

# Java-Version-Kompatibilität mit Spring Boot

## Fehler verstehen

Wenn Sie `mvn spring-boot:run` mit Spring Boot 3.4.3 ausführen und den Fehler “class file has wrong version 61.0, should be 52.0” erhalten, deutet dies auf ein Java-Version-Kompatibilitätsproblem hin. In Java haben Klassendateien eine Versionsnummer, die der Java-Version entspricht, mit der sie kompiliert wurden:

- **Version 52.0** entspricht Java 8.
- **Version 61.0** entspricht Java 17.

Der Fehler deutet darauf hin, dass die Klassendateien Ihrer Spring Boot-Anwendung mit Java 17 (Version 61.0) kompiliert wurden, die Java Virtual Machine (JVM), die versucht, sie auszuführen, jedoch Klassendateien erwartet, die mit Java 8 (Version 52.0) kompatibel sind. Da die JVM von Java 8 Klassendateien, die mit Java 17 kompiliert wurden, nicht ausführen kann (weil Java rückwärtskompatibel, aber nicht vorwärtskompatibel ist), erhalten Sie diesen Fehler.

## Warum dies passiert

Spring Boot 3.4.3 erfordert **Java 17 oder später** als Mindestunterstützte Version. Dies bedeutet, dass Ihr Projekt sowohl kompiliert als auch mit mindestens Java 17 ausgeführt werden sollte. Die Diskrepanz tritt wahrscheinlich auf, weil:

1. **Kompilierung:** Ihr Projekt ist so konfiguriert, dass es mit Java 17 kompiliert wird, wodurch Klassendateien mit der Version 61.0 erzeugt werden. Dies wird normalerweise in Ihrer `pom.xml` mit dem `maven-compiler-plugin` oder der `<java.version>`-Eigenschaft festgelegt.
2. **Laufzeit:** Wenn Sie `mvn spring-boot:run` ausführen, startet das Spring Boot Maven-Plugin einen neuen Java-Prozess, um die Anwendung auszuführen. Das verwendete java-Executable stammt jedoch von einer Java 8-Installation (Version 52.0), die die mit Java 17 kompilierten Klassendateien nicht verarbeiten kann.

## Schritte zur Fehlerbehebung

Um dies zu beheben, müssen Sie sicherstellen, dass sowohl die Kompilierungs- als auch die Laufzeitumgebung Java 17 verwenden. Hier ist, wie Sie dies beheben können:

1. **Überprüfen Sie die Java-Version Ihres Projekts** Stellen Sie zunächst sicher, dass Ihr Projekt so konfiguriert ist, dass es Java 17 verwendet. In Ihrer `pom.xml` überprüfen Sie Folgendes:

```
<properties>
  <java.version>17</java.version>
</properties>
```

Diese Eigenschaft weist das `maven-compiler-plugin` an, Ihren Code mit Java 17 zu kompilieren. Spring Boot 3.4.3 setzt dies standardmäßig fest, aber es ist gut, dies zu überprüfen. Wenn es fehlt oder auf eine andere Version (z. B. 8) gesetzt ist, aktualisieren Sie es auf 17.

## 2. Installieren Sie Java 17

Stellen Sie sicher, dass Java 17 auf Ihrem System installiert ist. Sie können es von folgenden Quellen herunterladen:

- Adoptium (Eclipse Temurin)
- Oracle JDK (wenn Sie die Lizenzbedingungen akzeptieren)

Um zu überprüfen, ob Java 17 installiert ist, öffnen Sie ein Terminal und führen Sie Folgendes aus:

```
java -version
```

Wenn es nicht Java 17 anzeigt (z. B. `openjdk 17.x.x` oder ähnlich), installieren Sie es und fahren Sie fort.

## 3. Aktualisieren Sie Ihre Umgebung auf Java 17

Das Spring Boot Maven-Plugin verwendet das `java`-Executable aus Ihrer Umgebung, das normalerweise durch die `JAVA_HOME`-Umgebungsvariable oder den `java`-Befehl in Ihrem Systempfad bestimmt wird. Wenn Ihr aktueller `java`-Befehl auf Java 8 zeigt, müssen Sie ihn aktualisieren.

**Option A: Setzen Sie `JAVA_HOME` und `PATH`** Setzen Sie Ihre `JAVA_HOME`-Umgebungsvariable so, dass sie auf die Java 17-Installation zeigt, und stellen Sie sicher, dass der `java`-Befehl in Ihrem `PATH` diese verwendet.

### • Unter Linux/Mac:

1. Finden Sie Ihr Java 17-Installationsverzeichnis (z. B. `/usr/lib/jvm/java-17-openjdk` oder wo immer Sie es installiert haben).
2. Setzen Sie `JAVA_HOME` und aktualisieren Sie `PATH` in Ihrem Terminal:

```
export JAVA_HOME=/pfad/zu/java-17  
export PATH=$JAVA_HOME/bin:$PATH
```

3. Überprüfen Sie:

```
java -version
```

Es sollte nun Java 17 anzeigen.

Um dies dauerhaft zu machen, fügen Sie die `export`-Zeilen zu Ihrer Shell-Konfigurationsdatei (z. B. `~/.bashrc`, `~/.zshrc`) hinzu.

### • Unter Windows:

1. Finden Sie das Java 17-Installationsverzeichnis (z. B. `C:\Program Files\Java\jdk-17`).

## 2. Setzen Sie JAVA\_HOME:

- Öffnen Sie das Systemeigenschaftenfenster (suchen Sie nach "Umgebungsvariablen" im Startmenü).
- Unter "Systemvariablen" fügen Sie JAVA\_HOME hinzu oder aktualisieren Sie es auf C:\Program Files\Java\jdk-17.

## 3. Aktualisieren Sie Path:

- In "Systemvariablen" bearbeiten Sie Path und fügen Sie %JAVA\_HOME%\bin hinzu (stellen Sie sicher, dass es oben steht oder entfernen Sie ältere Java-Pfade wie Java 8).

## 4. Öffnen Sie ein neues Eingabeaufforderungsfenster und überprüfen Sie:

```
java -version
```

Es sollte Java 17 anzeigen.

**Option B: Verwenden Sie eine spezifische Java-Version für Maven** Wenn Sie mehrere Java-Versionen installiert haben und Ihre systemweiten JAVA\_HOME-Einstellungen nicht ändern möchten, können Sie Maven mit einer spezifischen Java-Version ausführen:

- Bevor Sie mvn spring-boot:run ausführen, setzen Sie JAVA\_HOME nur für diesen Befehl:

```
JAVA_HOME=/pfad/zu/java-17 mvn spring-boot:run
```

Dies stellt sicher, dass das Spring Boot Maven-Plugin Java 17 verwendet, ohne Ihre Systemeinstellungen zu ändern.

## 4. Starten Sie die Anwendung erneut

 Mit Java 17 richtig eingestellt, führen Sie Folgendes aus:

```
mvn spring-boot:run
```

Das Plugin verwendet nun Java 17, um Ihre Anwendung auszuführen, was mit der kompilierten Klassendateiversion (61.0) übereinstimmt, und der Fehler sollte verschwinden.

## Zusätzliche Überprüfungen

- **Mavens Java-Version:** Führen Sie mvn -version aus, um zu sehen, welche Java-Version Maven verwendet. Es sollte mit Ihrem JAVA\_HOME übereinstimmen. Wenn nicht, stellen Sie sicher, dass Ihr PATH und JAVA\_HOME korrekt konfiguriert sind.
- **Abhängigkeiten:** Wenn der Fehler weiterhin besteht, könnte eine Abhängigkeit mit Java 17 kompiliert sein, während Ihre Laufzeit weiterhin falsch konfiguriert ist. Da Spring Boot 3.4.3 Java 17 erzwingt, sollte das Einstellen Ihrer Laufzeit auf Java 17 dies beheben.

## **Warum nicht Java 8?**

Spring Boot 3.4.3 unterstützt Java 8 nicht. Wenn Sie Java 8 verwenden müssen, müssen Sie auf eine ältere Spring Boot-Version (z. B. 2.x) herunterstufen, was jedoch nicht empfohlen wird, es sei denn, Sie haben spezifische Einschränkungen. Da Ihre Klassendateien die Version 61.0 haben, stimmt das Festhalten an Java 17 mit den Anforderungen von Spring Boot 3.4.3 überein.

## **Zusammenfassung der Endlösung**

Um "class file has wrong version 61.0, should be 52.0" beim Ausführen von `mvn spring-boot:run` mit Spring Boot 3.4.3 zu beheben:

1. Stellen Sie sicher, dass `<java.version>17</java.version>` in Ihrer `pom.xml` enthalten ist.
2. Installieren Sie Java 17, falls noch nicht vorhanden.
3. Setzen Sie `JAVA_HOME` auf Ihr Java 17-Installationsverzeichnis.
4. Aktualisieren Sie Ihren PATH, sodass `java -version` Java 17 anzeigt.
5. Führen Sie `mvn spring-boot:run` erneut aus.

Dies stellt sicher, dass Ihre Laufzeit mit Ihren kompilierten Java 17-Klassendateien übereinstimmt und das Kompatibilitätsproblem behoben wird.