Java 版本与 Spring Boot 兼容性

理解错误

当你在运行 `mvn spring-boot:run` 时遇到错误“class file has wrong version 61.0, should be 52.0”，这表明存在 Java 版本兼容性问题。在 Java 中，类文件有一个版本号，对应它们编译时使用的 Java 版本：

- **版本 52.0** 对应 Java 8。
- **版本 61.0** 对应 Java 17。

错误表明你的 Spring Boot 应用程序的类文件是用 Java 17（版本 61.0）编译的，但尝试运行它们的 Java 虚拟机（JVM）期望的是与 Java 8（版本 52.0）兼容的类文件。由于 Java 8 的 JVM 不能执行用 Java 17 编译的类文件（因为 Java 是向后兼容但不是向前兼容），你会遇到这个错误。

为什么会发生

Spring Boot 3.4.3 需要 **Java 17 或更高版本** 作为其最低支持版本。这意味着你的项目应该既编译又运行使用至少 Java 17。不匹配可能发生的原因是：

1. **编译**：你的项目配置为使用 Java 17 进行编译，生成版本为 61.0 的类文件。这通常在 `pom.xml` 中通过 `maven-compiler-plugin` 或 `<java.version>` 属性设置。
2. **运行时**：当你执行 `mvn spring-boot:run` 时，Spring Boot Maven 插件启动一个新的 Java 进程来运行应用程序。然而，它使用的是 Java 8 安装（版本 52.0）的 `java` 可执行文件，无法处理用 Java 17 编译的类文件。

解决问题的步骤

要解决这个问题，你需要确保编译和运行环境都使用 Java 17。以下是解决方法：

1. 验证项目的 Java 版本 首先，确认你的项目设置为使用 Java 17。在你的 `pom.xml` 中检查以下内容：

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

这个属性告诉 `maven-compiler-plugin` 使用 Java 17 编译代码。Spring Boot 3.4.3 默认设置这个属性，但最好验证一下。如果缺失或设置为不同版本（例如 8），将其更新为 17。

2. 安装 Java 17 确保你的系统上安装了 Java 17。你可以从以下链接下载：

- Adoptium (Eclipse Temurin)

- Oracle JDK（如果你接受许可条款）

要检查是否安装了 Java 17，打开终端并运行：

```
java -version
```

如果它没有显示 Java 17（例如 `openjdk 17.x.x` 或类似），安装它并继续。

3. 更新环境以使用 Java 17 Spring Boot Maven 插件使用环境中的 `java` 可执行文件，通常由 `JAVA_HOME` 环境变量或系统 PATH 中的 `java` 命令决定。如果当前的 `java` 命令指向 Java 8，你需要更新它。

选项 A：设置 JAVA_HOME 和 PATH 设置你的 `JAVA_HOME` 环境变量指向 Java 17 安装目录，并确保 PATH 中的 `java` 命令使用它。

- **在 Linux/Mac 上**：
  1. 找到你的 Java 17 安装目录（例如 `/usr/lib/jvm/java-17-openjdk` 或你安装的位置）。
  2. 在终端中设置 `JAVA_HOME` 并更新 `PATH`：

     ```
     export JAVA_HOME=/path/to/java-17
     export PATH=$JAVA_HOME/bin:$PATH
     ```

  3. 验证：

     ```
     java -version
     ```

     它现在应该显示 Java 17。

  要使其永久有效，将 `export` 行添加到你的 shell 配置文件（例如 `~/.bashrc`，`~/.zshrc`）。
- **在 Windows 上**：
  1. 找到 Java 17 安装目录（例如 `C:\Program Files\Java\jdk-17`）。
  2. 设置 `JAVA_HOME`：
     - 打开系统属性窗口（在开始菜单中搜索“环境变量”）。
     - 在“系统变量”中，添加或更新 `JAVA_HOME` 为 `C:\Program Files\Java\jdk-17`。
  3. 更新 `Path`：
     - 在“系统变量”中，编辑 `Path` 并添加 `%JAVA_HOME%\bin`（确保它在顶部或删除旧的 Java 路径，如 Java 8）。
  4. 打开新的命令提示符并验证：

     ```
     java -version
     ```

     它应该显示 Java 17。

选项 B：为 Maven 使用特定的 Java 版本 如果你安装了多个 Java 版本并且不想更改系统范围的 `JAVA_HOME`，可以在运行 Maven 时使用特定的 Java 版本：

- 在运行 `mvn spring-boot:run` 之前，为该命令设置 `JAVA_HOME`：

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

  这确保 Spring Boot Maven 插件使用 Java 17，而不改变你的系统设置。

4. 再次运行应用程序 设置好 Java 17 后，运行：

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

插件现在将使用 Java 17 执行你的应用程序，匹配编译的类文件版本（61.0），错误应该消失。

额外检查

- **Maven 的 Java 版本**：运行 `mvn -version` 以查看 Maven 使用的 Java 版本。它应该与你的 `JAVA_HOME` 匹配。如果不匹配，确保你的 PATH 和 `JAVA_HOME` 配置正确。
- **依赖项**：如果错误持续存在，某个依赖项可能是用 Java 17 编译的，而你的运行时仍然配置不当。由于 Spring Boot 3.4.3 强制使用 Java 17，确保你的运行时是 Java 17 可以解决这个问题。

为什么不是 Java 8?

Spring Boot 3.4.3 不支持 Java 8。如果你需要使用 Java 8，你必须降级到较旧的 Spring Boot 版本（例如 2.x），但除非你有特定的约束，否则不建议这样做。由于你的类文件是版本 61.0，坚持使用 Java 17 符合 Spring Boot 3.4.3 的要求。

最终解决方案总结

要解决在运行 `mvn spring-boot:run` 时遇到的“class file has wrong version 61.0, should be 52.0”错误，使用 Spring Boot 3.4.3：

1. 确保 `<java.version>17</java.version>` 在你的 `pom.xml` 中。
2. 如果尚未安装，安装 Java 17。
3. 将 `JAVA_HOME` 设置为你的 Java 17 安装目录。
4. 更新你的 PATH，使 `java -version` 显示 Java 17。
5. 再次运行 `mvn spring-boot:run`。

这确保你的运行时与编译的 Java 17 类文件匹配，解决兼容性问题。