Eclipse JDK-Probleme beheben

Wenn Sie Eclipse 4.6.3 starten, indem Sie `eclipse.exe` anklicken, und den Fehler "Java wurde gestartet, kehrte aber mit dem Exit-Code"mit einem Verweis auf `Oracle\Java\javapath\javaw.exe` erhalten, bedeutet dies, dass Eclipse versucht, eine Java Virtual Machine (JVM) zu verwenden, die entweder inkompatibel oder falsch konfiguriert ist. Der Pfad `Oracle\Java\javapath\javaw.exe` deutet darauf hin, dass Eclipse standardmäßig auf eine JVM zugreift, die in der PATH-Umgebungsvariable Ihres Systems definiert ist, was möglicherweise nicht für Eclipse 4.6.3 geeignet ist. Um dies zu beheben, müssen Sie den richtigen Java-Home-Pfad für Eclipse angeben, indem Sie ihn so konfigurieren, dass er eine kompatible JVM verwendet. Hier ist, wie Sie dies tun können:

Schritt-für-Schritt-Lösung

1. **Verstehen Sie die Anforderungen** Eclipse 4.6.3 (Neon) benötigt mindestens Java 8, um ordnungsgemäß zu funktionieren. Der Fehler deutet darauf hin, dass die aktuelle JVM (aus `Oracle\Java\javapath`) möglicherweise diese Anforderungen nicht erfüllt oder Konfigurationsprobleme hat. Sie müssen Eclipse auf eine kompatible Java-Installation verweisen, wie z.B. eine Java 8 JDK.

2. **Ihre Java-Installation finden** Identifizieren Sie, wo eine kompatible Java-Version (z.B. JDK 1.8.0) auf Ihrem System installiert ist. Gängige Orte auf Windows sind:

   - `C:\Program Files\Java\jdk1.8.0_XXX` (für 64-Bit Java)
   - `C:\Program Files (x86)\Java\jdk1.8.0_XXX` (für 32-Bit Java) Ersetzen Sie `XXX` durch die spezifische Aktualisierungsversion (z.B. `231` für JDK 1.8.0_231). In diesem Verzeichnis befindet sich die Datei `javaw.exe` im Unterverzeichnis `bin` (z.B. `C:\Program Files\Java\jdk1.8.0_XXX\bin\javaw.exe`).

   Tipp: Um die Version und Architektur zu bestätigen, öffnen Sie eine Eingabeaufforderung, navigieren Sie zum `bin`-Verzeichnis (z.B. `cd C:\Program Files\Java\jdk1.8.0_XXX\bin`) und führen Sie aus:

   ```
   java -version
   ```

   Achten Sie auf "64-Bit"oder "32-Bit"in der Ausgabe, um die Architektur zu überprüfen. Stellen Sie sicher, dass sie mit Ihrer Eclipse-Version übereinstimmt (wahrscheinlich 64-Bit, wenn Sie sie kürzlich heruntergeladen haben).

3. **Die `eclipse.ini`-Datei finden** Die `eclipse.ini`-Datei ist eine Konfigurationsdatei, die sich im selben Verzeichnis wie `eclipse.exe` befindet. Zum Beispiel, wenn Eclipse in `C:\eclipse` installiert ist, befindet sich die Datei unter `C:\eclipse\eclipse.ini`. Diese Datei ermöglicht es Ihnen, die JVM anzugeben, die Eclipse verwenden soll.

4. **Die `eclipse.ini`-Datei bearbeiten** Öffnen Sie `eclipse.ini` in einem Texteditor (z.B. Notepad) mit Administratorrechten. Sie werden es ändern, um den `-vm`-Argument hinzuzufügen, der Eclipse mitteilt, welche JVM verwendet werden soll. Folgen Sie diesen Schritten:

- **Überprüfen Sie den bestehenden Inhalt**: Suchen Sie nach einem `-vm`-Argument. Wenn es bereits vorhanden ist, wird es von einem Pfad in der nächsten Zeile gefolgt (z.B. `-vm` gefolgt von `C:/some/path/bin/javaw.exe`). Wenn es auf die problematische `Oracle\Java\javapath\javaw.exe` verweist, ersetzen Sie es. Wenn kein `-vm`-Argument vorhanden ist, fügen Sie es hinzu.
- **Fügen Sie das `-vm`-Argument hinzu oder ändern Sie es**: Fügen Sie die folgenden beiden Zeilen vor dem `-vmargs`-Abschnitt (falls vorhanden) oder nahe der Spitze der Datei nach den anfänglichen Startparametern ein:

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

  - Verwenden Sie Schrägstriche (`/`) anstelle von Backslashes (`\`), um Parsing-Probleme zu vermeiden.
  - Ersetzen Sie `C:/Program Files/Java/jdk1.8.0_XXX` durch den tatsächlichen Pfad zu Ihrer Java-Installation.
- **Stellen Sie die richtige Platzierung sicher**: Das `-vm`-Argument muss vor dem `-vmargs`-Abschnitt erscheinen, der normalerweise mit `-vmargs` beginnt, gefolgt von JVM-Optionen wie `-Xms256m` oder `-Xmx1024m`. Zum Beispiel könnte Ihre `eclipse.ini` nach dem Bearbeiten so aussehen:

```
-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
```

- **Vermieden Sie zusätzliche Leerzeichen oder leere Zeilen**: Stellen Sie sicher, dass keine unnötigen Leerzeichen oder leeren Zeilen unmittelbar nach den `-vm`- oder Pfadzeilen vorhanden sind, da dies Eclipse dazu veranlassen kann, die Konfiguration falsch zu interpretieren.

5. **Speichern und Testen** Speichern Sie die `eclipse.ini`-Datei und doppelklicken Sie auf `eclipse.exe`, um Eclipse zu starten. Wenn korrekt konfiguriert, sollte Eclipse ohne den Fehler “Java wurde gestartet, kehrte aber mit dem Exit-Code”starten.

Zusätzliche Hinweise

- **Architekturübereinstimmung**: Stellen Sie sicher, dass die JVM-Architektur (32-Bit oder 64-Bit) mit Ihrer Eclipse-Installation übereinstimmt. Eine 64-Bit Eclipse benötigt eine 64-Bit JVM, die normalerweise

in `C:\Program Files\Java` zu finden ist, während eine 32-Bit Eclipse eine 32-Bit JVM aus `C:\Program Files (x86)\Java` benötigt.

- **Überprüfen Sie den Pfad**: Stellen Sie sicher, dass `javaw.exe` an der angegebenen Stelle vorhanden ist. Wenn der Pfad falsch ist, wird Eclipse nicht starten.
- **Fallback-Option**: Wenn dies nicht funktioniert, überlegen Sie, die `JAVA_HOME`-Umgebungsvariable (z.B. `C:\Program Files\Java\jdk1.8.0_XXX`) festzulegen und den System-Pfad so zu aktualisieren, dass er `%JAVA_HOME%\bin` enthält. Das Bearbeiten von `eclipse.ini` ist jedoch die bevorzugte und zuverlässigere Methode für Eclipse.

Durch das Bearbeiten der `eclipse.ini`-Datei, um das `-vm`-Argument mit dem Pfad zu einer kompatiblen `javaw.exe` hinzuzufügen, geben Sie effektiv den Java-Home-Pfad für Eclipse an, sodass er die richtige JVM zum Starten verwendet.