

# Compatibilidad de Versiones de Java con Spring Boot

## Entendiendo el Error

Cuando ejecutas `mvn spring-boot:run` con Spring Boot 3.4.3 y encuentras el error “class file has wrong version 61.0, should be 52.0,” indica un problema de compatibilidad de versiones de Java. En Java, los archivos de clase tienen un número de versión que corresponde a la versión de Java con la que se compilaron:

- **Versión 52.0** corresponde a Java 8.
- **Versión 61.0** corresponde a Java 17.

El error sugiere que los archivos de clase de tu aplicación Spring Boot se compilaron con Java 17 (versión 61.0), pero la Máquina Virtual de Java (JVM) que intenta ejecutarlos espera archivos de clase compatibles con Java 8 (versión 52.0). Dado que la JVM de Java 8 no puede ejecutar archivos de clase compilados con Java 17 (porque Java es compatible hacia atrás pero no hacia adelante en este contexto), obtienes este error.

## Por Qué Sucede Esto

Spring Boot 3.4.3 requiere **Java 17 o posterior** como su versión mínima soportada. Esto significa que tu proyecto debe ser compilado y ejecutado utilizando al menos Java 17. La discrepancia probablemente ocurre porque:

1. **Compilación:** Tu proyecto está configurado para compilar con Java 17, produciendo archivos de clase con versión 61.0. Esto generalmente se establece en tu `pom.xml` con el `maven-compiler-plugin` o la propiedad `<java.version>`.
2. **Tiempo de Ejecución:** Cuando ejecutas `mvn spring-boot:run`, el plugin Maven de Spring Boot lanza un nuevo proceso de Java para ejecutar la aplicación. Sin embargo, el ejecutable `java` que utiliza proviene de una instalación de Java 8 (versión 52.0), que no puede manejar los archivos de clase compilados con Java 17.

## Pasos para Solucionar el Problema

Para resolver esto, necesitas asegurarte de que tanto el entorno de compilación como el de tiempo de ejecución utilicen Java 17. Aquí cómo hacerlo:

**1. Verifica la Versión de Java de tu Proyecto** Primero, confirma que tu proyecto está configurado para usar Java 17. En tu `pom.xml`, verifica lo siguiente:

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

Esta propiedad le indica al `maven-compiler-plugin` que compile tu código con Java 17. Spring Boot 3.4.3 establece esto por defecto, pero es bueno verificar. Si falta o está configurado para una versión diferente (por ejemplo, 8), actualízalo a 17.

**2. Instala Java 17** Asegúrate de que Java 17 esté instalado en tu sistema. Puedes descargarlo desde:

- Adoptium (Eclipse Temurin)
- Oracle JDK (si aceptas los términos de la licencia)

Para verificar si Java 17 está instalado, abre una terminal y ejecuta:

```
java -version
```

Si no muestra Java 17 (por ejemplo, `openjdk 17.x.x` o similar), instálalo y continúa.

**3. Actualiza tu Entorno para Usar Java 17** El plugin Maven de Spring Boot utiliza el ejecutable `java` de tu entorno, generalmente determinado por la variable de entorno `JAVA_HOME` o el comando `java` en el `PATH` de tu sistema. Si tu comando `java` actual apunta a Java 8, necesitas actualizarlo.

**Opción A: Establece `JAVA_HOME` y `PATH`** Establece tu variable de entorno `JAVA_HOME` para que apunte a la instalación de Java 17 y asegúrate de que el comando `java` en tu `PATH` lo utilice.

- **En Linux/Mac:**

1. Encuentra tu directorio de instalación de Java 17 (por ejemplo, `/usr/lib/jvm/java-17-openjdk` o donde lo instalaste).
2. Establece `JAVA_HOME` y actualiza `PATH` en tu terminal:

```
export JAVA_HOME=/ruta/a/java-17
```

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

3. Verifica:

```
java -version
```

Ahora debería mostrar Java 17.

Para hacer esto permanente, agrega las líneas `export` a tu archivo de configuración de shell (por ejemplo, `~/.bashrc`, `~/.zshrc`).

- **En Windows:**

1. Encuentra el directorio de instalación de Java 17 (por ejemplo, `C:\Program Files\Java\jdk-17`).
2. Establece `JAVA_HOME`:

- Abre la ventana de Propiedades del Sistema (busca “Variables de Entorno” en el menú de Inicio).
- En “Variables del Sistema,” agrega o actualiza JAVA\_HOME a C:\Program Files\Java\jdk-17.

### 3. Actualiza Path:

- En “Variables del Sistema,” edita Path y agrega %JAVA\_HOME%\bin (asegúrate de que esté en la parte superior o elimina rutas de Java anteriores como Java 8).

### 4. Abre un nuevo Símbolo del Sistema y verifica:

```
java -version
```

Ahora debería mostrar Java 17.

**Opción B: Usa una Versión Específica de Java para Maven** Si tienes múltiples versiones de Java instaladas y no deseas cambiar tu JAVA\_HOME del sistema, puedes ejecutar Maven con una versión específica de Java:

- Antes de ejecutar `mvn spring-boot:run`, establece JAVA\_HOME solo para ese comando:

```
JAVA_HOME=/ruta/a/java-17 mvn spring-boot:run
```

Esto asegura que el plugin Maven de Spring Boot utilice Java 17 sin alterar tus configuraciones del sistema.

### 4. Ejecuta la Aplicación Nuevamente

 Con Java 17 configurado correctamente, ejecuta:

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

El plugin ahora utilizará Java 17 para ejecutar tu aplicación, coincidiendo con la versión del archivo de clase compilado (61.0), y el error debería desaparecer.

### Verificaciones Adicionales

- **Versión de Java de Maven:** Ejecuta `mvn -version` para ver qué versión de Java utiliza Maven. Debería coincidir con tu JAVA\_HOME. Si no, asegúrate de que tu PATH y JAVA\_HOME estén configurados correctamente.
- **Dependencias:** Si el error persiste, una dependencia podría estar compilada con Java 17 mientras que tu tiempo de ejecución sigue mal configurado. Dado que Spring Boot 3.4.3 exige Java 17, asegurarte de que tu tiempo de ejecución sea Java 17 debería resolver esto.

## ¿Por Qué No Java 8?

Spring Boot 3.4.3 no soporta Java 8. Si necesitas usar Java 8, deberías bajar a una versión anterior de Spring Boot (por ejemplo, 2.x), pero eso no se recomienda a menos que tengas restricciones específicas. Dado que tus archivos de clase son de versión 61.0, mantenerte con Java 17 se alinea con los requisitos de Spring Boot 3.4.3.

## Resumen de la Solución Final

Para solucionar “class file has wrong version 61.0, should be 52.0” al ejecutar `mvn spring-boot:run` con Spring Boot 3.4.3:

1. Asegúrate de que `<java.version>17</java.version>` esté en tu `pom.xml`.
2. Instala Java 17 si no está presente.
3. Establece `JAVA_HOME` en tu directorio de instalación de Java 17.
4. Actualiza tu `PATH` para que `java -version` muestre Java 17.
5. Ejecuta `mvn spring-boot:run` nuevamente.

Esto asegura que tu tiempo de ejecución coincida con tus archivos de clase compilados en Java 17, resolviendo el problema de compatibilidad.