Java Auto Modules Aufwerten

common-cli und andere mit jlink packen

Onkobu Tanaake

2021-10-03T10:00:00

Java 9 hat Projekt JigSaw mitgebracht. Das gesamte JDK ist nun kein großer Klotz mehr. Stattdessen besteht es aus Modulen. Und weil Java WebStart entfernt wurde, braucht es ein neues Verteilformat. Davor musste man immer ein komplettes JRE herunterladen und installieren.

Ein neues Werkzeug names `jlink` wurde ergänzt. In Verbindung mit Moduldeklarationen kann der Entwickler alle notwendigen Module inklusive der des JDKs in ein einziges ZIP- Archiv packen. Das schließt ein Skript zum Starten ein. Wenn das dann heruntergeladen und entpackt wird, startet das Programm direkt.

Einige der von mir genutzten Bibliothken sind etwas antiquiert und werden wahrscheinlich niemals ein ordentliches Java-Modul. Aber ich möchte sie dennoch als Teil meiner Programme mit verpacken. Der Aufruf von `jlink` steigt sonst mit einer Fehlermeldung aus, Auto-Module könnten nicht analysiert werden.

```
Error: jdk.tools.jlink.plugin.PluginException: module-info.class
```

Die neuen Werkzeuge litten und leiden an lang bekannten Problemen. Ich nenne gern JDK-8229396 [https://bugs.openjdk.java.net/browse/JDK-8229396] und die Geschichte des kaputten `jdeps`. Schließlich kämen wir zu JDK-8130047 [https://bugs.openjdk.java.net/browse/JDK-8130047] oder warum `jlink` wahrscheinlich nie mehr korrigiert wird. Deswegen fanden clevere Köpfe einen Weg, die JARs neu zu verpacken und sie zu spezifischen Modulen zu machen:

1. Automatisch modules-info.java mit `jdeps` erzeugen
2. Die Datei mit `javac` kompilieren
3. Das JAR mit der kompilierten Modul-Definition aktualisieren
4. Eine Maven-Signatur aktualisieren (signierte Module können nicht korrigiert werden)

Eines meiner Soliton-Skripte erledigt das alles für dich: Codeberg.org [https://codeberg.org/onkobu/soliton/src/branch/master/bin/to-explicit.sh].