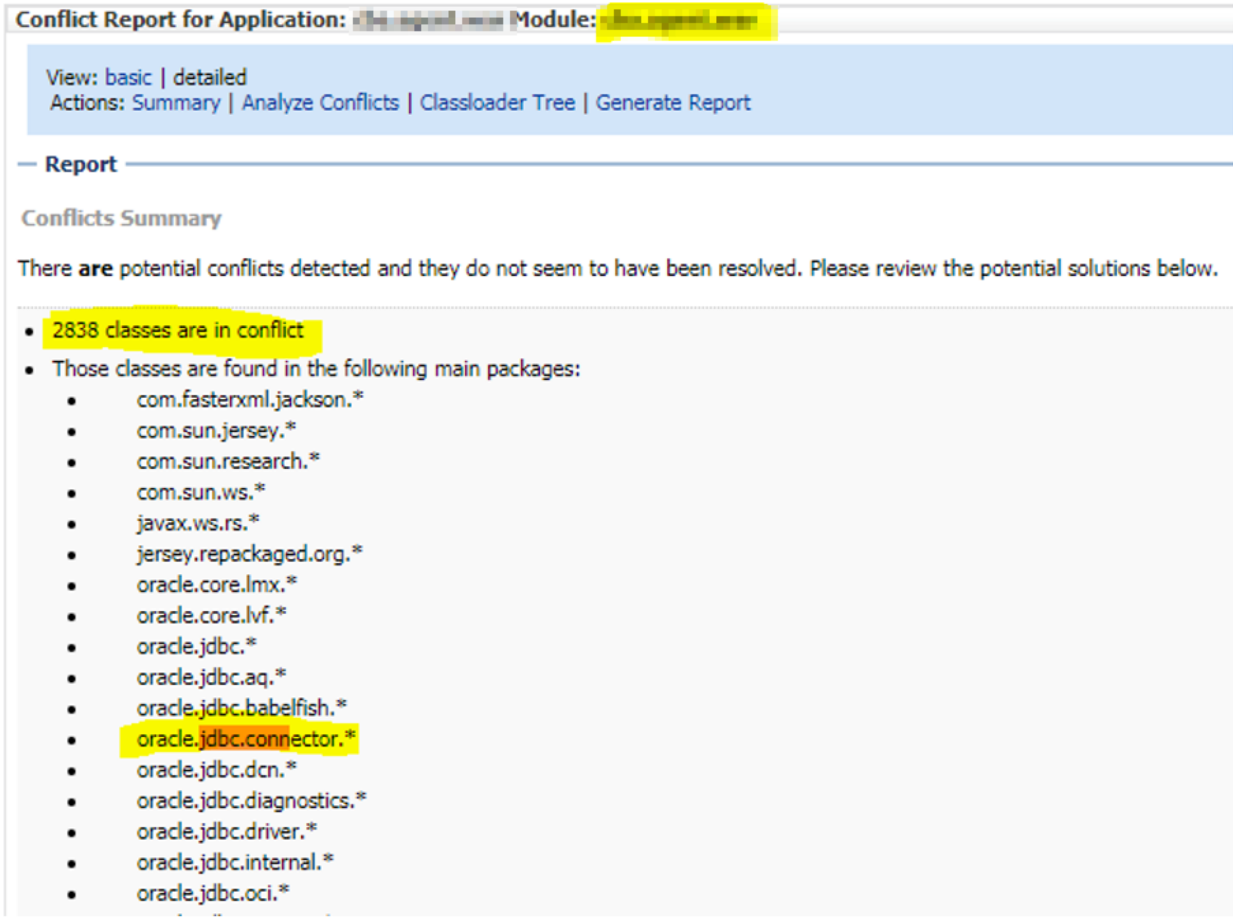
The CAT navigation pane displays a tree of all the deployed applications
description of the application will be shown in the right-hand pane. All ac
would be executed on the currently selected application.

Using CAT to analyze class conflict

In this section two examples related to class conflicts are show.

Example 1

The application CAT identifies class conflicts



Conflict Report for Application: [redacted] **Module:** [redacted]

View: basic | detailed
Actions: Summary | Analyze Conflicts | Classloader Tree | Generate Report

Report

Conflicts Summary

There **are** potential conflicts detected and they do not seem to have been resolved. Please review the potential solutions below.

- 2838 classes are in conflict
- Those classes are found in the following main packages:
  - com.fasterxml.jackson.*
  - com.sun.jersey.*
  - com.sun.research.*
  - com.sun.ws.*
  - javax.ws.rs.*
  - jersey.repackaged.org.*
  - oracle.core.lmx.*
  - oracle.core.lvf.*
  - oracle.jdbc.*
  - oracle.jdbc.aq.*
  - oracle.jdbc.babelfish.*
  - oracle.jdbc.connector.*
  - oracle.jdbc.dcn.*
  - oracle.jdbc.diagnostics.*
  - oracle.jdbc.driver.*
  - oracle.jdbc.internal.*
  - oracle.jdbc.oci.*

As an example we can analyse oracle.jdbc.* to be specific the class called **oracle.jdbc.connector.OracleLocalTransaction** as can be seen in the following report generated by CAT.

Resource: oracle.jdbc.connector.OracleLocalTransaction

Checksum: 1f8d1e637d6813c0d486ff626c60f1d2
Load Location: jar:file: $WL_HOME/oracle_common/modules/oracle.jdbc/ojdbc7.jar!/oracle/jdbc/connector/OracleLocalTransaction.class
Classloader Type: com.oracle.classloader.weblogic.LaunchClassLoader
Classloader Hash Code: 572145572
Classloader Search Order: 318781939 ->572145572
Alternative Locations:

$DOMAIN_HOME/servers/ServerExample/tmp/_WL_user/yyy.xxx.war/2eqtxp/war/WEB-INF/lib/ojdbc7-12.1.0.2.0.jar!/oracle/jdbc/connector/OracleLocalTransaction.class

In the previous figure two class loaders are identified: **318781939 ->572145572**, which means both have the class, but only one of them loads the class. According to the previous report, this class can be loaded from two locations.

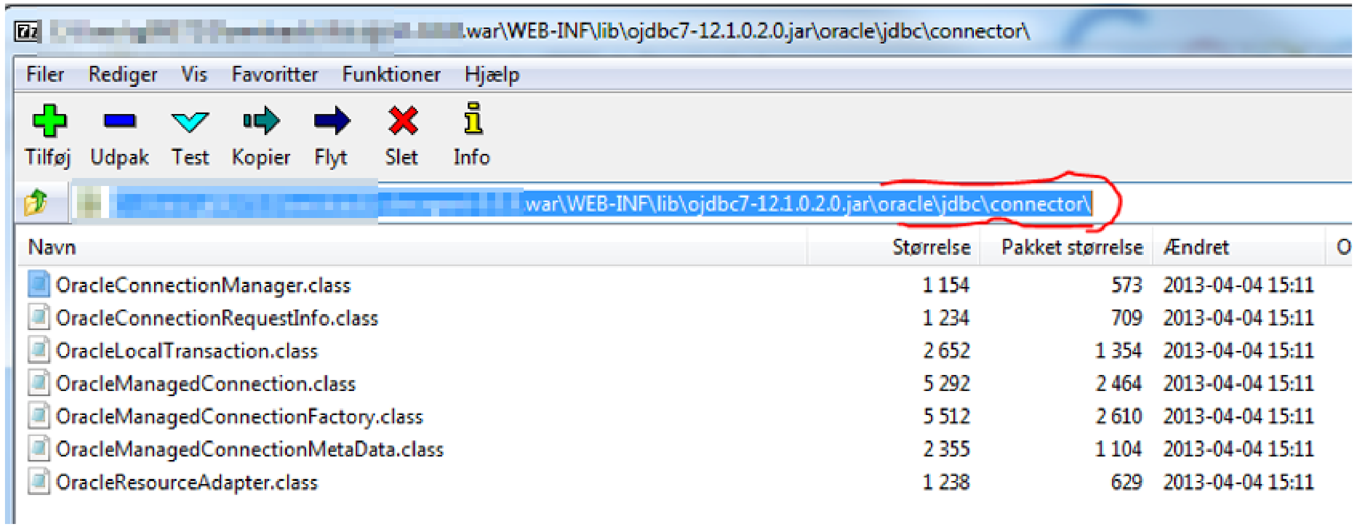
Location:

jar:file: $WL_HOME/oracle_common/modules/oracle.jdbc/ojdbc7.jar!/oracle/jdbc/connector/OracleLocalTransaction.class

Alternative Locations:

$DOMAIN_HOME/servers/ServerExample/tmp/_WL_user/yyy.xxx.war /2eqtxp/war/WEB-INF/lib/ojdbc7-12.1.0.2.0.jar!/oracle/jdbc/connector/OracleLocalTransaction.class

The following figure shows the JDBC library inside the application called yyy.xxx.war. Thus, one question is why does the application need a library that is provided by the application server?

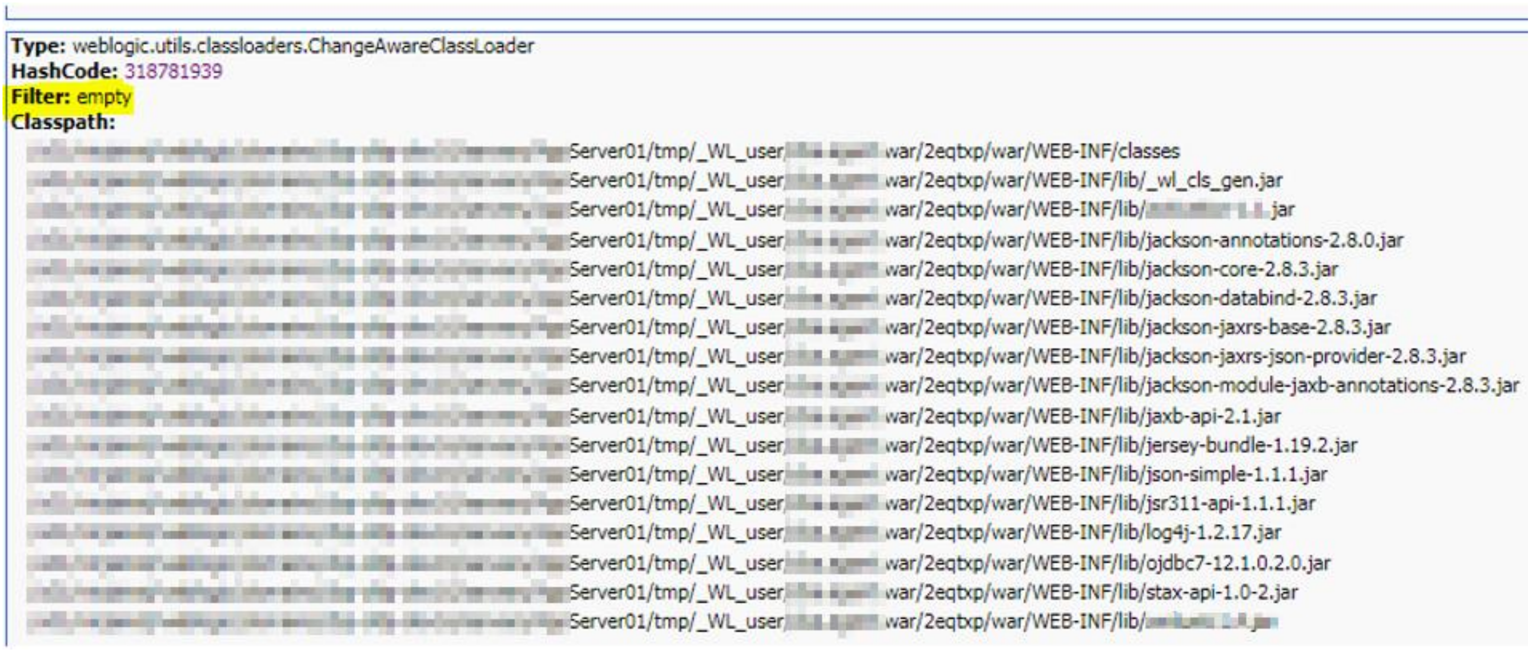


[redacted].war\WEB-INF\lib\ojdbc7-12.1.0.2.0.jar\oracle\jdbc\connector\

[redacted].war\WEB-INF\lib\ojdbc7-12.1.0.2.0.jar\oracle\jdbc\connector\

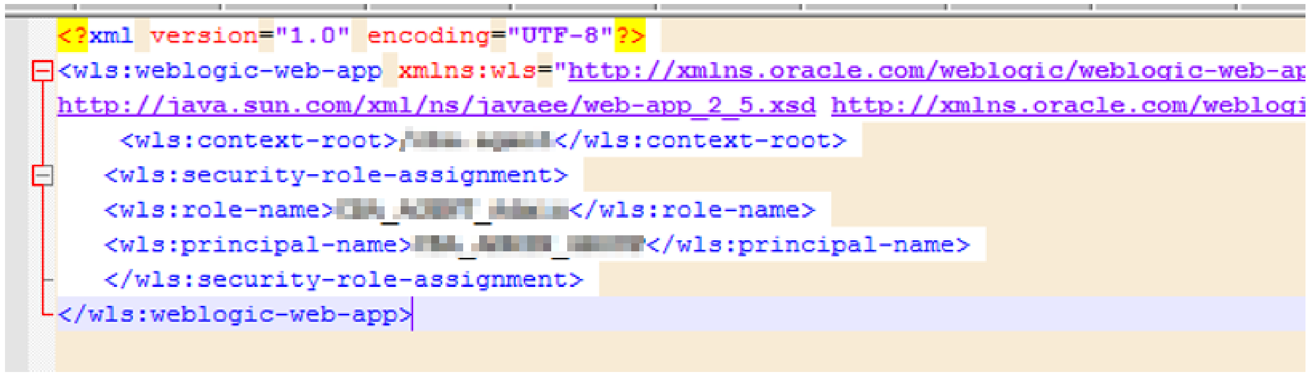
Navn	Størrelse	Pakket størrelse	Ændret
OracleConnectionManager.class	1 154	573	2013-04-04 15:11
OracleConnectionRequestInfo.class	1 234	709	2013-04-04 15:11
OracleLocalTransaction.class	2 652	1 354	2013-04-04 15:11
OracleManagedConnection.class	5 292	2 464	2013-04-04 15:11
OracleManagedConnectionFactory.class	5 512	2 610	2013-04-04 15:11
OracleManagedConnectionMetaData.class	2 355	1 104	2013-04-04 15:11
OracleResourceAdapter.class	1 238	629	2013-04-04 15:11

It is possible to see the application does not define any filter to use the library from the application instead of the Weblogic library.



```
Type: weblogic.utils.classloaders.ChangeAwareClassLoader
HashCode: 318781939
Filter: empty
Classpath:
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/classes
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/_wl_cls_gen.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/[redacted].jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-annotations-2.8.0.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-core-2.8.3.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-databind-2.8.3.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-jaxrs-base-2.8.3.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-jaxrs-json-provider-2.8.3.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jackson-module-jaxb-annotations-2.8.3.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jaxb-api-2.1.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jersey-bundle-1.19.2.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/json-simple-1.1.1.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/jsr311-api-1.1.1.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/log4j-1.2.17.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/ojdbc7-12.1.0.2.0.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/stax-api-1.0-2.jar
[redacted]Server01/tmp/_WL_user/[redacted]war/2eqtxp/war/WEB-INF/lib/[redacted]
```

This can be confirmed after reading the Weblogic.xml file

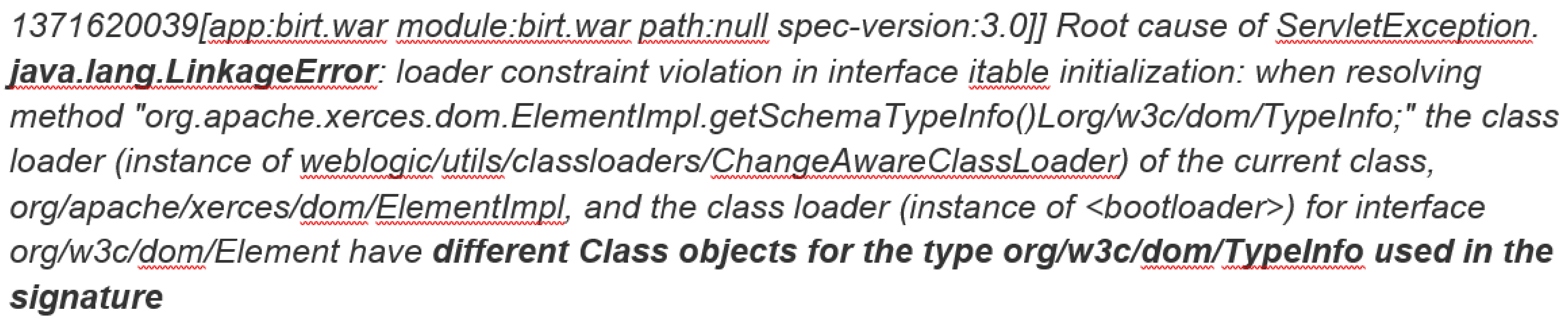


```
<?xml version="1.0" encoding="UTF-8"?>
<wls:weblogic-web-app xmlns:wls="http://xmlns.oracle.com/weblogic/weblogic-web-ap
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd http://xmlns.oracle.com/weblogi
    <wls:context-root>/[redacted]</wls:context-root>
   <wls:security-role-assignment>
   <wls:role-name>[redacted]</wls:role-name>
   <wls:principal-name>[redacted]</wls:principal-name>
   </wls:security-role-assignment>
</wls:weblogic-web-app>
```

In cases like this, the recommendation is to get rid of JAR files that do not make sense for the application since the application is using the one provided by the application server.

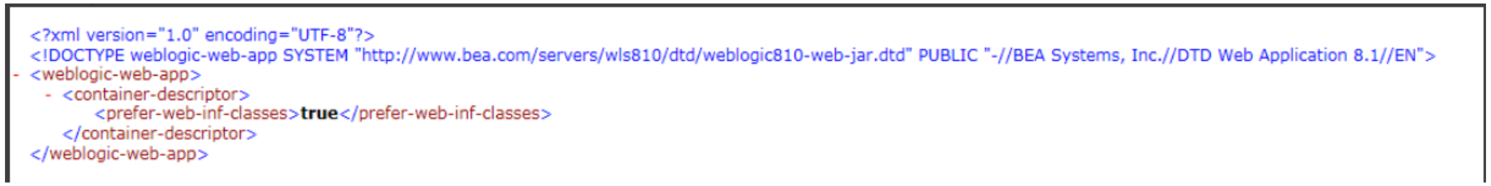
Example 2

The following case shows that even when the tag **<prefer-web-inf-classes>true</prefer-web-inf-classes>** is used there could be some problems such as:



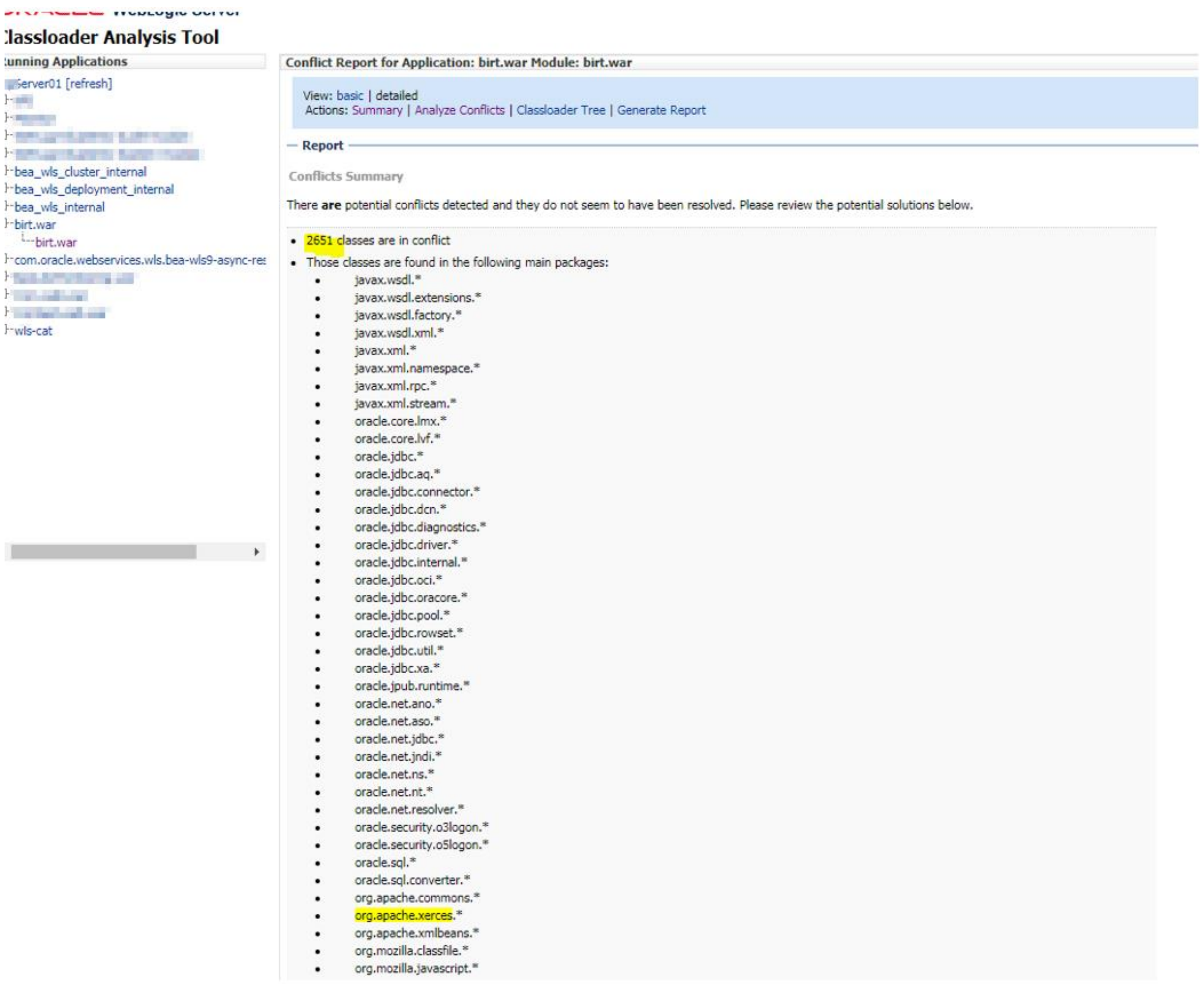
*1371620039[app:birt.war module:birt.war path:null spec-version:3.0]] Root cause of ServletException. **java.lang.LinkageError**: loader constraint violation in interface itable initialization: when resolving method "org.apache.xerces.dom.ElementImpl.getSchemaTypeInfo()Lorg/w3c/dom/TypeInfo;" the class loader (instance of weblogic/utils/classloaders/ChangeAwareClassLoader) of the current class, org/apache/xerces/dom/ElementImpl, and the class loader (instance of <bootloader>) for interface org/w3c/dom/Element have **different Class objects for the type org/w3c/dom/TypeInfo used in the signature***

According to the developer this was impossible because the application was using this



```
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE weblogic-web-app SYSTEM "http://www.bea.com/servers/wls810/dtd/weblogic810-web-jar.dtd" PUBLIC "-//BEA Systems, Inc.//DTD Web Application 8.1//EN">
- <weblogic-web-app>
  - <container-descriptor>
        <prefer-web-inf-classes>true</prefer-web-inf-classes>
    </container-descriptor>
  </weblogic-web-app>
```

With CAT it was possible to see a big number of conflicts



WebLogic Server

lassloader Analysis Tool

unning Applications

Server01 [refresh]

- [redacted]
- [redacted]
- [redacted]
- [redacted]
- bea_wls_cluster_internal
- bea_wls_deployment_internal
- bea_wls_internal
- birt.war
  - birt.war
- com.oracle.webservices.wls.bea-wls9-async-re
- [redacted]
- [redacted]
- [redacted]
- wls-cat

Conflict Report for Application: birt.war Module: birt.war

View: basic | detailed
Actions: Summary | Analyze Conflicts | Classloader Tree | Generate Report

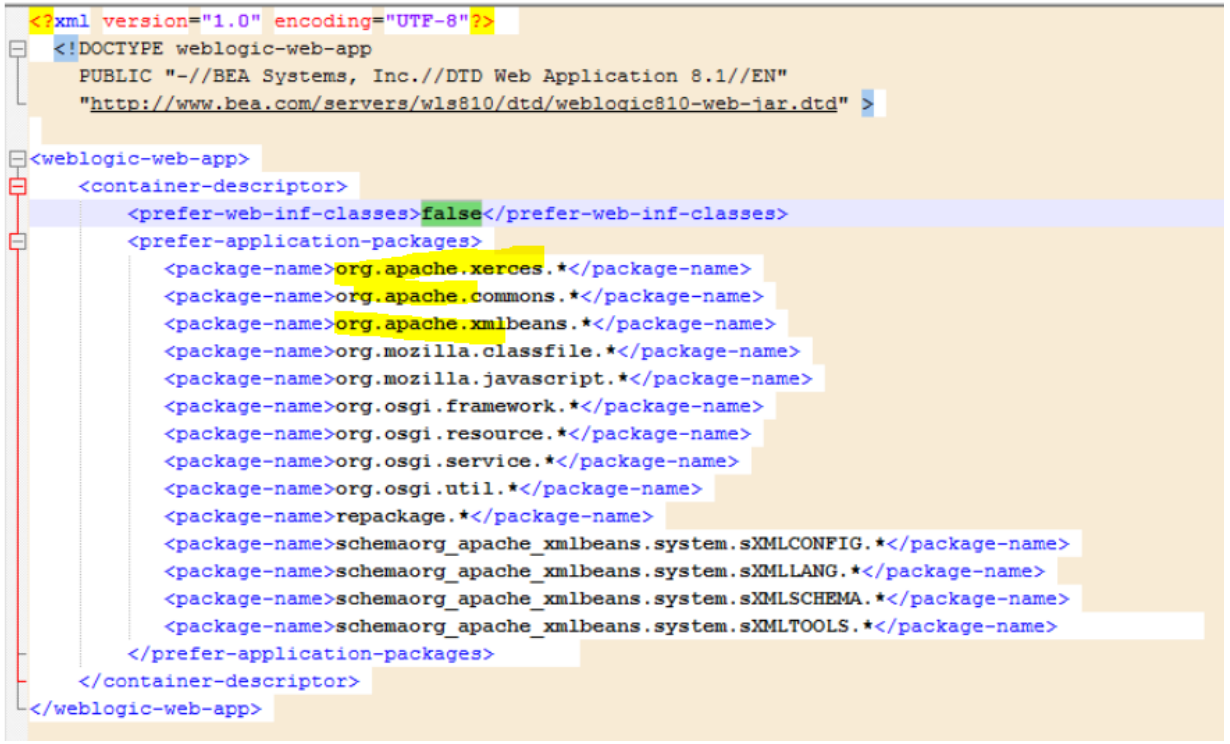
Report

Conflicts Summary

There **are** potential conflicts detected and they do not seem to have been resolved. Please review the potential solutions below.

- 2651 classes are in conflict
- Those classes are found in the following main packages:
  - javax.wsdl.*
  - javax.wsdl.extensions.*
  - javax.wsdl.factory.*
  - javax.wsdl.xml.*
  - javax.xml.*
  - javax.xml.namespace.*
  - javax.xml.rpc.*
  - javax.xml.stream.*
  - oracle.core.lmx.*
  - oracle.core.lvf.*
  - oracle.jdbc.*
  - oracle.jdbc.aq.*
  - oracle.jdbc.connector.*
  - oracle.jdbc.dcn.*
  - oracle.jdbc.diagnostics.*
  - oracle.jdbc.driver.*
  - oracle.jdbc.internal.*
  - oracle.jdbc.oci.*
  - oracle.jdbc.oracore.*
  - oracle.jdbc.pool.*
  - oracle.jdbc.rowset.*
  - oracle.jdbc.util.*
  - oracle.jdbc.xa.*
  - oracle.jpub.runtime.*
  - oracle.net.ano.*
  - oracle.net.aso.*
  - oracle.net.jdbc.*
  - oracle.net.jndi.*
  - oracle.net.ns.*
  - oracle.net.nt.*
  - oracle.net.resolver.*
  - oracle.security.o3logon.*
  - oracle.security.o5logon.*
  - oracle.sql.*
  - oracle.sql.converter.*
  - org.apache.commons.*
  - org.apache.xerces.*
  - org.apache.xmlbeans.*
  - org.mozilla.classfile.*
  - org.mozilla.javascript.*

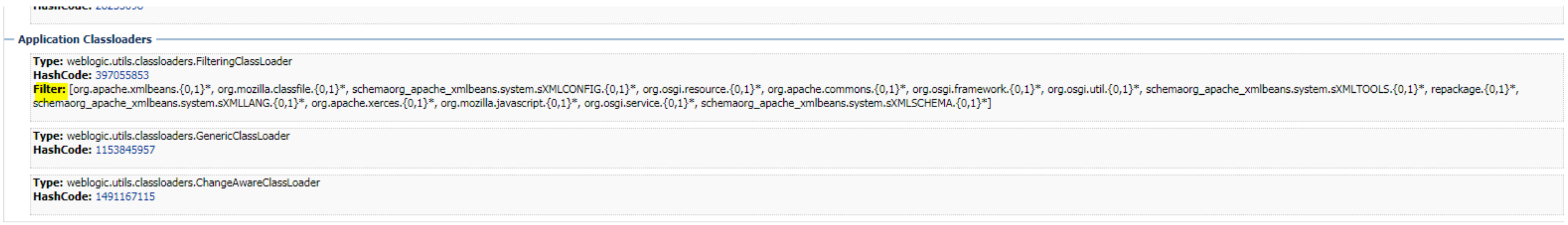
Using CAT to see the conflicts, the Weblogic.xml was modified and now it looks like this:



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE weblogic-web-app
  PUBLIC "-//BEA Systems, Inc.//DTD Web Application 8.1//EN"
  "http://www.bea.com/servers/wls810/dtd/weblogic810-web-jar.dtd" >

<weblogic-web-app>
    <container-descriptor>
        <prefer-web-inf-classes>false</prefer-web-inf-classes>
        <prefer-application-packages>
           <package-name>org.apache.xerces.*</package-name>
           <package-name>org.apache.commons.*</package-name>
           <package-name>org.apache.xmlbeans.*</package-name>
           <package-name>org.mozilla.classfile.*</package-name>
           <package-name>org.mozilla.javascript.*</package-name>
           <package-name>org.osgi.framework.*</package-name>
           <package-name>org.osgi.resource.*</package-name>
           <package-name>org.osgi.service.*</package-name>
           <package-name>org.osgi.util.*</package-name>
           <package-name>repackage.*</package-name>
           <package-name>schemaorg_apache_xmlbeans.system.sXMLCONFIG.*</package-name>
           <package-name>schemaorg_apache_xmlbeans.system.sXMLLANG.*</package-name>
           <package-name>schemaorg_apache_xmlbeans.system.sXMLSCHEMA.*</package-name>
           <package-name>schemaorg_apache_xmlbeans.system.sXMLTOOLS.*</package-name>
        </prefer-application-packages>
    </container-descriptor>
</weblogic-web-app>
```

The previous filter can be seen using CAT



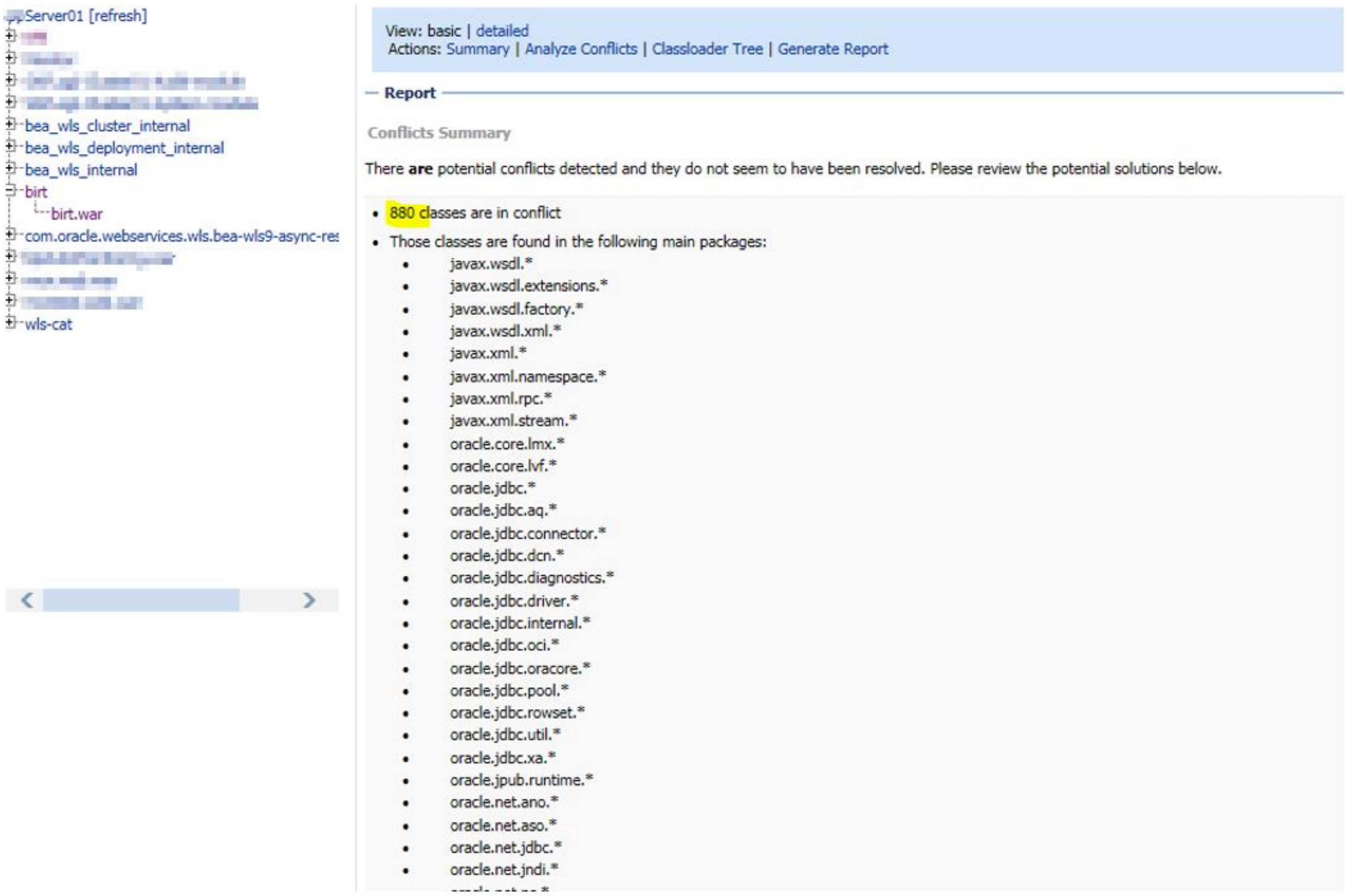
Application Classloaders

Type: weblogic.utils.classloaders.FilteringClassLoader
HashCode: 397055853
Filter: [org.apache.xmlbeans.{0,1}*, org.mozilla.classfile.{0,1}*, schemaorg_apache_xmlbeans.system.sXMLCONFIG.{0,1}*, org.osgi.resource.{0,1}*, org.apache.commons.{0,1}*, org.osgi.framework.{0,1}*, org.osgi.util.{0,1}*, schemaorg_apache_xmlbeans.system.sXMLTOOLS.{0,1}*, repackage.{0,1}*, schemaorg_apache_xmlbeans.system.sXMLLANG.{0,1}*, org.apache.xerces.{0,1}*, org.mozilla.javascript.{0,1}*, org.osgi.service.{0,1}*, schemaorg_apache_xmlbeans.system.sXMLSCHEMA.{0,1}*]

Type: weblogic.utils.classloaders.GenericClassLoader
HashCode: 1153845957

Type: weblogic.utils.classloaders.ChangeAwareClassLoader
HashCode: 1491167115

Moreover, the number of conflicts was reduced


Server01 [refresh]
- bea_wls_cluster_internal
- bea_wls_deployment_internal
- bea_wls_internal
- birt
  - birt.war
- com.oracle.webservices.wls.bea-wls9-async-re:
- wls-cat

View: basic | detailed
Actions: Summary | Analyze Conflicts | Classloader Tree | Generate Report

Report

Conflicts Summary

There **are** potential conflicts detected and they do not seem to have been resolved. Please review the potential solutions below.

- 880 classes are in conflict
- Those classes are found in the following main packages:
  - javax.wsdl.*
  - javax.wsdl.extensions.*
  - javax.wsdl.factory.*
  - javax.wsdl.xml.*
  - javax.xml.*
  - javax.xml.namespace.*
  - javax.xml.rpc.*
  - javax.xml.stream.*
  - oracle.core.lmx.*
  - oracle.core.lvf.*
  - oracle.jdbc.*
  - oracle.jdbc.aq.*
  - oracle.jdbc.connector.*
  - oracle.jdbc.dcn.*
  - oracle.jdbc.diagnostics.*
  - oracle.jdbc.driver.*
  - oracle.jdbc.internal.*
  - oracle.jdbc.oci.*
  - oracle.jdbc.oracore.*
  - oracle.jdbc.pool.*
  - oracle.jdbc.rowset.*
  - oracle.jdbc.util.*
  - oracle.jdbc.xa.*
  - oracle.jpub.runtime.*
  - oracle.net.ano.*
  - oracle.net.aso.*
  - oracle.net.jdbc.*
  - oracle.net.jndi.*

Therefore, in cases like this using a filter within the Weblogic.xml file is better than using the tag **<prefer-web-inf-classes>true</prefer-web-inf-classes>**

Conclusion

At least there are two ways to solve this kind of conflicts deleting JAR files that are not used by the application or filtering classes through the Weblogic.xml file where a filter is recommended as can be seen in the second case described in this document and in the bug described by Oracle in [4]

References list

[1] Oracle (2015) Using the Classloader Analysis Tool (CAT) [Online document] Available from: https://docs.oracle.com/middleware/1213/wls/WLPRG/classloading.htm (Accessed on: 23/01/2018)

[2] Oracle (n.d.) Java Virtual Machine Specification [Online document] Available from: https://docs.oracle.com/javase/specs/jvms/se7/html/jvms-5.html (Accessed on: 25/01/2018)

[3] Oracle (2017) JDK1.8 ClassLoader Doesn't Load META-INF/services/* From Libraries In WEB-INF/lib With prefer-web-inf-classes=true (Doc ID 2229218.1) [Online document] Available from: https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=524837554942480&id=2229218.1&_adf.ctrl-state=ab6oaidok_126 (Accessed on: 25/01/2018)

[4] Oracle (2017) Log4j Initialization Error in WebServices Deploy With prefer-web-inf-classes=true (Doc ID 2266334.1) [Online document] Available from: https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=525271175379472&id=2266334.1&displayIndex=1&_afrWindowMode=0&_adf.ctrl-state=ab6oaidok_228#SYMPTOM (Accessed on: 25/01/2018)