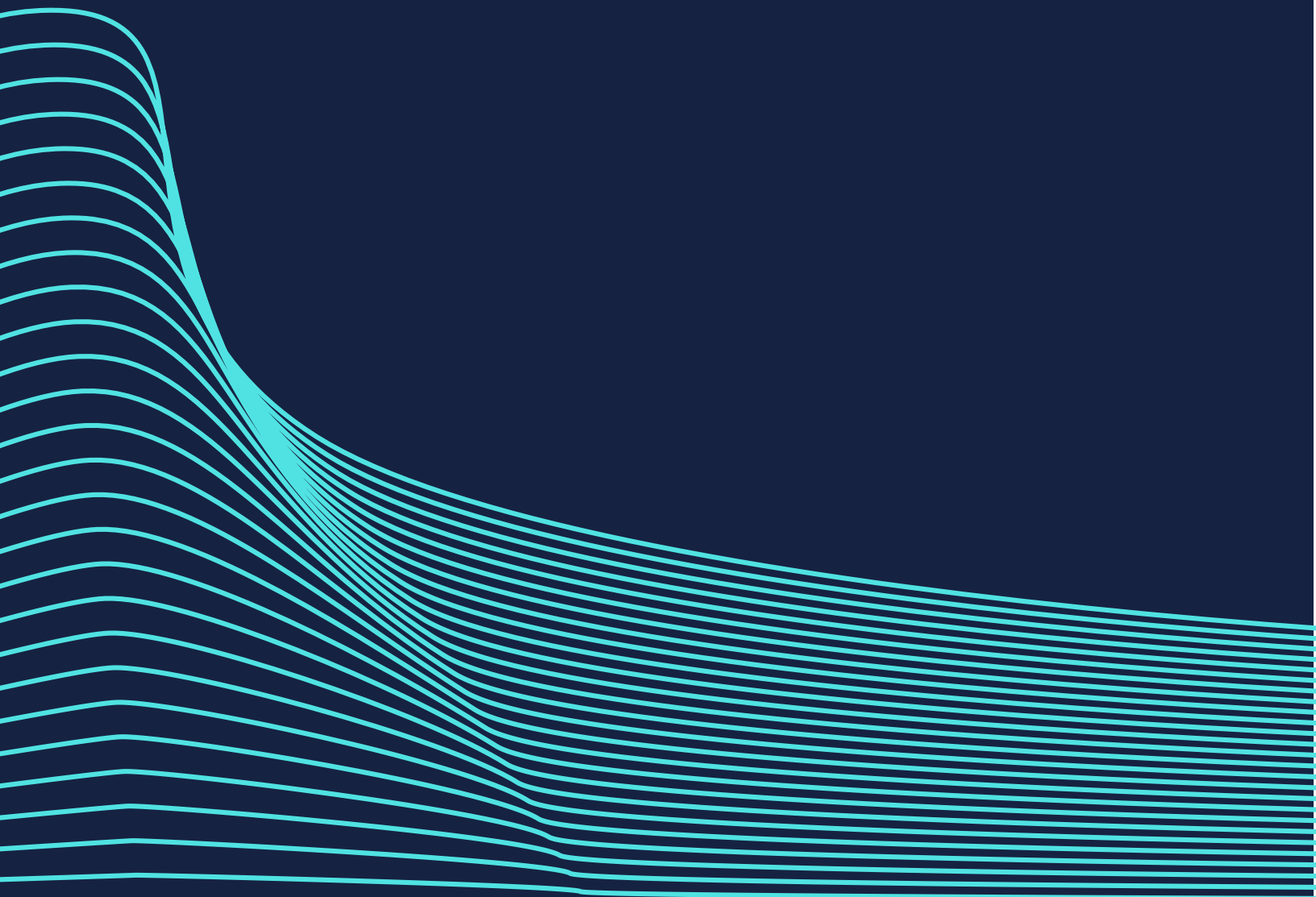
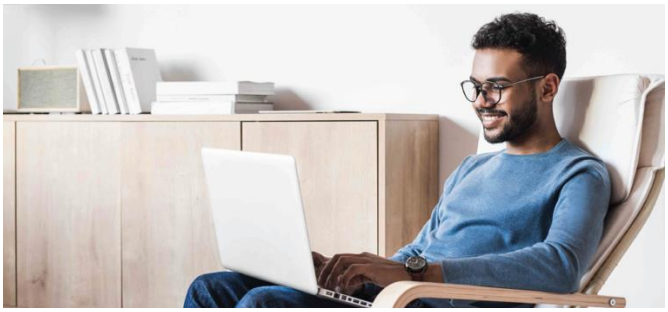
LOGICALIS
Architects of Change

azul

Übergang von Oracle JDK zu Azul Zulu Builds von OpenJDK





Gute Neuigkeiten: Jeder Workload-Übergang von Oracle JDK zu Azul Zulu Builds von OpenJDK war erfolgreich! Das sind Milliarden von Workloads!

Das ist **kein** langer, anstrengender Prozess, wie Oracle Ihnen weismachen möchte. Einer unserer Bankkunden konvertierte 2.500 Anwendungen über das Wochenende; eines der größten Unterhaltungsunternehmen auf der Welt stellte Tausende von Anwendungen innerhalb einiger Monate um.

Das Azul Zulu JDK, kostenlos herunterladbar und kommerziell als Teil des Azul Platform Core-Pakets unterstützt, ist ein gleichwertiger Ersatz für Oracle JDK und bietet ein identisches Serviceniveau und dieselben Updates.

Oracle JDK baut auf dem OpenJDK-Repository auf und zentrale Funktionen wie die JVM, Bibliotheken usw. sind mit der Azul Zulu OpenJDK-Binärdistribution vollständig austauschbar. Es sind keine Modifizierungen am Quellcode und auch keine Neukompilierung des Anwendungscode nötig.

Außerdem besteht die Binärdatei Azul Zulu alle Tests des Technology Compatibility Kit (TCK), das als Teil der relevanten Java Specification Request (JSR) bereitgestellt wird. Es gibt über 150.000 Tests, die sicherstellen, dass eine Binärdatei die definierten Spezifikationen erfüllt und einen hohen Grad an Sicherheit in Bezug auf funktionelle Gleichwertigkeit zwischen

Azul schlägt eine 3-phasige Herangehensweise für den Übergang von Anwendungen zu Azul Platform Core vor:

- Planung
- Implementierung
- Test

getesteten JDKs bietet.

Basierend auf Kundenfeedback, und um die Unkompliziertheit des Übergangs im Voraus zu zeigen, beginnen wir diese Anleitung mit der Implementierungsphase, wobei mehrere Herangehensweisen an die Planung später im Dokument behandelt werden.

Phase 1: Planung: Identifizierung der Maschinen/Instanzen für den Übergang und der wichtigen verwendeten JDK-Versionen. (nähere Informationen siehe Seite 3)

Phase 2: Implementierung : Zeit für den Übergang!

Download der Azul Zulu JDK: Azul Platform Core-Kunden haben Anmeldeberechtigungen für den Zugriff auf die Site zum Herunterladen der Azul-Binärdatei. Für maximale Flexibilität wird eine Vielzahl von Formaten für jede unterstützte Version bereitgestellt.

Installation des Azul Zulu JDK: Der Installationsvorgang hängt davon ab, welches Format der Distribution verwendet wird:

Zip-Datei. Verwenden Sie ein Dienstprogramm oder Tool zum Entzippen des Archivs je nach verwendetem Betriebssystem. Es handelt sich dabei um eine manuelle Installationsmethode, das heißt das Azul JDK kann in einem Verzeichnis Ihrer Wahl installiert werden.

Komprimierte tar-Datei (.tar.gz). Verwenden Sie den UNIX-Befehl `tar -xvf <zulu-package>.tar.gz`. Dabei handelt es sich ebenfalls um eine manuelle Installationsmethode, das heißt es kann ein beliebiges Verzeichnis ausgewählt werden.

Windows MSI-Datei. Führen Sie von der Windows-Befehlszeile `aus msiexec /i <zulu-package>.msi /qn aus. Dadurch wird Zulu im Verzeichnis C:\Programme\Zulu\<zu-lu-jdk>\` installiert.

Linux RPM-Datei: Führen Sie die Installation von einer Eingabeaufforderung mithilfe des folgenden Befehls aus:

- Auf Red Hat Enterprise Linux: `yum install <zulu-package>.rpm`
- Auf SUSE Linux Enterprise Server: `zipper install <zulu-package>.rpm`

Linux DEB-Datei: Verwenden Sie auf Ubuntu oder Debian den folgenden Befehl: `apt install <zulu-package.deb>`.

MacOS DMG-Datei. Dies kann grafisch vom Desktop aus oder über die Befehlszeile mithilfe des folgenden Befehls installiert werden: `hdiutil mount <zulu_package>.dmg`

Eine vollständige Anleitung für die Installation finden Sie in der Azul-Dokumentation unter https://docs.azul.com/zulu/zuludocs/ZuluUserGuide/Title.htm.

Nähere Informationen zur Verwendung von Repositorys finden Sie im Anhang.

Zusätzliche Schriftarten: Oracle JDK vor JDK 11 enthielt zusätzliche Lucida-Schriftarten. Für die Kompatibilität mit dem Oracle JDK liefert Azul das Azul Commercial Compatibility Kit (CCK). Auf Desktop-Computern sollte das CCK installiert werden, wenn grafische Anwendungen verwendet werden. Details der verfügbaren CCK-Dateien und eine Installationsanleitung finden Sie unter https://www.azul.com/products/components/commercial-compatibility-kit/.

Desktop-Übergang: Wenn Sie Desktop-Anwendungen mit Browser Plug-ins (Applet) und Java Web Start-Funktionalität verwenden, lesen Sie bitte den Abschnitt „Desktop-Computer-Übergang“ am Ende dieses Dokuments. Dies ist nicht in Azul Zulu OpenJDK-Binärdistributionen enthalten.

Anwendungskonfiguration: Nach Installation des Azul Zulu JDK auf einem Computer kann es nötig sein, Anwendungen für die Verwendung des neuen JDK neu zu konfigurieren.

Wie Anwendungen bestimmen, wo sich die ausführbare Java-Datei befindet, unterscheidet sich jeweils. Hier sind einige gängige Szenarios.

1. Die Umgebungsvariable `PATH`. Diese wird je nach verwendetem Betriebssystem unterschiedlich gesetzt. Die Umgebungsvariable `PATH` sollte so modifiziert werden, dass sie das bin-Verzeichnis der Azul Zulu JDK-Installation als ersten Ort enthält, in dem sich eine ausführbare Datei mit dem Namen java befindet.

2. Die Umgebungsvariable `JAVA_HOME`. Ähnlich wie `PATH` wird dies ebenfalls je nach verwendetem Betriebssystem unterschiedlich gesetzt. `JAVA_HOME` gibt an, wo das JDK installiert ist, und sollte dementsprechend für das Azul Zulu JDK gesetzt werden. Beachten Sie, dass es sich dabei um das Installationsverzeichnis handelt, daher sollten Sie, anders als beim Ändern von PATH, nicht auf das bin-Unterverzeichnis verweisen. `JAVA_HOME` wird in einigen Anwendungen (z. B. einige Anwendungsserver), aber nicht universell von allen Anwendungen verwendet.

3. Für den Tomcat-Server sollte die Standardkonfigurationsdatei so modifiziert werden, dass sie den Ort der Azul Zulu JDK-Installation widerspiegelt.

Phase 3: Test : Funktionell gibt es keine Unterschiede zwischen dem Oracle JDK und dem Azul Zulu OpenJDK (außer denen, die bereits für Desktop-Computer angegeben wurden). Das bedeutet, es gibt keine Unterschiede bei der Ausführung Ihrer Java-Anwendung mit einem der JDK.

Sie sollten jedoch Ihre Standardtests für die verwendeten Anwendungen ausführen, um sicherzustellen, dass keine dieser Änderungen zwischen JDK-Updates das Anwendungsverhalten beeinflusst. Dies entspricht einer guten Vorgehensweise für genaue Like-for-Like-Versionsreleases.

Zurück zur Phase-1-Planung: Ihre Herangehensweise an die Übergangsplanung hängt von verschiedenen Faktoren ab:

- Zentrale oder dezentrale Herangehensweise an den Übergang
- Verfügbarkeit (oder Nichtverfügbarkeit) eines Software-Asset-Management-Produkts (SAM)
    - Wenn Sie ein SAM haben, ist die Erstellung eines Berichts installierter JDKs relativ einfach.
    - Sollte kein SAM vorhanden sehen, lesen Sie im Anhang nach, wie Sie auf verschiedenen Betriebssystemen bestimmen, welches JDK installiert ist. Bei einer dezentralen Herangehensweise können individuelle

Eigentümer diese Informationen bereits haben, dieser Vorgang ist daher möglicherweise nicht erforderlich.

- Möchten Sie, dass nur bestimmte Versionen des JDK und die neusten Sicherheits-Releases verwendet werden?

Abschließend ist zu sagen, dass Azul Zulu Builds von OpenJDK ein direkter Satz für das Oracle JDK sind (mit Ausnahme der vorher angegebenen Desktop-Funktionen). Sobald Java-Versionen identifiziert sind, erfordert der Übergang einfach die Installation der neuesten Versionen des Azul Zulu JDK und kleinere Anwendungen an der Anwendungskonfiguration, um den neuen Ort des JDK widerzuspiegeln.

Azul Zulu Builds von OpenJDK sind ein echter, gleichwertiger Ersatz für das Oracle JDK und bieten Benutzern erhebliche Kosteneinsparungen.

Anhang

So bestimmen Sie auf verschiedenen Betriebssystemen, welche JDK-Version bereitgestellt ist:

Ubuntu/Debian Linux

Sofern das JDK unter Verwendung des Package Managers installiert wurde, können installierte Versionen mithilfe des folgenden Befehls aufgelistet werden:

```
$ dpkg -l | egrep [jJ][rR][eE]\|[jJ][dD][kK]
```

Dies erzeugt eine Ausgabe ähnlich diesem Beispiel von einem Computer, der mit Ubuntu läuft.

```
ii default-jre-headless
2:1.11-68ubuntu1~18.04.1 amd64 Standard
Java oder Java-kompatible Laufzeit
(Headless)
ii jdk1.8 1.8.0202-1 amd64 Java
Platform Standard Edition Development Kit
ii openjdk-11-jre-headless:amd64 11.0.7
+10-2ubun-tu2~18.04 amd64 OpenJDK Java
Laufzeit, unter Verwendung von Hotspot JIT (Headless)
ii zulu-8 8.48.0.51-1 amd64 Azul
Systems Zulu JDK 8.48.0.51 (8u262-b19)
```

Dies zeigt, dass vier Java-Laufzeiten installiert sind:

1. Default Ubuntu 18.04 Java Runtime Environment (JRE).
2. Oracle JDK 8 Update 202 (auch wenn Oracle nicht explizit angegeben ist, besagt dies die Verwendung des Java(TM) Markenzeichens).
3. OpenJDK 11 Update 7 aus der Ubuntu Linux-Distribution.
4. Azul Zulu OpenJDK 8 Update 262.
   Eine alternative Herangehensweise ist das Durchsuchen des gesamten Dateisystems, wobei auch alle JDKs gefunden werden, die manuell installiert wurden (wie etwa durch Entpacken einer zip-Datei). Verwenden Sie den folgenden Befehl zum Suchen nach ausführbaren Dateien mit dem Namen java.

```
# find / -perm u+x -type f -name java
```

HINWEIS: Um einen vollständigen Scan des Dateisystems zu ermöglichen, sollte dies als Rootbenutzer ausgeführt werden.

Die aufgelisteten Pfade geben häufig die genaue Version des JDK an. Wenn das Verzeichnis jedoch `/opt/jdk8` lautet, muss die genaue installierte Version durch Ausführung des folgenden Java-Befehls bestimmt werden:

```
$ /opt/jdk8/bin/java -version
```

Dies erzeugt eine Ausgabe ähnlich der folgenden:

```
java version "1.8.0_202"

Java(TM) SE Runtime Environment (Build 1.8.0_202-b08)
Java HotSpot(TM) 64-Bit Server VM (Build 25.202-b08,
gemischter Modus)
```

Noch einmal, dies ist das Oracle JDK 8 Update 202.

Oracle/Red Hat Enterprise Linux/SUSE Linux Enterprise Server

Verwenden Sie den folgenden Befehl (als Root):

```
# rpm -qa --queryformat “%{NAME} %{VERSION} %{VENDOR}
\n” | egrep [jJ][rR][eE]\|[jJ][dD][kK]
```

Dies erzeugt eine Ausgabe wie:

```
jdk1.8 1.8.0_202 Oracle Corporation
```

Alternativ kann die vollständige Dateisystem-Suchmethode verwendet werden, wie für Ubuntu/Debian beschrieben.

Windows

Führen Sie von einer Eingabeaufforderung den Befehl wmic aus, der eine interaktive Shell startet. Geben Sie den folgenden Befehl ein:

```
product get name
```

In dieser Ausgabe sehen Sie etwas wie das Folgende:

```
Java 8 Update 202 (64-bit)
Java SE Development Kit 8 Update 202 (64-bit)
```

MacOS X

Um JDKs aufzulisten, die mithilfe des Packaging-Systems installiert wurden, verwenden Sie den folgenden Befehl:

```
$ pkgutil --pkgs | egrep jre\|jdk
```

Dies erzeugt eine Ausgabe wie diese:

```
com.oracle.jre
com.oracle.jdk-14.0.2
com.oracle.jdk8u202
```

Das Paket com.oracle.jre liefert keine Versionsnummer. Sie kann mit dem folgenden Befehl erhalten werden:

```
$ pkgutil –pkg-info com.oracle.jre
```

Der eine Ausgabe wie die folgende erzeugt, die zeigt, dass es sich um Java SE 10.0.2 handelt.

```
package-id: com.oracle.jre
version: 10.0.2.0.13
volume: /
location: Library/Internet Plug-Ins/
JavaAppletPlugin.plugin install-time: 1538383780
```

MacOS ist ein UNIX-basiertes Betriebssystem, daher kann auch die für Ubuntu/Debian Linux beschriebene vollständige Dateisystem-Suchmethode verwendet werden.

Solaris

Um JDKs aufzulisten, die mithilfe des Packaging-Systems installiert wurden, verwenden Sie diese zwei Befehle:

```
$ pkg list | egrep [jJ][rR][eE]\|[jJ][dD][kK]
$ pkginfo | egrep [jJ][rR][eE]\|[jJ][dD][kK]
```

Dies erzeugt eine Ausgabe wie dies hier: `runtime/java/jre-8`

```
1.8.0.181.12
```

und

```
  system    SUNWj8cfg    JDK 8.0 Host Config(1.8.0_202)
  system    SUNWj8dev    JDK 8.0 Dev. Tools (1.8.0_202)
  system    SUNWj8jmp    JDK 8.0 Man Pages: Japan
                         (1.8.0_202)
  system    SUNWj8man    JDK 8.0 Man Pages (1.8.0_202)
  system    SUNWj8rt     JDK 8.0 64-bit Runtime Env.
                         (1.8.0_202)
```

Solaris ist ein UNIX-basiertes Betriebssystem, daher kann auch die für Ubuntu/Debian Linux beschriebene vollständige Dateisystem-Suchmethode verwendet werden.

Desktop-Computer-Übergang

Vor dem Release von JDK 11 enthielt das Oracle JDK eine Reihe kommerzieller Funktionen, die nicht Teil des OpenJDK-Quellcodes sind. Zwei davon gelten speziell für die Bereitstellung von Anwendungen auf Desktop-Systemen, in diesem Fall bitte beachten:

1. Das Browser-Plugin: Dies wird verwendet, um die Verwendung von Applets über einen Web-Browser zu ermöglichen. Es gibt keine Open-Source-Alternative dazu und das Azul Zulu JDK beinhaltet keine gleichwertige Funktion. Die meisten Browser-Anbieter unterstützen keine Plugins mehr und Oracle hat die Unterstützung für das Browser-Plugin im März 2019 beendet (selbst für diejenigen mit einem kommerziellen Supportvertrag). In solchen Situationen gibt es die Option, Ihr vorhandenes JDK weiter zu verwenden und die potenziellen Sicherheitsrisiken zu akzeptieren, die damit verbunden sind, dass bekannte Anfälligkeiten nicht behandelt werden können.

2. Java Web Start: Diese Bereitstellungstechnologie sorgt dafür, dass Anwendungen sich selbst aktualisieren, wenn der Benutzer sie ausführt. Obwohl das Java Network Launch Protocol (JNLP), das Teil von Web Start ist, ein JSR hat, wurde dafür keine Referenzimplementierung bereitgestellt. Azul kann Builds von IcedTea-Web, eine Open-Source-Alternative zu Java Web Start, liefern. Dies ist kein gleichwertiger Ersatz, daher ist ein gewisser zusätzlicher Übergangsaufwand nötig. Azul hat mit anderen Kunden bei diesem Thema zusammengearbeitet. Der IcedTea-Web-Übergang ist nicht Gegenstand dieses Dokuments. Download unter https://www.azul.com/products/components/icedtea-web/.

Kontakt zu Azul

385 Moffett Park Drive, Suite 115

Sunnyvale, CA 94089 USA

+1.650.230.6500

www.azul.com

Kontakt zu Logicalis

Logicalis GmbH
Martin-Behaim-Straße 19-21
63263 Neu-Isenburg

de.azul@logicalis.de

azul