

Java バージョンと Spring Boot の互換性

エラーの理解

`mvn spring-boot:run` を実行するときに、Spring Boot 3.4.3 でエラー“class file has wrong version 61.0, should be 52.0”が表示された場合、これは Java バージョンの互換性問題を示しています。Java では、クラスファイルにはコンパイルされた Java バージョンに対応するバージョン番号があります。

- **バージョン 52.0** は Java 8 に対応します。
- **バージョン 61.0** は Java 17 に対応します。

このエラーは、Spring Boot アプリケーションのクラスファイルが Java 17（バージョン 61.0）でコンパイルされているが、実行を試みる Java バージョンマシン（JVM）は Java 8（バージョン 52.0）に対応するクラスファイルを期待していることを示しています。Java 8 の JVM は、Java 17 でコンパイルされたクラスファイルを実行できないため（Java は後方互換性がありますが、前方互換性はありません）、このエラーが発生します。

なぜこれが起こるのか

Spring Boot 3.4.3 は **Java 17 以降** を最低サポートバージョンとしています。これは、プロジェクトがコンパイルと実行の両方で少なくとも Java 17 を使用する必要があることを意味します。この不一致は以下の理由で発生します。

1. **コンパイル**: プロジェクトは Java 17 でコンパイルされるように設定されており、バージョン 61.0 のクラスファイルを生成します。これは通常、`pom.xml` で `maven-compiler-plugin` または `<java.version>` プロパティで設定されます。
2. **ランタイム**: `mvn spring-boot:run` を実行すると、Spring Boot Maven プラグインは新しい Java プロセスを起動してアプリケーションを実行します。しかし、使用される `java` 実行ファイルは Java 8 のインストール（バージョン 52.0）からであり、Java 17 でコンパイルされたクラスファイルを処理できません。

問題を解決するための手順

これを解決するには、コンパイル環境とランタイム環境の両方で Java 17 を使用するようにする必要があります。以下にその方法を示します。

1. プロジェクトの Java バージョンを確認する まず、プロジェクトが Java 17 を使用するよう設定されていることを確認します。`pom.xml` で以下を確認します。

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

このプロパティは、maven-compiler-plugin にコードを Java 17 でコンパイルするように指示します。Spring Boot 3.4.3 はデフォルトでこれを設定しますが、確認することは良いです。設定が欠けているか、他のバージョン（例：8）に設定されている場合は、17 に更新してください。

2. Java 17 をインストールする システムに Java 17 がインストールされていることを確認します。以下からダウンロードできます。

- Adoptium (Eclipse Temurin)
- Oracle JDK（ライセンス条件を受け入れる場合）

Java 17 がインストールされているかどうかを確認するには、ターミナルを開いて以下を実行します。

```
java -version
```

Java 17（例：openjdk 17.x.x など）が表示されない場合は、インストールしてから続けます。

3. 環境を Java 17 に更新する Spring Boot Maven プラグインは、環境変数 JAVA_HOME またはシステムの PATH にある java コマンドから java 実行ファイルを使用します。現在の java コマンドが Java 8 を指している場合は、更新する必要があります。

オプション A: JAVA_HOME と PATH を設定する JAVA_HOME 環境変数を Java 17 のインストール先に設定し、PATH にある java コマンドがそれを使用するようにします。

• **Linux/Mac:**

1. Java 17 のインストールディレクトリを確認します（例：/usr/lib/jvm/java-17-openjdk またはインストール先）。
2. ターミナルで JAVA_HOME を設定し、PATH を更新します。

```
export JAVA_HOME=/path/to/java-17
```

```
export PATH=$JAVA_HOME/bin:$PATH
```

3. 確認します。

```
java -version
```

これで Java 17 が表示されるはずです。

永続的に設定するには、シェル設定ファイル（例： ~/.bashrc、 ~/.zshrc）に export 行を追加します。

• **Windows:**

1. Java 17 のインストールディレクトリを確認します（例：C:\Program Files\Java\jdk-17）。
2. JAVA_HOME を設定します。

- 「環境変数」を検索して「システムのプロパティ」ウィンドウを開きます。
 - 「システム変数」の下で JAVA_HOME を追加または更新し、C:\Program Files\Java\jdk-17 に設定します。
3. Path を更新します。
- 「システム変数」で Path を編集し、%JAVA_HOME%\bin を追加します（Java 8 などの古い Java パスを削除するか、最上位に設定します）。
4. 新しいコマンドプロンプトを開き、確認します。

```
java -version
```

これで Java 17 が表示されるはずです。

オプション B: Maven で特定の Java バージョンを使用する 複数の Java バージョンがインストールされている場合、システム全体の JAVA_HOME を変更せずに Maven を特定の Java バージョンで実行することもできます。

- mvn spring-boot:run を実行する前に、そのコマンド用に JAVA_HOME を設定します。

```
JAVA_HOME=/path/to/java-17 mvn spring-boot:run
```

これにより、Spring Boot Maven プラグインが Java 17 を使用するようになります。

4. アプリケーションを再度実行する Java 17 が正しく設定されたら、以下を実行します。

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

プラグインはこれで Java 17 を使用してアプリケーションを実行し、コンパイルされたクラスファイルのバージョン（61.0）と一致するため、エラーは解消されます。

追加の確認

- **Maven の Java バージョン:** mvn -version を実行して、Maven が使用する Java バージョンを確認します。これは JAVA_HOME と一致している必要があります。一致しない場合は、PATH と JAVA_HOME の設定を確認してください。
- **依存関係:** エラーが解消されない場合、依存関係の一部が Java 17 でコンパイルされている可能性があります。Spring Boot 3.4.3 は Java 17 を強制するため、ランタイムが Java 17 に設定されていることを確認することで解決するはずです。

なぜ Java 8 ではないのか

Spring Boot 3.4.3 は Java 8 をサポートしていません。Java 8 を使用する必要がある場合は、古い Spring Boot バージョン（例：2.x）にダウングレードする必要がありますが、特定の制約がない限りは推奨されません。クラスファイルがバージョン 61.0 の場合、Java 17 を使用することで Spring Boot 3.4.3 の要件に適合します。

最終的な解決策のまとめ

“class file has wrong version 61.0, should be 52.0”エラーを `mvn spring-boot:run` で Spring Boot 3.4.3 を実行した際に解決するためには、以下の手順を実行します。

1. `pom.xml` に `<java.version>17</java.version>` を設定します。
2. Java 17 がインストールされていない場合はインストールします。
3. `JAVA_HOME` を Java 17 のインストールディレクトリに設定します。
4. `PATH` を更新し、`java -version` が Java 17 を表示するようにします。
5. 再度 `mvn spring-boot:run` を実行します。

これにより、ランタイムがコンパイルされた Java 17 のクラスファイルと一致し、互換性の問題が解消されます。