Java Modules

Arten von Java-Splitting

- **Buildmanager:**
- **Maven:** Zentrale `pom.xml`-Konfiguration für Abhängigkeiten, spezifisch für Java gemacht
- **Gradle:** Unterstützt mehrere Sprachen, aber sehr gut für Java; nutzt Groovy- oder Kotlin-DSL (Domain Specific Language) zur Konfiguration
- **Apache Ant:** Älter als Maven und Gradle, nutzt ebenfalls XML (`build.xml`) zur Konfiguration; wird wegen Einfachheit und Direktheit weiterhin benutzt
- **Java Platform Module System (JPMS/Jigsaw)** *(verstehe ich noch nicht ganz)*: Seit Java 9; erlaubt es, Code auf Sprachebene in Module aufzuteilen. Jedes Modul deklariert in `module-info.java`, welche Pakete es exportiert und welche Module es benötigt – interne Pakete bleiben für andere unsichtbar.

  ```
  module meine.app {
      requires java.sql;          // Ich brauche das SQL-Modul
      exports com.beispiel.api;   // Dieses Paket dürfen andere benutzen
      // Alle anderen Pakete bleiben versteckt, auch wenn Klassen public sind!
  }
  ```

- **Git Submodule:** Wenn ein Projekt aus mehreren Git-Repos besteht, kann man diese als Submodule einbinden; erlaubt die Bearbeitung als einzelnes Projekt in der IDE, obwohl es auf unterschiedlichen Repos gehostet wird
- **Docker und Microservices:** Jede Komponente als eigener Service; dies fördert die Isolation, erleichtert das Deployment und ermöglicht eine effiziente Skalierung

IDEs

IDEs unterstützen die Entwicklung in vielen verschiedenden System sowie autoamtische Task generierenung basierend auf den Config Dateien. Zudem erlauben IDEs heutztzutage auch einfach Code zu spliten.

Sources

- Java-Projekt und Artefakte
- Möglichkeiten, ein Maven-Projekt zu modularisieren