JavaFX: Installation

Eine kleine Übersicht über die notwendigen Abhängigkeiten von JavaFX – das sich zum Glück rein auf die Abhängigkeit zu Java reduzieren lässt, so dass es in diesem Blog nach minimaler Einstufung von JavaFX in erster Linie um die Installation der „richtigen“ Java Version geht.

## Teile der JavaFX Serie (aktualisiert)

* [JavaFX 01: Übersicht / Planung](https://blog.kneitzel.de/2020/06/29/javafx-uebersicht-planung/) ([YouTube](https://youtu.be/emCUpGFpf_Q))
* [JavaFX 02: Installation](https://blog.kneitzel.de/2020/06/24/javafx-uebersicht/) ([YouTube](https://youtu.be/ZmDUr7fwABM))
* [JavaFX 03: Maven & Gradle](https://blog.kneitzel.de/2020/11/02/javafx-maven-und-gradle/) ([YouTube](https://youtu.be/daVCphI0qZ0))
* [JavaFX 04: IntelliJ](https://blog.kneitzel.de/2020/11/02/javafx-intellij/) (YouTube)

### Teile der JavaFX Serie (alt / geplant)

* [JavaFX 05: Scene Builder](https://blog.kneitzel.de/2020/06/26/javafx-scene-builder/)
* [JavaFX --: Erläuterung der Applikation](https://blog.kneitzel.de/2020/06/26/javafx-erlaeuterungen-der-applikation/)
* [JavaFX 06: Model / View / Controller](https://blog.kneitzel.de/2020/06/26/javafx-model-view-controller/)
* [JavaFX 07: MVVM](https://blog.kneitzel.de/2020/06/29/javafx-mvvm/)
* JavaFX: Komplette MVVM Applikation (geplant)
* JavaFX: Layouts (geplant)
* JavaFX: Formulare einfach erstellen (geplant)
* JavaFX: Multi Plattform (geplant)

## Code der JavaFX Serie

Der Source Code sowie alle Dokumente finden sich auf GitHub: <https://github.com/kneitzel/blog-javafx-series>

## Übersicht über JavaFX

JavaFX ist ein modernes Framework zur Erstellung von graphischen Oberflächen in Java. JavaFX ist eine eigenständige Komponente, die oft nicht im Java Development Kit integriert ist. Bei Oracle / OpenJDK war JavaFX in den Downloads ab Java SE 7 Update 2 bis einschließlich Version 10. Ab Version 11 war dies nicht mehr dabei. Es gibt aber teilweise Anbieter, die auch aktuelle OpenJDK Versionen mit integriertem JavaFX anbieten.

## Unterschied zu Swing

Ein wichtiger Unterschied zu Swing ist, dass eine Oberfläche in einem separaten File beschrieben werden kann, statt diese in Java selbst zusammen zu stellen (deklarative Oberfläche im Gegensatz zu der programmierten Oberfläche).

Des Weiteren bietet JavaFX einige technische Aspekte wie z.B. das Binding von Controls an Modelle.

## Installation

Da es nicht mehr fester Bestandteil von Java ist, sind manuelle Aktivitäten notwendig. Dazu gibt es zwei Möglichkeiten:

### Eigenständige Installation

Es zwei Möglichkeiten für eine Installation. Zum einen kann auf ein JDK zurückgegriffen werden, in dem das JavaFX noch eingebunden ist. Dies ist sehr einfach, aber alle Entwickler am Projekt müssen genau solch eine Installation wählen.

Oder das JavaFX wird eigenständig installiert. Damit dies aber gefunden wird, muss der Ort konfiguriert werden, indem ein Module Path gesetzt wird. Dies erfordert Angaben bei in der Entwicklungsumgebung oder auf der Kommandozeile, die dann auch jeder Entwickler machen muss.

Wenn dieser Weg gewählt wird, dann kann auch ein JDK mit eingebautem JavaFX benutzt werden. Dann ist keine komplexe Konfiguration notwendig, aber die Entwickler sind auf eines der eher raren Angebote angewiesen. Ein solches JDK findet sich u.a. auf <https://bell-sw.com/>.

### Gradle / Maven Build Tool

JavaFX kann als Abhängigkeit in den Build Tools Gradle oder Maven eingetragen werden. Neben den Abhängigkeiten sind auch noch ein paar weitere Einträge notwendig, aber diese sind für alle Entwickler an einem Projekt gleich. Die Entwickler müssen auf ihren Computern keine weiteren Dinge installieren.

Dies ist die Methode, welche ich im Rahmen der JavaFX Serie nutzen werde und euch im Detail vorstellen werde. Dabei werde ich mich vor allem auf Gradle konzentrieren, aber ich werde auch ein Maven Projekt kurz vorstellen.

|  |
| --- |
| **Hinweis** |
| Bei den Build Tools ist in der Regel keine systemweite Installation notwendig. Statt dessen wird ein sogenannter Wrapper im Projekt integriert, der die korrekte Version des Build Tools im Projektverzeichnis zur Verfügung stellt. |

## Java Version

Bei der Auswahl der richtigen Java Installation sind ein paar Punkte zu beachten.

### JRE vs. JDK

Generell wird bei Java zwischen dem Runtime Environment (JRE) und dem Development Kit (JDK) unterschieden.

Um Java Programme auszuführen wird nur ein JRE benötigt. Das JDK enthält neben einem JRE noch alle Komponenten, die der Entwicklung von Java Programmen dienen, z.B. den Java Compiler.

|  |
| --- |
| **Wichtig** |
| Wir benötigen somit auf jeden Fall ein JDK. |
| **Achtung** |
| Oracle bietet auf [http://java.com](http://java.com/) ein altes JRE an. Dieses sollte man nicht installieren! Es wird nicht benötigt und neigt eher dazu, Probleme zu bereiten! |

### Java vs. OpenJDK

Es gibt das Produkt Java von Oracle, welches unter einer speziellen Lizenz von Oracle steht und frei heruntergeladen werden kann. Des Weiteren gibt es das OpenJDK, welches die open source Variante ist und einem vollwertigen Java entspricht und das auch von vielen Firmen aktiv weiterentwickelt wird. (z.B. IBM, Azul, …)

|  |
| --- |
| **Tipp** |
| Die OpenJDK Lizenz gibt einem deutlich mehr Freiheiten und birgt bei der Nutzung deutlich geringere Risiken einer Lizenzverletzung. Oracle Java lässt sich meist frei nutzen, aber es ist die Lizenz zu prüfen um sicher zu gehen, dass die Nutzung abgedeckt ist und keine kommerzielle Lizenz erworben werden muss! |

### LTS vs. aktueller Version

Java wird aktiv weiterentwickelt und es kommen regelmäßig neue Versionen. Aktuell sind wir zum Zeitpunkt dieser Bearbeitung bei Java 15. Sobald eine neue Version herausgegeben wurde, wird die alte Version in der Regel nicht mehr aktualisiert. Für Security Updates ist es dann notwendig, dass die neue Version installiert wird – was leider auch zu Kompatibilitätsproblemen führen kann.

Diese Problematik wird gelöst, indem sogenannte Long Time Support (LTS) Version bereitgestellt werden. Diese Versionen werden deutlich länger mit Updates versorgt und verringern die Wahrscheinlichkeit, dass es bei dem Installieren eines neueren Builds der LTS Version zu Problem kommt. Derzeit werden die LTS Version Java 8 und Java 11 bereitgestellt.

### Diverse JVM

Es gibt teilweise unterschiedliche Java Virtual Machines (JVM). Die original JVM ist die HotSpot JVM von Oracle. Die Eclipse Foundation hat die OpenJ9 JVM entwickelt, die mehr Wert auf schnelle Startzeit und geringen Speicherverbrauch legt. Bei AdoptOpenJDK kann zwischen den beiden Varianten gewählt werden. Da beide universell einsetzbar sein, ist dies eine Wahl, bei der man nicht viel falsch machen kann.

|  |
| --- |
| **Empfehlung** |
| Ich rate zu der Version 11 eines OpenJDK. Unter Windows / macOS kann auf ein JDK von [https://adoptopenjdk.net](https://adoptopenjdk.net/) zurück gegriffen werden. Unter Linux wird einfach das OpenJDK aus dem Repository verwendet, z.B. mittels sudo apt-get install openjdk-11-jdk |

## Links

* JavaFX / openjfx Homepage: https://openjfx.io/
* Java OpenJDK AdoptOpenJDK
* Entwicklungsumgebungen: JetBrains IntelliJ Community, Eclipse, Netbeans
* Repository mit Code für diese JavaFX Blog Serie: https://github.com/kneitzel/blog-javafx-series

## Änderungsnachweis

* 2020-10-29 Komplette Überarbeitung
* 2020-07-03 Wechsel zu einem Repository für die ganze Serie.