# Lab 2 [Upgrade Java]

**Lab 2 covers:**

- Use Case 4 – **Upgrade Java**

Use Case 4 Upgrade Java [From Java 8 to Java 21]

This scenarios follows the previous scenario: Modernize to Liberty. It requires that the application is already modernized to Liberty. It begins with the code in the state that we finished the Modernize to Liberty scenario

1. Right click on the project, select **watsonx Code Assistant for Enterprise Java Applications,** then select **Upgrade Java version**.
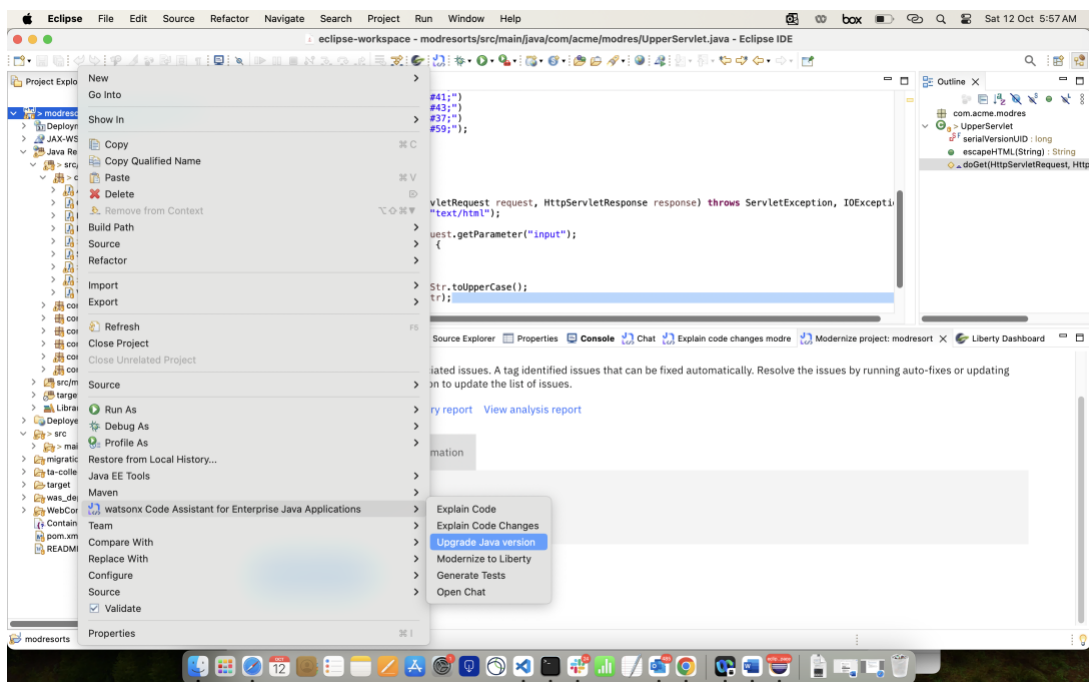
You should see the following panel appear:
(Note: Click Start over in the newly opened tab to see this panel)
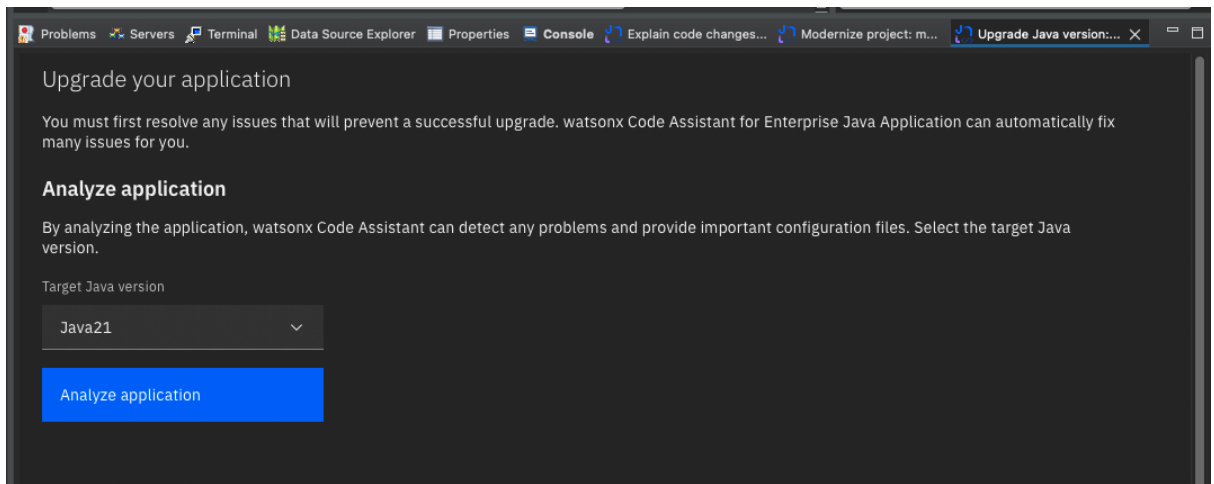
2. Before you Analyze the application, the application needs to be built. If it's not built at this point build it now (command line: **mvn clean package**, or via the IDE UI if you have that configured). See also the previous note on Java versions in the building section. *(Note: Only perform this step if the application is not already 'build and refresh' in the previous lab)*

3. Ensure **Java21** is selected and Click the **Analyze application** button. After a few moments you should see the issues. There is one issue with an auto fix, and one issue with a LLM assisted fix:
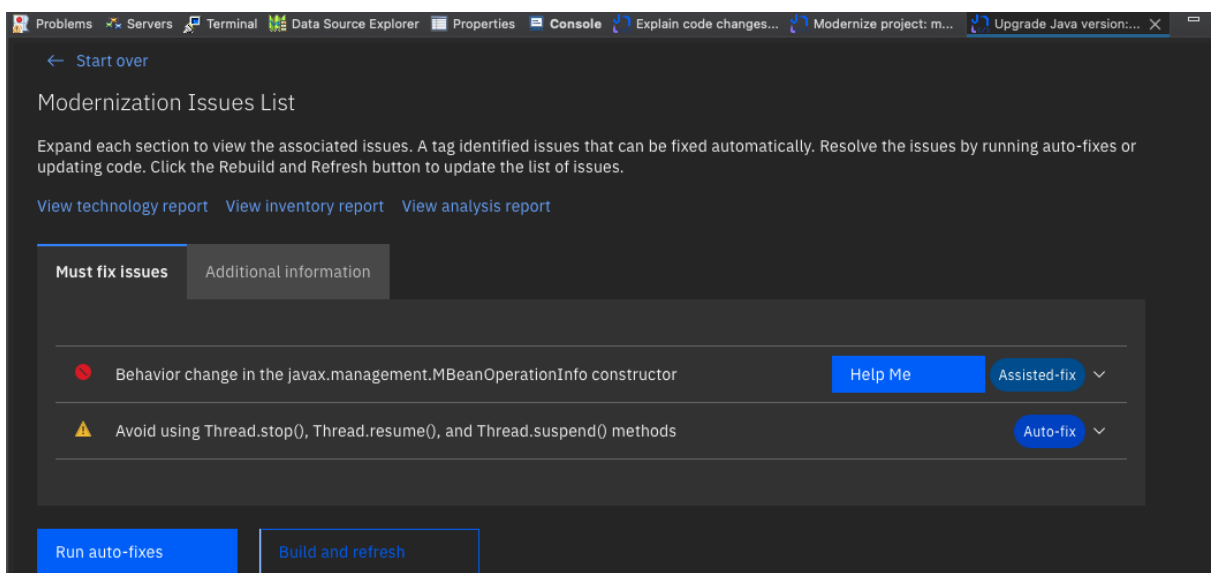
4. At this point we can run ModResorts in a Java21 environment to observe how the application is not functioning correctly **BEFORE** we fix the Java upgrade issues.

    - Start the application using **Liberty tools**. Make sure its running with Java 21 but clicking **Start...** on the Liberty tools dashboard, and then selecting Java 21 in the JRE tab:
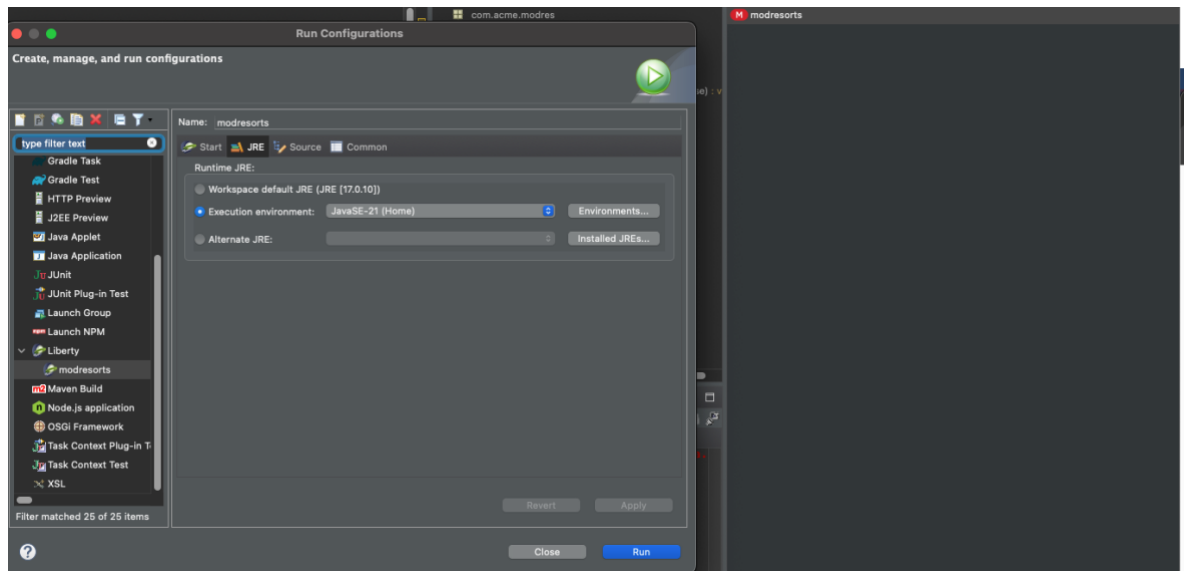
    - When the application is running, go to:
      http://localhost:9080/resorts

      and click the Where to drop-down. Select any value. You should observer ERRORs in the UI:
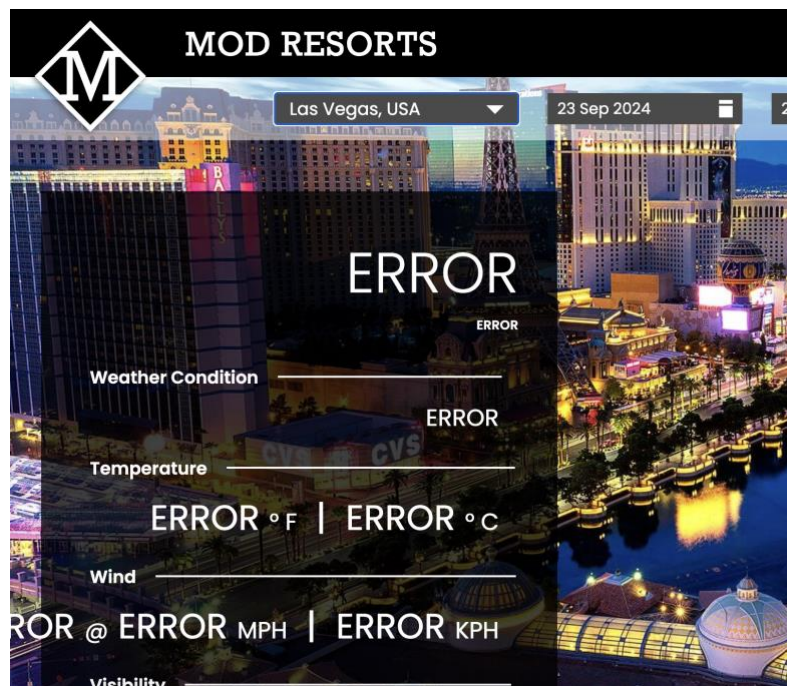
- These errors are ultimately the result of the Java upgrade issue with the MBeanOperatorInfo constructor which causes a server error when fetching data for the UI.

5. Back in Eclipse, in Upgrade Java tab. Click **Build and Refresh** then the **Run auto-fixes** button.
6. Now we will resolve the other issue (**Behavior change in the javax.management.MBeanOperationInfo constructor)** with the help of the watsonx code assistant.
7. Due to limitation in private preview release, it is not easy to know which piece of code the issue relates to. To see this, you need to click on the **View analysis report** link. In the report, find the issue and click on **Show result**s to reveal the file, method and line for the issue.

Critical Rules

Java SE version migration / Java SE 11 compatibility impacts

🚫 Behavior change in the javax.management.MBeanOperationInfo constructor (1)   Show rule help   Close results

Results

FILE NAME	REFERENCE DETAILS	MATCH CRITERIA	LINE NUMBER
modresorts-2.0.0.war			
WEB-INF/classes/com/acme/modres/mbean/DMBeanUtils.class	Method getOps	javax.management.MBeanOperationInfo(java.lang.String, java.lang.String, javax.management.MBeanParameterInfo[], java.lang.String, int, javax.management.Descriptor)	28

8. Open the **DMBeanUtils** class in the IDE, as shown in the above path.
9. Select the **getOps** method beginning with the "public static" and finishing with the enclosing "}" and click **Help me** button:
(code snippet as shown below)

```java
package com.acme.modres.mbean;

import java.util.logging.Level;

public final class DMBeanUtils {
    private static final Logger logger = Logger.getLogger(DMBeanUtils.class.getName());

    public static MBeanOperationInfo[] getOps(OpMetadataList opList) {
        MBeanOperationInfo[] ops = null;
        if (opList == null || opList.getOpMetadatList() == null) {
            logger.log(Level.WARNING, "No operation is configured");
            return ops;
        }

        int numOps = opList.getOpMetadatList().size();
        if (numOps > 0) {
            ops = new MBeanOperationInfo[numOps];
            int i = 0;
            for (OpMetadata opMetadata : opList.getOpMetadatList()) {
                String name = opMetadata.getName();
                String desc = opMetadata.getDescription();
                String type = opMetadata.getType();
                int impact = opMetadata.getImpact();

                MBeanOperationInfo opInfo = new MBeanOperationInfo(name, desc, /* signature
                ops[i++] = opInfo;
            }
        }

        return ops;
    }
}
```

Modernization Issues List

Expand each section to view the associated issues. A tag identified issues that can be fixed automatically. Resolve the issues by running auto-fixes or updating code. Click the Rebuild and Refresh button to update the list of issues.

View technology report   View inventory report   View analysis report

| Must fix issues | Additional information |

🚫 Behavior change in the javax.management.MBeanOperationInfo constructor   Help Me   Assisted-fix ⌄

10. The code snippet should consist of the full **getOps** method, with the offending lines fixed. Replace your **getOps** method with the code returned from the LLM. *(new code snippet as shown below)*

```
public final class DMBeanUtils {
    private static final Logger logger = Logger.getLogger(DMBeanUtils.class.getName());

    public static MBeanOperationInfo[] getOps(OpMetadataList opList) {
        MBeanOperationInfo[] ops = null;
        if (opList == null || opList.getOpMetadatList() == null) {
            logger.log(Level.WARNING, "No operation is configured");
            return ops;
        }

        int numOps = opList.getOpMetadatList().size();
        if (numOps > 0) {
            ops = new MBeanOperationInfo[numOps];
            int i = 0;
            for (OpMetadata opMetadata : opList.getOpMetadatList()) {
                String name = opMetadata.getName();
                String desc = opMetadata.getDescription();
                String type = opMetadata.getType();
                int impact = opMetadata.getImpact();

                try {
                    MBeanOperationInfo opInfo = new MBeanOperationInfo(name, desc, /* signature */ null, type,
impact, /* descriptor */ null);
                    ops[i++] = opInfo;
                } catch (IllegalArgumentException e) {
                    logger.log(Level.WARNING, "MBeanOperationInfo.UNKNOWN is used as the value for the impact
field");
                    impact = MBeanOperationInfo.UNKNOWN;
                    MBeanOperationInfo opInfo = new MBeanOperationInfo(name, desc, /* signature */ null, type,
impact, /* descriptor */ null);
                    ops[i++] = opInfo;
                }
```
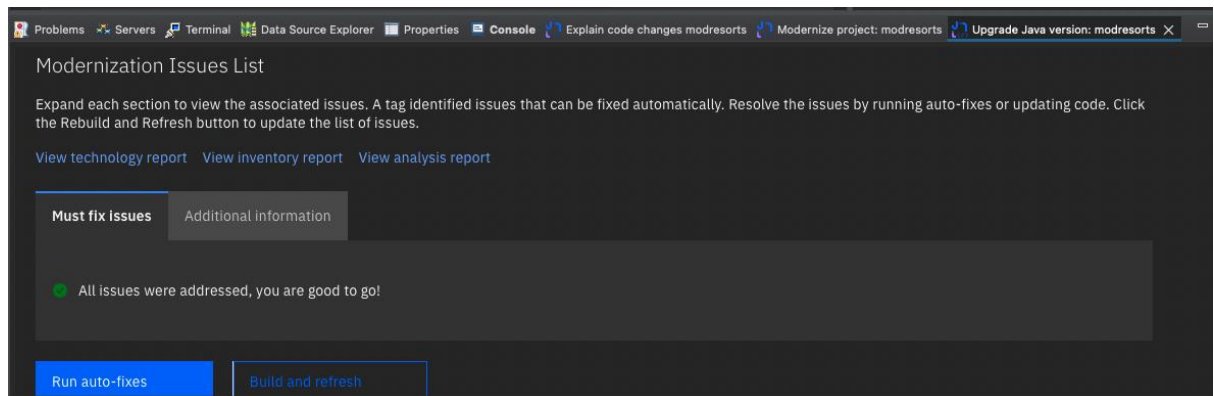
11. After saving your work, go back to the **Upgrade Java** panel, and click **Build and refresh**. After a couple of moments, you should observe all the issues are fixed.

12. Return to http://localhost:9080/resorts

and click the **Where to** down. Select any value. You should observe that the errors have disappeared.

ModResorts is now a **Liberty application** and is upgraded to **Java21**!