MIGRATING CONTROLS CODE FROM JAVA 8 TO JAVA 11

Mira Welner, Lyle Beaulac, Mikhail Fedorov

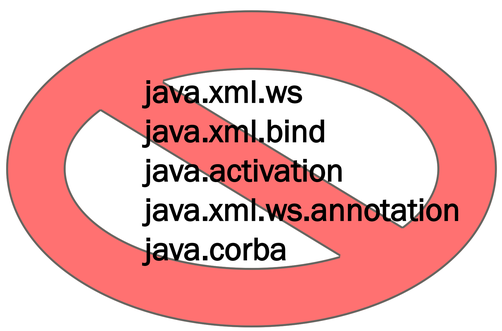
The National Ignition Facility

The National Ignition Facility (NIF) houses the most energetic laser in the world. That laser is essential for researching clean energy and the nature of stars, as well as maintaining the nuclear stockpile. The laser is controlled by the Integrated Computer Controls System, (ICCS), a complex software system that integrates a multitude of devices and controllers. It is essential to keep the ICCS up to date, and so this summer the NIF computing division began the work of updating the controls code from Java 8 to Java 11.

NIF

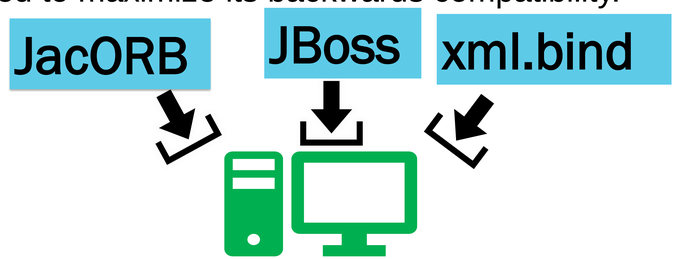
The Java 11 Update

Code written for Java 8 will not always compile under Java 11. Java 11 has many distinctions from Java 8. Starting at Java 9, many features in the Java Standard Library were deprecated, meaning that they would be removed at sometime in the future. In the Java 11, update, they were removed from Java Standard. Specifically, JDK Enhancement Proposal 320[1] removed the Java EE and the CORBA modules, both modules which were being used in ICCS. This resulted in the code for the ICCS being unable to run in Java 11.



Updating the Code

Since the libraries we used were no longer part of the JSE, we imported several libraries from their official Maven repositories, including JBoss, to provide support for CORBA, as well as java.xml.bind, JacORB, and many more. Some of the libraries directly replaced the previous library. However, some were a bit different and the code had to be slightly altered to make it behave the way it was supposed to. When we altered the code, we ensured that it was backwards compatible and could be run in Java 8 as well. There were some locations were that was not possible, but we tried to maximize its backwards compatibility.

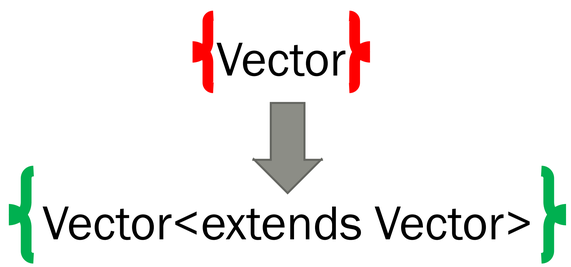


The changes in the libraries were quite well documented by Oracle. However, there were some smaller alterations that were less well documented.
For example, the DefaultTableModel class had different signatures for its constructor parameters. Originally it was either

(vector, int) or **(vector, vector)**[2]

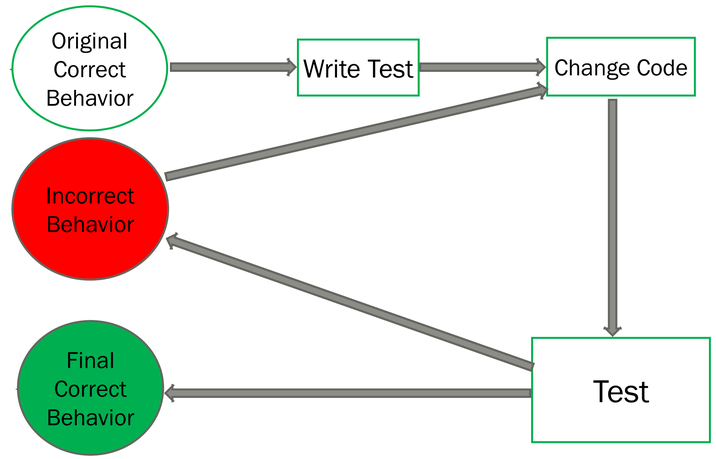
but in Java 9 and later it became

(vector<?>,int) or **(vector<?extends Vector>,Vector<?>)**[3]



Testing the Code

When code is altered, there is always the risk that the behavior of the code was changed along with the syntax. So, we wrote low- level unit tests to ensure that the code performed the same as the original Java 8 code. The test would be written to pass with the original Java 8 code. Then it would be run on the Java 11 code, and the Java 11 code would be altered to pass the test if necessary.



Future Goals

To the extent of our knowledge, the code currently runs well on Java 11. In the future however we will be performing integration tests as well as manual GUI testing to ensure that the code performs the way we are hoping for it to perform.

References

[1] EP 320: Remove the Java EE and CORBA Modules.” *Open JDK*, Oracle “http://openjdk.java.net/jeps/320”
[2] Java API Reference.” *DefaultTableModel,* Oracle, 7 Sept. 2018
[3] Java API Reference.” *DefaultTableModel (Java Platform SE 8),* Oracle, 6 Mar. 2019

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

LLNL-POST-783278