

Compatibilité des versions Java avec Spring Boot

Comprendre l'erreur

Lorsque vous exécutez `mvn spring-boot:run` avec Spring Boot 3.4.3 et que vous rencontrez l'erreur "class file has wrong version 61.0, should be 52.0," cela indique un problème de compatibilité de version de Java. En Java, les fichiers de classe ont un numéro de version correspondant à la version de Java avec laquelle ils ont été compilés :

- **Version 52.0** correspond à Java 8.
- **Version 61.0** correspond à Java 17.

L'erreur suggère que les fichiers de classe de votre application Spring Boot ont été compilés avec Java 17 (version 61.0), mais la machine virtuelle Java (JVM) qui tente de les exécuter s'attend à des fichiers de classe compatibles avec Java 8 (version 52.0). Puisque la JVM de Java 8 ne peut pas exécuter les fichiers de classe compilés avec Java 17 (car Java est rétrocompatible mais pas compatible vers l'avant dans ce contexte), vous obtenez cette erreur.

Pourquoi cela se produit

Spring Boot 3.4.3 nécessite **Java 17 ou une version ultérieure** comme version minimale prise en charge. Cela signifie que votre projet doit être à la fois compilé et exécuté en utilisant au moins Java 17. Le désaccord est probablement dû au fait que :

1. **Compilation** : Votre projet est configuré pour être compilé avec Java 17, produisant des fichiers de classe avec la version 61.0. Cela est généralement défini dans votre `pom.xml` avec le `maven-compiler-plugin` ou la propriété `<java.version>`.
2. **Exécution** : Lorsque vous exécutez `mvn spring-boot:run`, le plugin Maven Spring Boot lance un nouveau processus Java pour exécuter l'application. Cependant, l'exécutable `java` qu'il utilise provient d'une installation Java 8 (version 52.0), qui ne peut pas gérer les fichiers de classe compilés avec Java 17.

Étapes pour corriger le problème

Pour résoudre ce problème, vous devez vous assurer que les environnements de compilation et d'exécution utilisent tous deux Java 17. Voici comment le faire :

1. Vérifiez la version de Java de votre projet Tout d'abord, confirmez que votre projet est configuré pour utiliser Java 17. Dans votre `pom.xml`, vérifiez ce qui suit :

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

Cette propriété indique au `maven-compiler-plugin` de compiler votre code avec Java 17. Spring Boot 3.4.3 définit cela par défaut, mais il est bon de vérifier. Si elle est manquante ou définie sur une version différente (par exemple, 8), mettez-la à jour sur 17.

2. Installez Java 17 Assurez-vous que Java 17 est installé sur votre système. Vous pouvez le télécharger depuis :

- Adoptium (Eclipse Temurin)
- Oracle JDK (si vous acceptez les termes de la licence)

Pour vérifier si Java 17 est installé, ouvrez un terminal et exécutez :

```
java -version
```

Si ce n'est pas Java 17 (par exemple, `openjdk 17.x.x` ou similaire), installez-le et poursuivez.

3. Mettez à jour votre environnement pour utiliser Java 17 Le plugin Maven Spring Boot utilise l'exécutable `java` de votre environnement, généralement déterminé par la variable d'environnement `JAVA_HOME` ou la commande `java` dans le `PATH` de votre système. Si votre commande `java` actuelle pointe vers Java 8, vous devez la mettre à jour.

Option A : Définir `JAVA_HOME` et `PATH` Définissez votre variable d'environnement `JAVA_HOME` pour qu'elle pointe vers l'installation de Java 17 et assurez-vous que la commande `java` dans votre `PATH` l'utilise.

- **Sur Linux/Mac :**

1. Trouvez votre répertoire d'installation de Java 17 (par exemple, `/usr/lib/jvm/java-17-openjdk` ou là où vous l'avez installé).
2. Définissez `JAVA_HOME` et mettez à jour `PATH` dans votre terminal :

```
export JAVA_HOME=/chemin/vers/java-17
export PATH=$JAVA_HOME/bin:$PATH
```

3. Vérifiez :

```
java -version
```

Il devrait maintenant afficher Java 17.

Pour rendre cela permanent, ajoutez les lignes `export` à votre fichier de configuration de shell (par exemple, `~/.bashrc`, `~/.zshrc`).

- **Sur Windows :**

1. Trouvez le répertoire d'installation de Java 17 (par exemple, `C:\Program Files\Java\jdk-17`).
2. Définissez `JAVA_HOME` :
 - Ouvrez la fenêtre des propriétés du système (recherchez "Variables d'environnement" dans le menu Démarrer).
 - Sous "Variables système", ajoutez ou mettez à jour `JAVA_HOME` à `C:\Program Files\Java\jdk-17`.
3. Mettez à jour `Path` :
 - Dans "Variables système", éditez `Path` et ajoutez `%JAVA_HOME%\bin` (assurez-vous qu'il est en haut ou supprimez les chemins Java plus anciens comme Java 8).
4. Ouvrez un nouvel Invite de commandes et vérifiez :

```
java -version
```

Il devrait afficher Java 17.

Option B : Utiliser une version spécifique de Java pour Maven Si vous avez plusieurs versions de Java installées et que vous ne souhaitez pas changer votre `JAVA_HOME` système, vous pouvez exécuter Maven avec une version spécifique de Java :

- Avant d'exécuter `mvn spring-boot:run`, définissez `JAVA_HOME` juste pour cette commande :

```
JAVA_HOME=/chemin/vers/java-17 mvn spring-boot:run
```

Cela garantit que le plugin Maven Spring Boot utilise Java 17 sans modifier vos paramètres système.

4. Exécutez l'application à nouveau Avec Java 17 correctement défini, exécutez :

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

Le plugin utilisera maintenant Java 17 pour exécuter votre application, correspondant à la version des fichiers de classe compilés (61.0), et l'erreur devrait disparaître.

Vérifications supplémentaires

- **Version de Java de Maven** : Exécutez `mvn -version` pour voir quelle version de Java utilise Maven. Elle doit correspondre à votre `JAVA_HOME`. Si ce n'est pas le cas, assurez-vous que votre `PATH` et `JAVA_HOME` sont correctement configurés.
- **Dépendances** : Si l'erreur persiste, une dépendance pourrait être compilée avec Java 17 tandis que votre runtime est toujours mal configuré. Puisque Spring Boot 3.4.3 impose Java 17, assurer que votre runtime est Java 17 devrait résoudre cela.

Pourquoi pas Java 8 ?

Spring Boot 3.4.3 ne prend pas en charge Java 8. Si vous devez utiliser Java 8, vous devrez rétrograder vers une version plus ancienne de Spring Boot (par exemple, 2.x), mais ce n'est pas recommandé sauf si vous avez des contraintes spécifiques. Puisque vos fichiers de classe sont de version 61.0, rester avec Java 17 est conforme aux exigences de Spring Boot 3.4.3.

Résumé de la solution finale

Pour corriger "class file has wrong version 61.0, should be 52.0" lors de l'exécution de `mvn spring-boot:run` avec Spring Boot 3.4.3 :

1. Assurez-vous que `<java.version>17</java.version>` est dans votre `pom.xml`.
2. Installez Java 17 s'il n'est pas déjà présent.
3. Définissez `JAVA_HOME` sur votre répertoire d'installation de Java 17.
4. Mettez à jour votre `PATH` pour que `java -version` affiche Java 17.
5. Exécutez `mvn spring-boot:run` à nouveau.

Cela garantit que votre runtime correspond à vos fichiers de classe Java 17 compilés, résolvant le problème de compatibilité.