

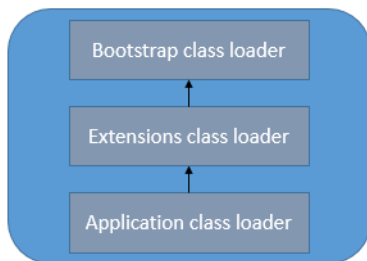
CHAPTER 11: CLASS LOADERS

Theory

With the help of the class loaders, WebSphere Application Server enables applications to access available class and resources. Having known the structure and the hierarchy of class loaders will help you to understand and troubleshoot advances environments against errors such as “ClassNotFoundException”.

Each Java class must be loaded by a class loader that locates and loads the required classes. During the startup of a JVM, following class loaders are used:

- The *bootstrap class loader*, loads the core Java libraries that are located under “<JAVA_HOME>/jre/lib”. This class loader is written in native code.
- The *extensions class loader*, loads the code in the extensions directories such as “<JAVA_HOME>/jre/lib/ext” or any other directory that is specified by the “java.ext.dirs” system property. This class loader is implemented by the “sun.misc.Launcher\$ExtClassLoader” class.
- The *application class loader*, loads the code defined in the “java.class.path” that maps to the “CLASSPATH” environment variable. This is implemented by the “sun.misc.Launcher\$AppClassLoader” class.

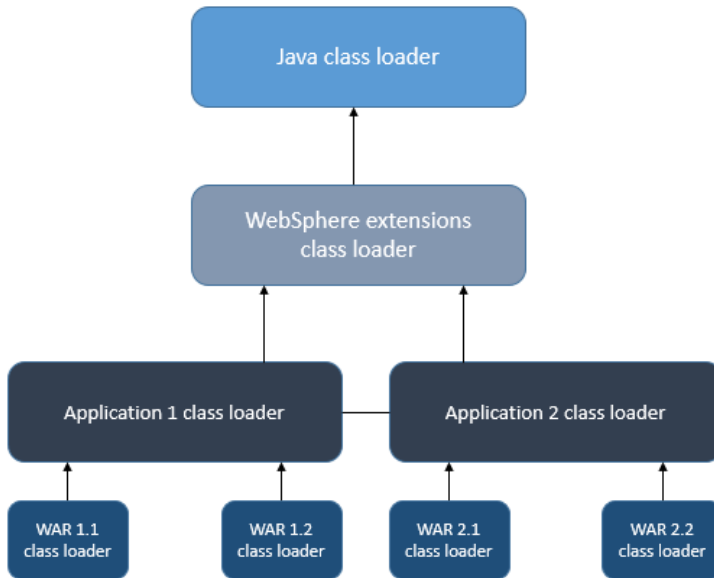


Java class loaders hierarchy

The *WebSphere extensions class loader* is where WebSphere Application Server itself is loaded. The class path that is used by the extensions class loader is retrieved from the “ws.ext.dirs” system property, which is initially derived from the “WAS_EXT_DIRS” environment variable set in the “setupCmdLine” script file. Each directory listed in the ws.ext.dirs environment variable is added to the WebSphere extensions class loaders class path, and every .jar file and .zip file in the directory is added to the class path.

Application or web module class loaders load elements of enterprise applications running on the server. These application elements can be:

- Web modules
- EJB modules
- Application client modules
- Resource adapter archives
- Utility JAR files



WebSphere class loaders hierarchy

Each class loader is a child of the previous class loader. WebSphere Application Server provides the ability to change the class loading behavior.

- **Application Server class loaders:** You can configure 2 types of class loader policies,
 - Single: In this case, applications are not isolated and uses a single application loader to load all of the modules, libraries and JAR files that are required by all the applications installed into the JVM.
 - Multiple: All applications are isolated and each of them has its own class loader to load EJB modules, libraries and etc.
 You can also configure the class loading mode as **parent first** that sets the loading of the classes to its parent class loader before attempting to load classes from its local path, or as **parent last** that starts loading classes from its local class path before asking its parent.

- **Application class loaders:** You can set the class loader order and WAR class loader policy within WebSphere Application Server.

For class loader order you have 2 options:

- *Classes loaded with parent class loader first*, sets the loading of classes to its parent class loader before attempting to load the class from its local class path.
- *Classes loaded with application class loader first*, tells the class loader to start with loading classes from its local class path before asking its parent.

For WAR class order policy, you have the following options:

- *Class loader for each WAR file in application* that creates a separate class loader is assigned to each WAR file.
- *Single class loader for application* single class loader is assigned to all WAR files.

- **Web module class loaders:** You can set the class loader order that specifies the search order of the class whether to look first in the parent class loader or in the application class loader. You can set the order as following:
 - *Classes loaded with parent class loader first*, the class loader searches first the application class loader
 - *Classes loaded with application class loader first*, the class loader searches first in the WAR class loaders to load a class.

AIM

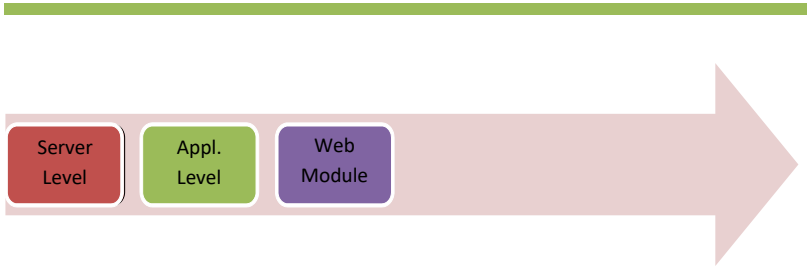
The aim of the lab exercise is to get familiar with class loader settings in different levels. In order to achieve this goal, you need to perform following tasks:

Task 1: Configure class loaders on server level

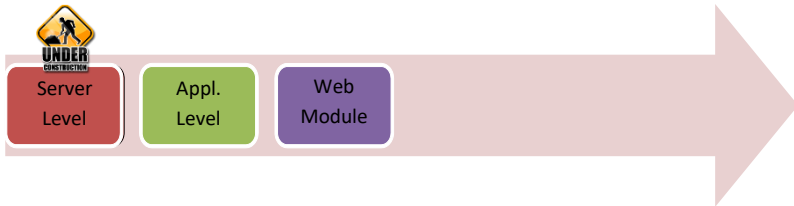
Task 2: Configure class loaders on application level

Task 3: Configure web module class loader

Lab Exercise 11: CLASS LOADERS



- 1. Configure class loaders on server level**
- 2. Configure class loaders on application level**
- 3. Configure web module class loader**



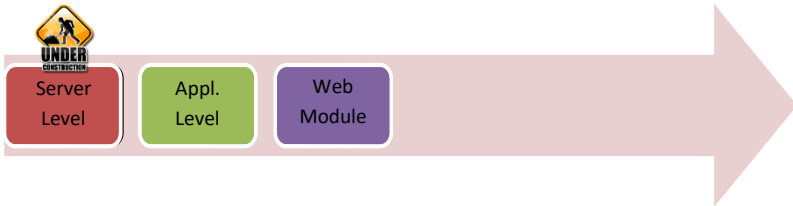
Task 1: Configure class loaders on server level

Step 1: Navigate to “Servers>Server Types>WebSphere application servers” and then click on the server name, “server1”.

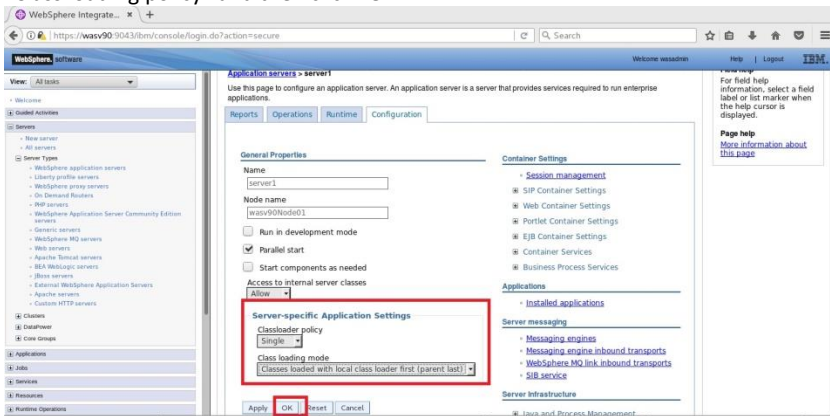
The screenshot shows the WebSphere Integrated Solutions console. The left sidebar displays the navigation tree, with 'WebSphere application servers' selected. The main area displays the 'Application servers' page, which includes a table of application servers. The table has columns for Name, Node, Host Name, Version, Cluster Name, and Status. The 'server1' row is highlighted.

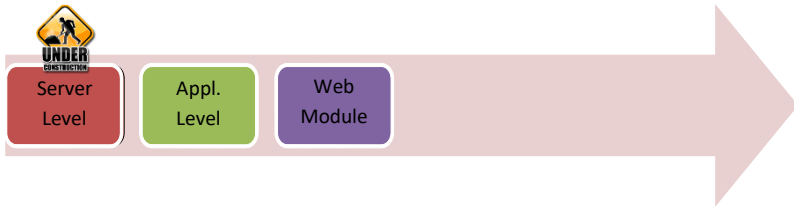
Select	Name	Node	Host Name	Version	Cluster Name	Status
<input type="checkbox"/>	App_Server01	wasv0Node01	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	App_Server02	wasv0Node02	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input checked="" type="checkbox"/>	server1	wasv0Node01	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	server2	wasv0Node02	wasv90	ND 9.0.0.0	WAS_CLUSTER	→

Total 4



Step 2: Under the “Server-specific Application Settings”, select “Single” as “Classloader policy” and “Classes loaded with local class loader first (parent last)” as “Class loading policy” and then click “OK”.





Step 3: Click “Save” to write changes directly.

The screenshot shows the WebSphere Integrated Solutions console. A red box highlights a 'Messages' dialog box with the following text:

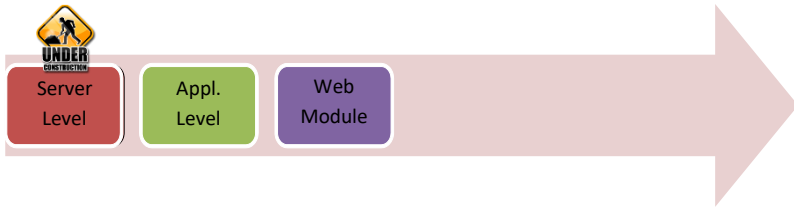
- Changes have been made to your local configuration. You can:
 - Save directly to the master configuration.
 - Revert changes before saving or discarding.
- An option to synchronize the configuration across multiple nodes after saving can be enabled in Preferences.
- The server may need to be restarted for these changes to take effect.

Below the dialog box, the 'Application servers' section is visible, showing a table of servers:

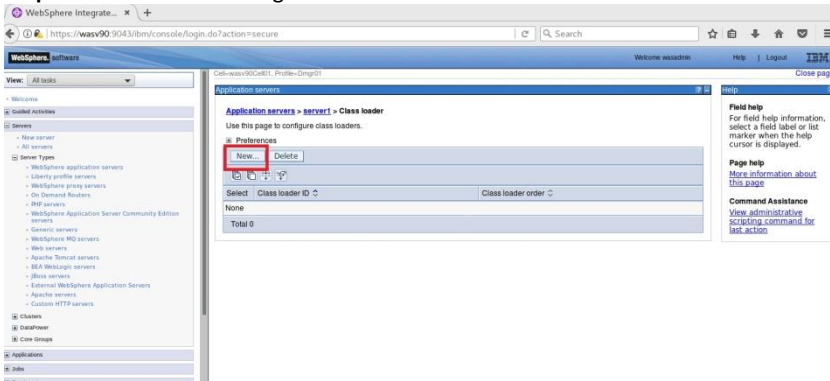
Select	Name	Node	Host Name	Version	Cluster Name	Status
<input type="checkbox"/>	App_Server01	wasv90Node01	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	App_Server02	wasv90Node02	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	server1	wasv90Node01	wasv90	ND 9.0.0.0		→
<input type="checkbox"/>	server2	wasv90Node02	wasv90	ND 9.0.0.0		→

Step 4: Navigate to “Server Infrastructure>Java and Process Management>Class loader”.

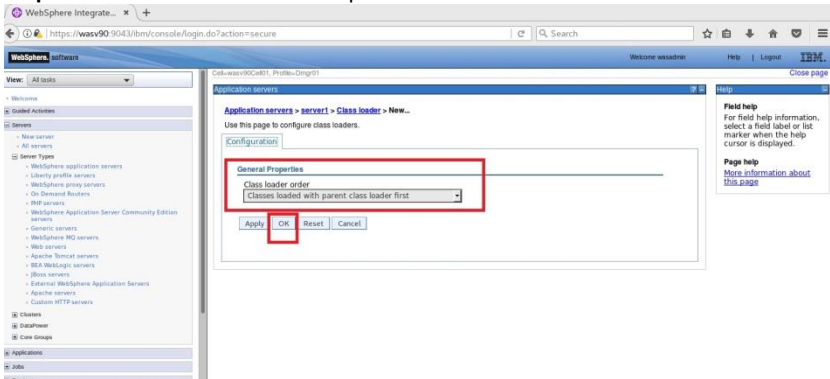
The screenshot shows the WebSphere Integrated Solutions console with the 'Class loader' configuration page selected. The 'Server Infrastructure' section is expanded, and 'Class loader' is highlighted. The 'Class loader policy' is set to 'Single', and the 'Class loading mode' is set to 'Classes loaded with local class loader first (parent last)'. The 'Server-specific Application Settings' section is also visible.

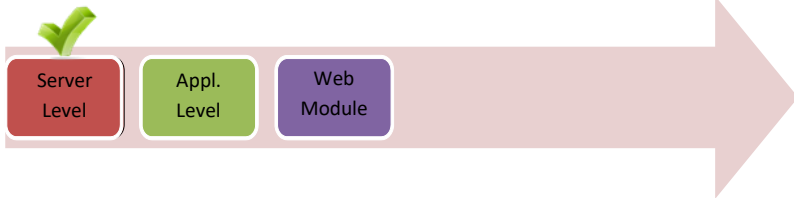


Step 5: Click “New” to configure new class loader.



Step 6: Select “Classes loaded with parent class loader first” then click “OK”.





Step 7: Click “Save” to write changes to the master file.

The image displays two screenshots of the WebSphere Integrated Solutions console, specifically the 'Application servers > server1 > Class loader' configuration page. The top screenshot shows a message box indicating that changes have been made to the local configuration and can be saved directly to the master configuration. The bottom screenshot shows the 'Class loader' configuration page with a red box highlighting the 'Class loader ID' field, which is set to 'Classloader_1526800014027'.

WebSphere Integrated Solutions Console - Application servers > server1 > Class loader

Use this page to configure class loaders.

Preferences

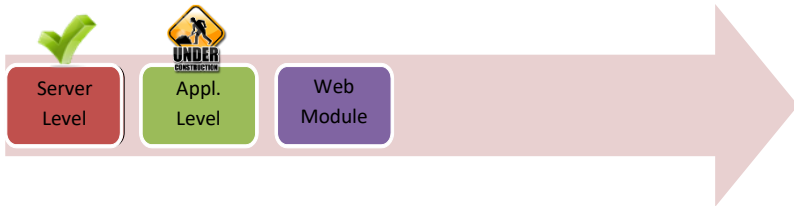
Select Class loader ID: Class loader order:

You can administer the following resources:

Class loader ID	Class loader order	Classes loaded with parent class loader first
<input type="checkbox"/> Classloader_1526800014027	1	<input type="checkbox"/>

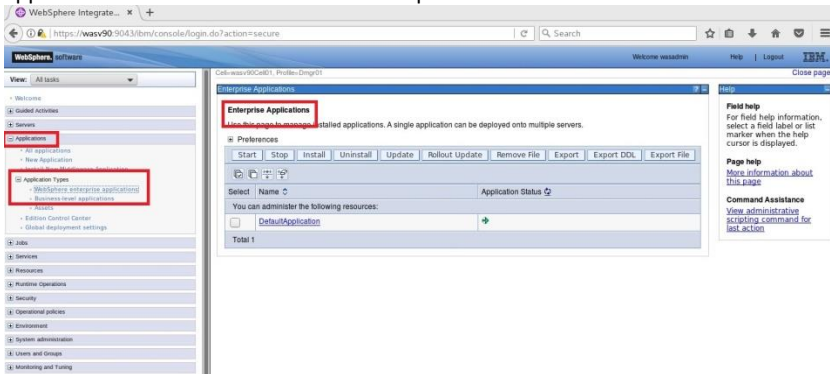
Total: 1

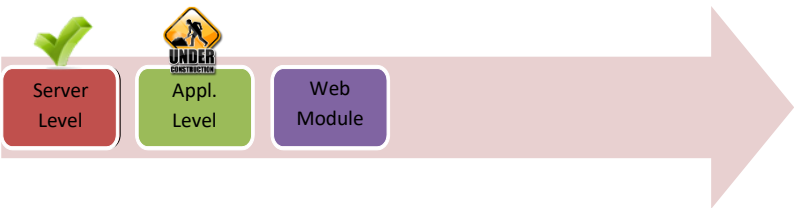
Task 1 is complete!



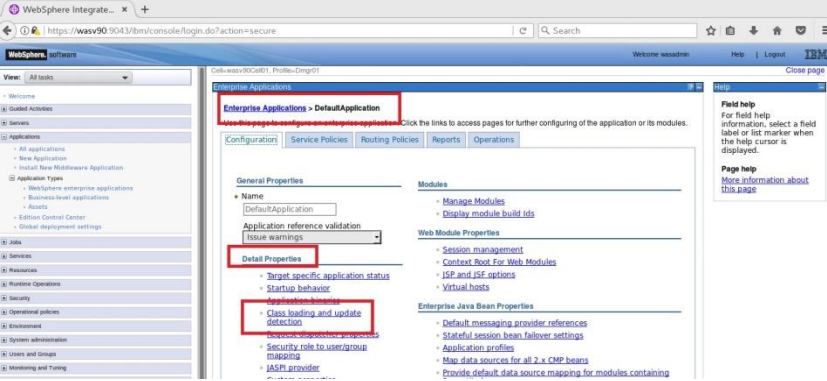
Task 2: Configure class loaders on application level

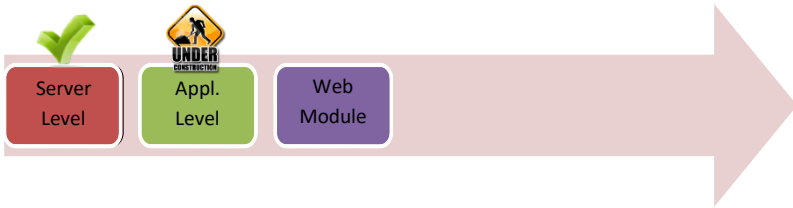
Step 1: Navigate to “Applications>Application Types>WebSphere enterprise applications” and then click on “DefaultApplication”.



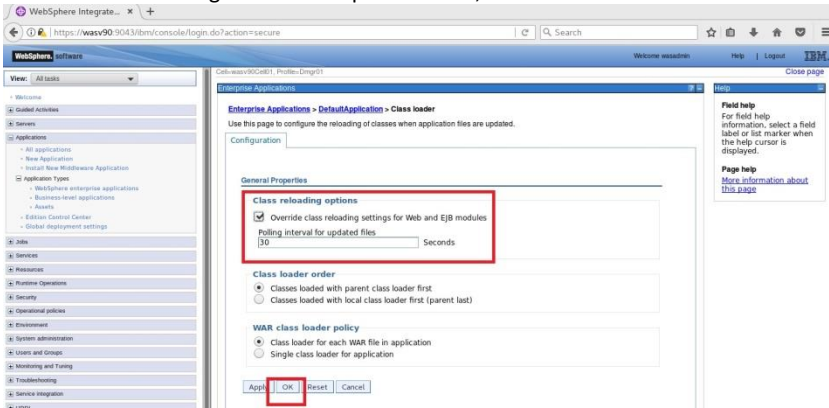


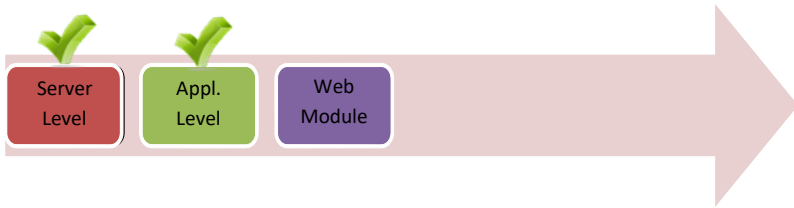
Step 2: Under “Default Properties”, click on “Class loading and update detection”.



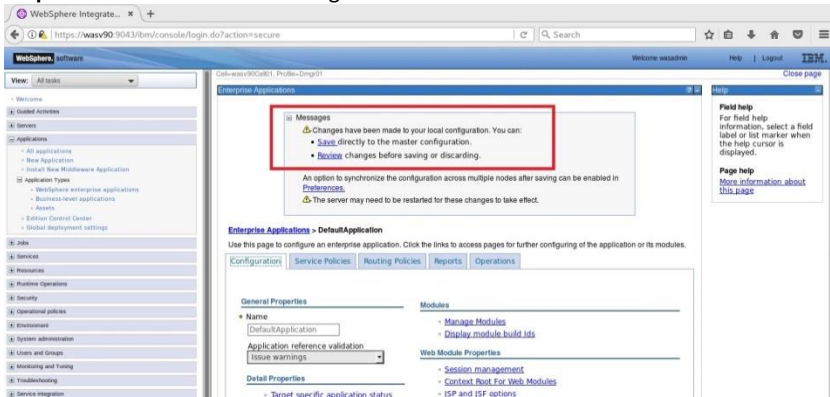


Step 3: Select “Override class reloading settings for the Web and EJB module” and enter “30” for “Polling interval for updated files”, then click “OK”.

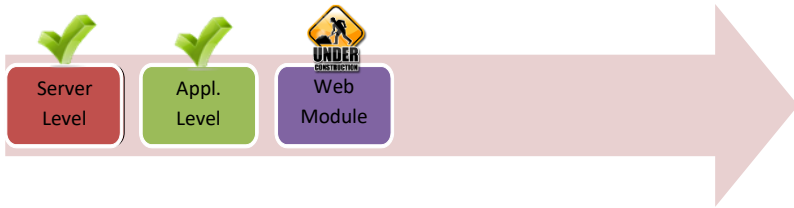




Step 4: Click “Save” to write changes to master file.

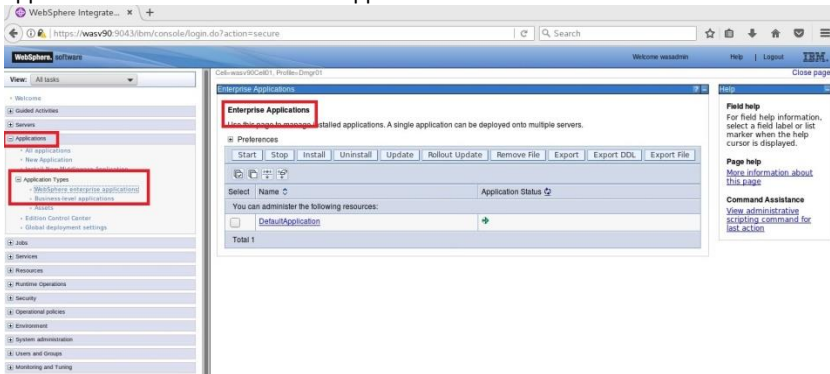


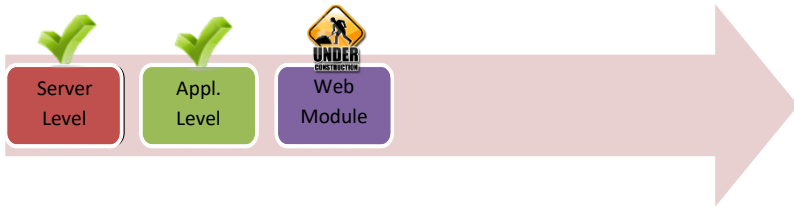
Task 2 is complete!



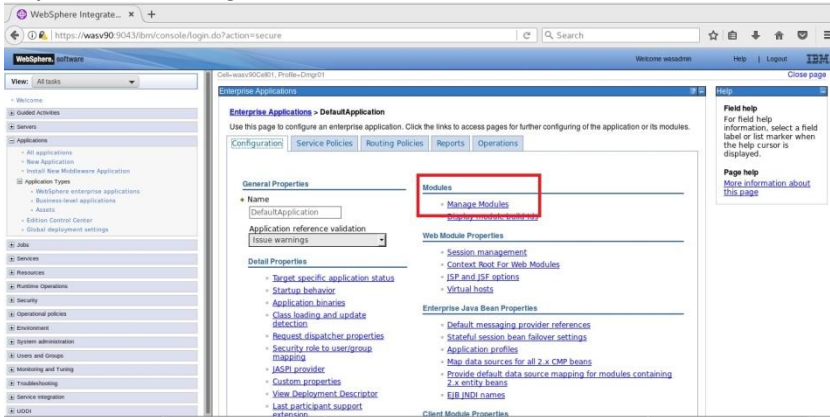
Task 2: Configure web module class loader

Step 1: Navigate to “Applications>Application Types>WebSphere enterprise applications” and click on “DefaultApplication”.

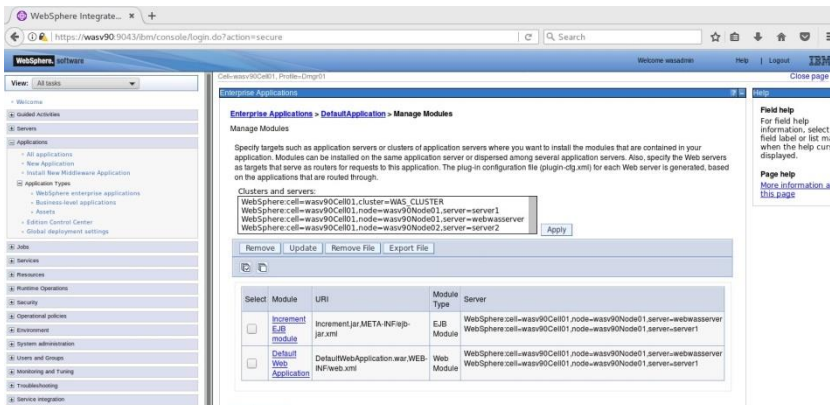


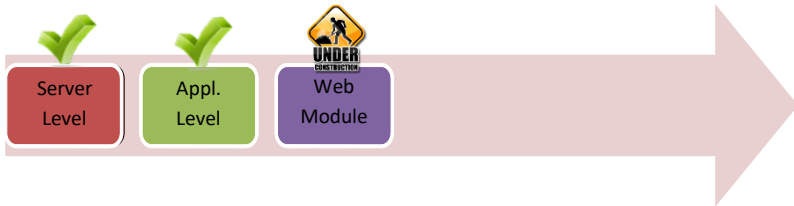


Step 2: Click on “Manage Modules” under “Modules”.

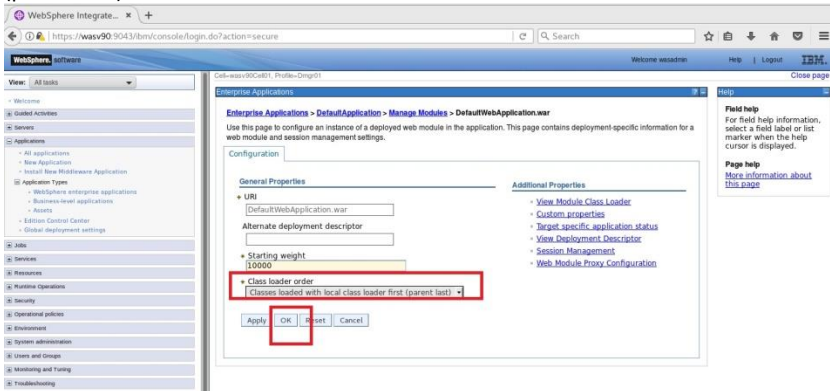


Step 3: Click on the web module name, “Default Web Application”.



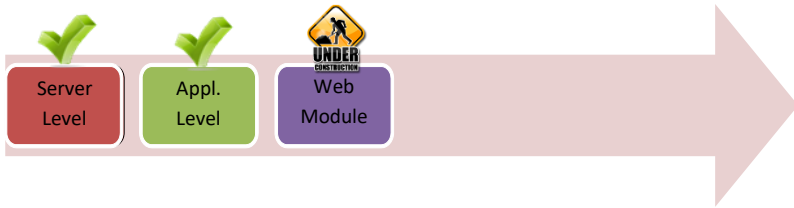


Step 4: Change the class loader order, “Classes loaded with local class loader first (parent last)”.



Step 5: Click “Save” to write changes.





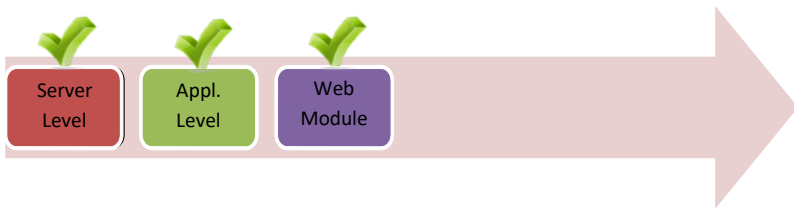
Step 6: Click on “View Module Class Loader” under “Additional Properties”.

The screenshot shows the WebSphere Integrated Solutions console. The breadcrumb trail is 'Enterprise Applications > DefaultApplication > Manage Modules > DefaultWebApplication.war'. The 'Configuration' tab is active. Under the 'Additional Properties' section, the 'View Module Class Loader' link is highlighted with a red rectangle. Other links in this section include 'Custom properties', 'Target specific application status', 'View Deployment Descriptor', 'Session Management', and 'Web Module Proxy Configuration'. The left sidebar shows the navigation tree with 'Applications' selected.

Step 7: Click on the server name, “server1”.

The screenshot shows the WebSphere Integrated Solutions console. The breadcrumb trail is 'Enterprise Applications > DefaultApplication > Manage Modules > DefaultWebApplication.war > View Class Loader (Select Server)'. The 'Preferences' tab is active. A table lists the servers available for selection. The 'server1' entry is highlighted with a red rectangle. The table has columns for 'Server Name' and 'Node name'. The 'Total 1' row is also visible.

Server Name	Node name
server1	wasv90Node01
Total 1	



Step 8: You should see the class loaders loaded files of a module to debug problems with class loaders.

WebSphere Integrated Solutions console

Enterprise Applications > DefaultApplication > Manage Modules > DefaultWebApplication.war > View Class Loader (Select Server) >

Class loader viewer

Use this page to examine the class loaders visible to a Web module (.war file) or enterprise bean (.ejb file) in an installed enterprise application. This page helps you to determine which class loaders loaded files of a module and to diagnose problems with class loaders.

Hierarchy Search Order

Export Table View Search

ClassLoader - Search Order

- 1. J2K Extension - sun.misc.Launcher\$ExtClassLoader
- 2. J2K Application - sun.misc.Launcher\$AppClassLoader
- 3. OSGi - org.eclipse.osgi.internal.bundleloader.DefaultClassLoader
- 4. Extension - com.ibm.wa.bootstrap.ExtClassLoader
- 5. WebSphere Application Server Protected Class Loader - com.ibm.ws.classloader.ProtectedClassLoader
- 6. Module - com.ibm.ws.classloader.CompoundClassLoader
- 7. Module - com.ibm.ws.classloader.CompoundClassLoader

Task 3 is complete!

SUMMARY

WebSphere Application Server enables applications to access available class and resources using class loaders. Understand the class loaders will help you to identify and solve the issues in an advanced WebSphere environment. WebSphere Application Server enables you to configure class loaders on server level, application level and web module level.

REFERENCES

- http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Fcrun_classload.html
- http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Furun_rclassloader_inst.html
- <https://publib.boulder.ibm.com/infocenter/iseres/v5r4/index.jsp?topic=%2Frzamy%2F50%2Fprogram%2Fclsadmcons.htm>

INDEX

application class loader.....	364
Application class loaders.....	366
Application Server class loaders	365
bootstrap class loader.....	364
class loader	364
extensions class loader	364
Web module class loaders	366

