Resolving Eclipse JDK Issues

When you start Eclipse 4.6.3 by clicking `eclipse.exe` and encounter the error “Java was started but returned exit code”with a reference to `Oracle\Java\javapath\javaw.exe`, it means Eclipse is attempting to use a Java Virtual Machine (JVM) that is either incompatible or misconfigured. The path `Oracle\Java\javapath\javaw.exe` indicates that Eclipse is defaulting to a JVM defined in your system’s PATH environment variable, which may not be suitable for Eclipse 4.6.3. To resolve this, you need to specify the correct Java home for Eclipse by configuring it to use a compatible JVM. Here’s how to do it:

Step-by-Step Solution

1. **Understand the Requirement**
   Eclipse 4.6.3 (Neon) requires at least Java 8 to run properly. The error suggests that the current JVM (from `Oracle\Java\javapath`) may not meet this requirement or has configuration issues. You’ll need to point Eclipse to a compatible Java installation, such as a Java 8 JDK.

2. **Locate Your Java Installation**
   Identify where a compatible Java version (e.g., JDK 1.8.0) is installed on your system. Common locations on Windows include:

   - `C:\Program Files\Java\jdk1.8.0_XXX` (for 64-bit Java)

   - `C:\Program Files (x86)\Java\jdk1.8.0_XXX` (for 32-bit Java)
     Replace `XXX` with the specific update version (e.g., `231` for JDK 1.8.0_231). Inside this directory, the `javaw.exe` file is located in the `bin` subdirectory (e.g., `C:\Program Files\Java\jdk1.8.0_XXX\bin\javaw.exe`).

   Tip: To confirm the version and architecture, open a command prompt, navigate to the `bin` directory (e.g., `cd C:\Program Files\Java\jdk1.8.0_XXX\bin`), and run:

   ```
   java -version
   ```

   Look for “64-Bit”or “32-Bit”in the output to verify the architecture. Ensure it matches your Eclipse version (likely 64-bit if downloaded recently).

3. **Find the `eclipse.ini` File**
   The `eclipse.ini` file is a configuration file located in the same directory as `eclipse.exe`. For example, if Eclipse is installed in `C:\eclipse`, the file will be at `C:\eclipse\eclipse.ini`. This file allows you to specify the JVM that Eclipse should use.

4. **Edit the `eclipse.ini` File**
   Open `eclipse.ini` in a text editor (e.g., Notepad) with administrative privileges. You’ll modify it to include the `-vm` argument, which tells Eclipse which JVM to use. Follow these steps:

  - **Check the Existing Content**: Look for a `-vm` argument. If it's already present, it will be followed by a path on the next line (e.g., `-vm` followed by `C:/some/path/bin/javaw.exe`). If it points to the problematic `Oracle\Java\javapath\javaw.exe`, you'll replace it. If no `-vm` argument exists, you'll add it.
  - **Add or Modify the `-vm` Argument**: Insert the following two lines before the `-vmargs` section (if it exists) or near the top of the file after initial startup parameters:

    ```
    -vm
    C:/Program Files/Java/jdk1.8.0_XXX/bin/javaw.exe
    ```

    - Use forward slashes (/) instead of backslashes (\) to avoid parsing issues.
    - Replace `C:/Program Files/Java/jdk1.8.0_XXX` with the actual path to your Java installation.
  - **Ensure Proper Placement**: The `-vm` argument must appear before the `-vmargs` section, which typically starts with `-vmargs` followed by JVM options like `-Xms256m` or `-Xmx1024m`. For example, your `eclipse.ini` might look like this after editing:

    ```
    -startup
    plugins/org.eclipse.equinox.launcher_1.3.201.v20161025-1711.jar
    --launcher.library
    plugins/org.eclipse.equinox.launcher.win32.win32.x86_64_1.1.401.v20161122-1740
    -vm
    C:/Program Files/Java/jdk1.8.0_XXX/bin/javaw.exe
    -vmargs
    -Dosgi.requiredJavaVersion=1.8
    -Xms256m
    -Xmx1024m
    ```

  - **Avoid Extra Spaces or Blank Lines**: Ensure there are no unnecessary spaces or blank lines immediately after the `-vm` or path lines, as this can cause Eclipse to misinterpret the configuration.

5. **Save and Test**
   Save the `eclipse.ini` file, then double-click `eclipse.exe` to start Eclipse. If configured correctly, Eclipse should launch without the "Java was started but returned exit code"error.

Additional Notes

- **Architecture Match**: Ensure the JVM architecture (32-bit or 64-bit) matches your Eclipse installation. A 64-bit Eclipse requires a 64-bit JVM, typically found in `C:\Program Files\Java`, while a 32-bit Eclipse needs a 32-bit JVM from `C:\Program Files (x86)\Java`.
- **Verify the Path**: Double-check that `javaw.exe` exists at the specified location. If the path is incorrect, Eclipse will fail to start.

- **Fallback Option**: If this doesn't work, consider setting the `JAVA_HOME` environment variable (e.g., `C:\Program Files\Java\jdk1.8.0_XXX`) and updating the system PATH to include `%JAVA_HOME%\bin`. However, modifying `eclipse.ini` is the preferred and more reliable method for Eclipse.

By editing the `eclipse.ini` file to include the `-vm` argument with the path to a compatible `javaw.exe`, you effectively specify the Java home for Eclipse, ensuring it uses the correct JVM to start.