

# Tools für dein (S)EP Build Systems

Nico Mexis

13. Juli 2023

# Wofür ist das gut?

- ▶ Automatisches Bauen des Projekts
- ▶ Determinismus
- ▶ Reproduzierbarkeit
- ▶ Bibliothekenverwaltung
- ▶ Früher: `javac -classpath C:/libs/lib1.jar;C:/libs/lib1.jar -d classes src/TestClass.java ...`
- ▶ Heute: `mvn package`

- ▶ Ant/Ivy
- ▶ Gradle
- ▶ **Maven**

- ▶ Eines der ersten Build-Systeme für Java
- ▶ Build-Optionen in `build.xml`-Datei festgelegt
- ▶ Also: Einfache XML-Syntax
- ▶ Nur wenige, primitive Plugins
- ▶ Verbos, aber klar
- ▶ URL: <https://ant.apache.org/>
- ▶ URL: <https://ant.apache.org/ivy/>

- ▶ Build-Optionen in `build.gradle(.kts)`-Datei festgelegt
- ▶ Groovy- oder Kotlin-Script-Syntax
- ▶ Schwerer zu lernen, sehr mächtig
- ▶ Viele Plugins verfügbar
- ▶ URL: <https://gradle.org/>

- ▶ Build-Optionen in `pom.xml`-Datei festgelegt
- ▶ Also: Einfache XML-Syntax
- ▶ Schnell zu lernen, aber sehr mächtig
- ▶ Viele Plugins verfügbar
- ▶ URL: <https://maven.apache.org/>

- ▶ Theoretischer Vergleich: <https://www.baeldung.com/ant-maven-gradle>
- ▶ Praktischer Vergleich: <https://github.com/nelenkov/android-backup-extractor>
- ▶ JF Demo Projekt: <https://git.fim.uni-passau.de/lukasczy/jsfdemo>