OpenRewrite

Workshop

Simon Gartner × 3IT

11.12.2025

gepardec
simplify your business

Agenda

// Willkommen & Vorstellung
// Einleitung und Rezept ausführen
// Composite Recipes
// Vorbereitungen & Vorgehensweise
// Composite Recipes - Advanced
// Recipes schreiben
// Multi-Module Setups & Hilfsmittel
// Gängige Probleme beim EAP-8 Upgrade; Diskussion

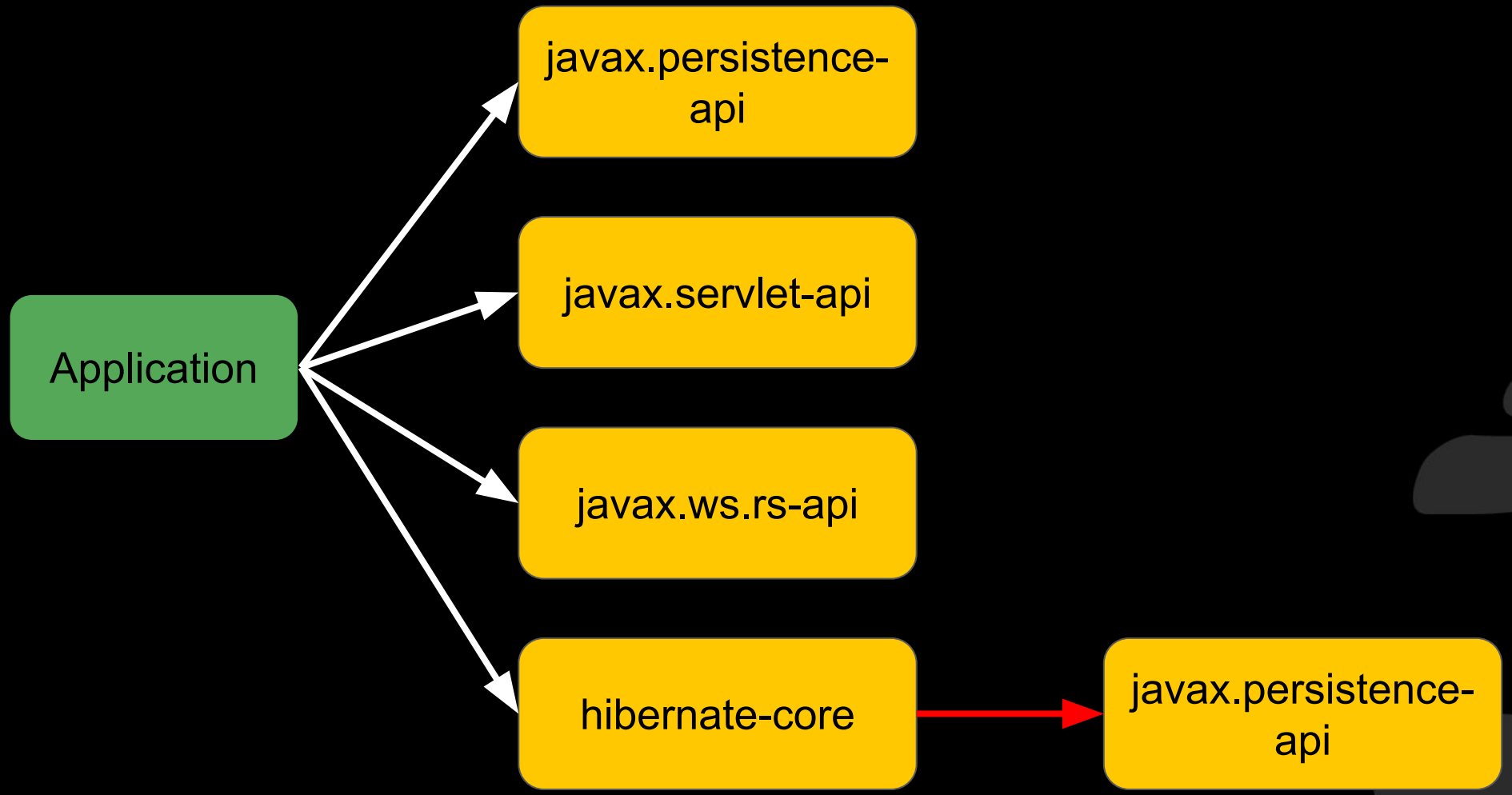
Das EAP-8 Upgrade

// **javax**.inject.Inject -> **jakarta**.inject.Inject

// **javax**.naming.InitialContext -> **javax**.naming.InitialContext

gepardec
simplify your business

Das EAP-8 Upgrade



gepardec
simplify your business

Manuelle Upgrades

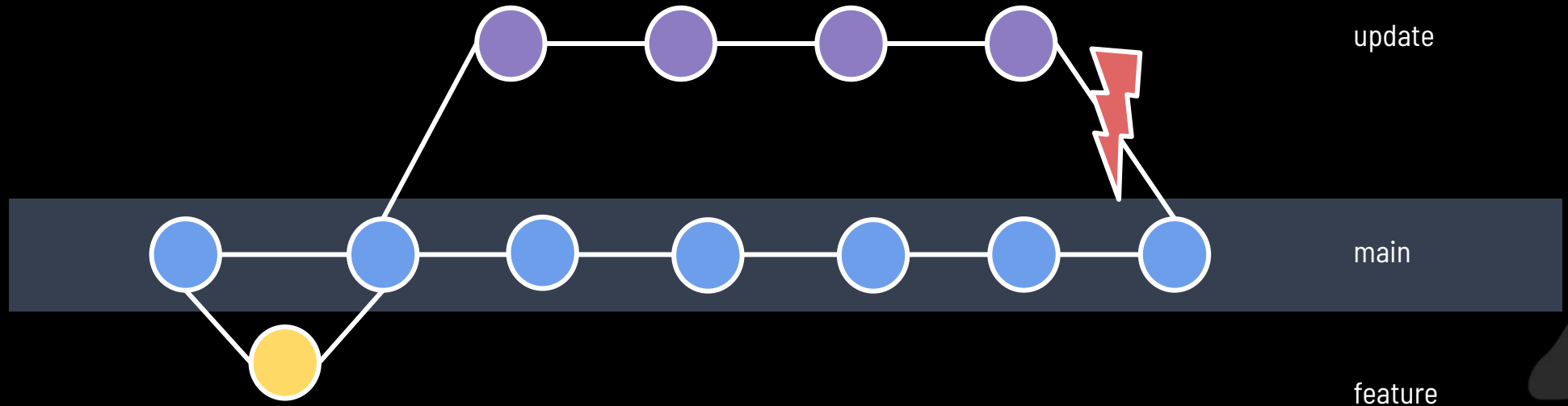
// Zeitintensiv

// Fehleranfällig

// Machen keinen Spaß

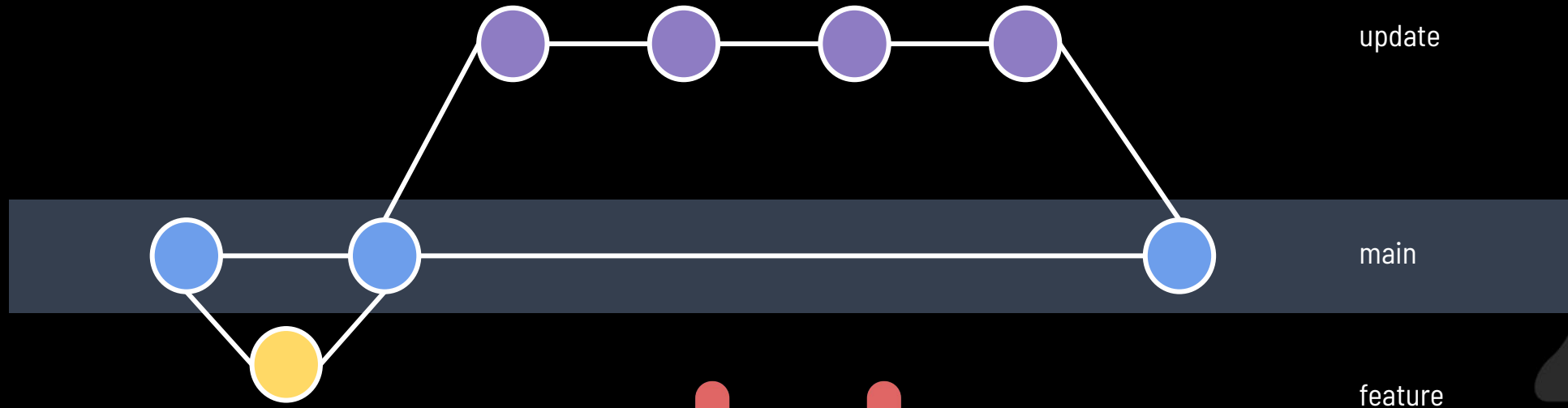
gepardec
simplify your business

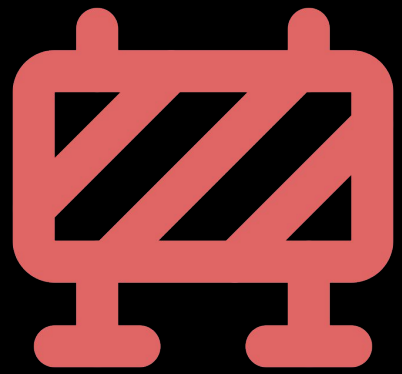
Manuelle Upgrades - Merge Conflict



gepardec
simplify your business

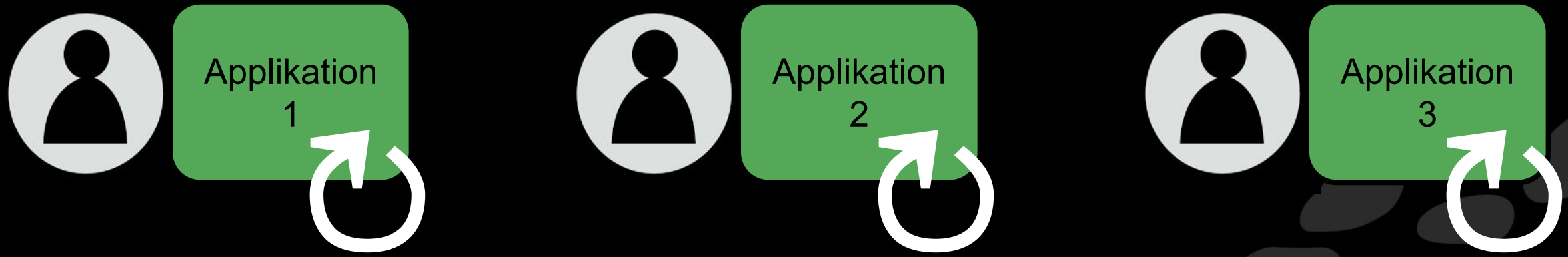
Manuelle Upgrades - Stop the World





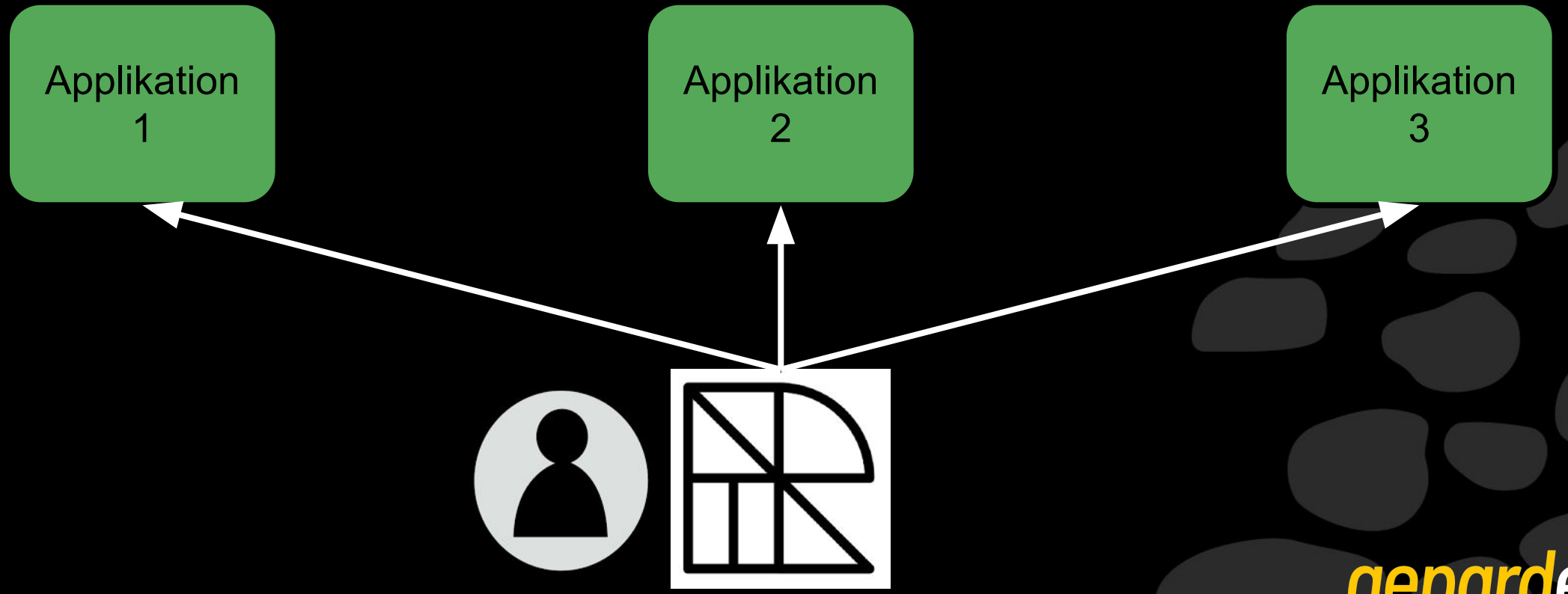
gepardec
simplify your business

Manuelle Upgrades - Repetetiv



gepardec
simplify your business

Und so wurde OpenRewrite geboren!



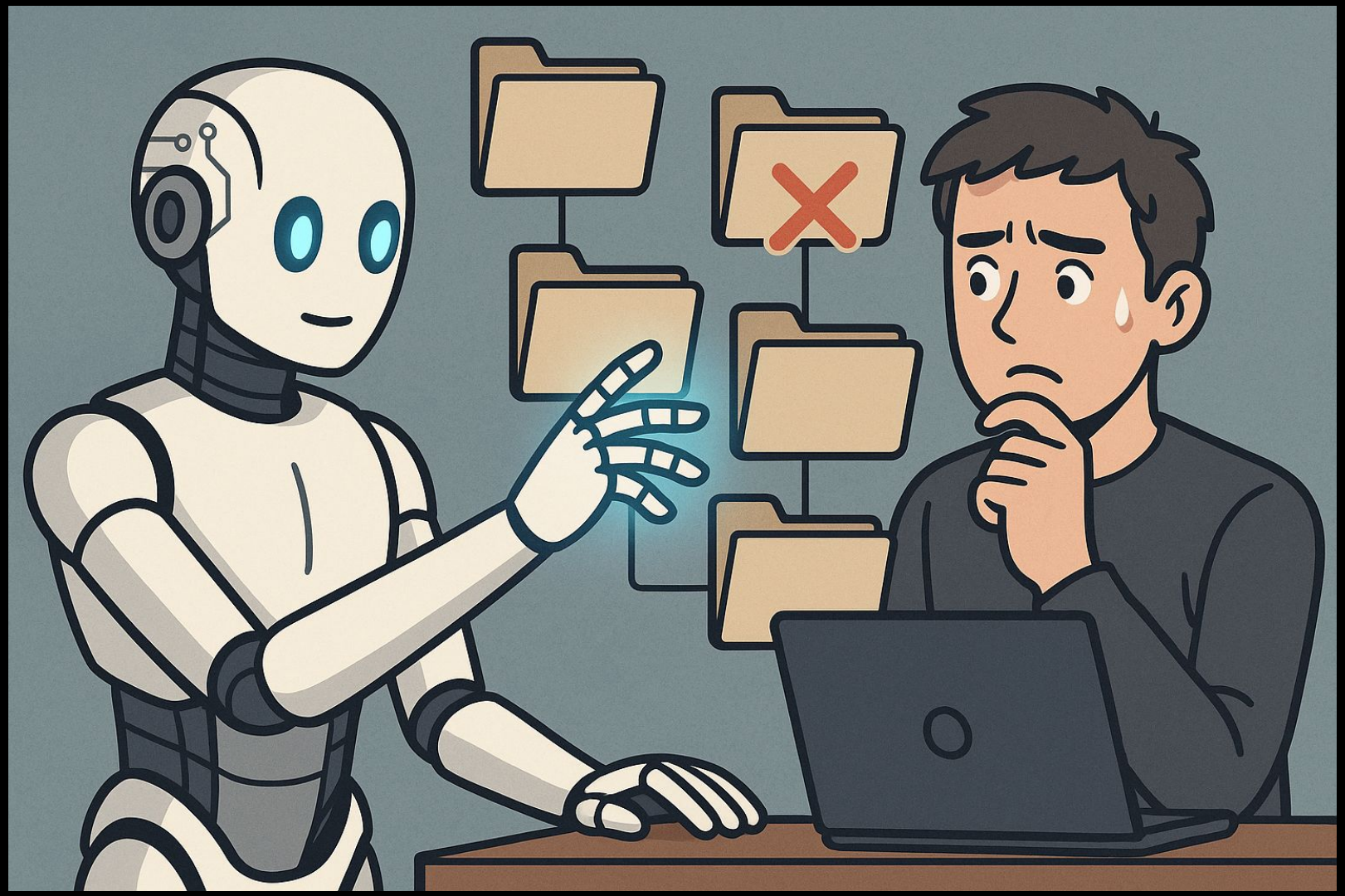
gepardec
simplify your business

Alternative: Find and Replace

```
public void example() {
  Type x = new Type();
  x.doSomething();
  OtherType y = new OtherType();
  y.doSomething();
}
```

```
public void example() {
  Type x = new Type();
  x.doNothing();
  OtherType y = new OtherType();
  y.doNothing();
}
```

Alternative: (Reine) LLMs



gepardec
simplify your business

Was ist ein Rezept?

// Grundeinheit von OpenRewrite

// Teilaufgabe einer Migration

// Kleine "Grundbausteine" <-> große Frameworkupgrades

// Schreibend und/oder Lesend

Rezepte

// Grundeinheit von OpenRewrite

https://docs.openrewrite.org/recipes

gepardec
simplify your business

Weitere Eigenschaften von Rezepten

// Formatübergreifende Operationen möglich
    // Z.B. mit Informationen aus einer XML-Datei Java-Code anpassen

// Div. Formate unterstützt
    // Support entwickelt sich laufend weiter

Unterstützte Formate

// Java

// XML / Maven

// Groovy / Gradle

// JSON

// YAML

// Properties

// ...

gepardec

simplify your business

Recipes anwenden

gepardec
simplify your business

rewrite-maven-plugin

```
<plugin>
    <groupId>org.openrewrite.maven</groupId>
    <artifactId>rewrite-maven-plugin</artifactId>
    <version>6.25.0</version>
    <configuration>
        <activeRecipes>
            <recipe>com.gepardec.EAP8Upgrade</recipe>
        </activeRecipes>
    </configuration>
    <dependencies>
        <dependency>
            <groupId>com.gepardec</groupId>
            <artifactId>recipe-collection</artifactId>
            <version>1.0.0</version>
        </dependency>
    </dependencies>
</plugin>
```

Ausführungsmodi

	Direktes Schreiben	Patch-File
Erbt Maven-Prozess	`run`	`dryRun`
Nur Compile	`runNoFork`	`dryRunNoFork`

Nach Jakarta EE 10 migrieren

`org.openrewrite.java.migrate.jakarta.JakartaEE10`

https://docs.openrewrite.org/recipes/java/migrate/jakarta/jakartaee10

gepardec
simplify your business

Übung #1 Ausführen eines Rezepts

// Maven-Projekt “ticket-monster”
    // Demo-Applikation mit JEE
    // Verwendet noch javax.* Namespaces
// TODO: Auf Jakarta EE 10 Migrieren
    // Mittels rewrite-maven-plugin
    // Das JakartaEE10 Migration Rezept ausführen
    // rewrite:run und rewrite:dryRun

Arten von Rezepten

// Composite Recipe (Declarative)
    // YAML-Datei mit Auflistungen an anderen Rezepten

// Refaster Templates: Java + Annotation Processing

// Custom Recipe (Imperative): Java

Composite Recipe

// YAML-File bestehend aus Rezeptdeklarationen

// Spätestens benötigt, wenn Argumente übergeben werden müssen

// Default-Pfad /rewrite.yml
    // oder /src/main/resources/META-INF/rewrite/*.yml
    // Alternativ mit der Plugin-Konfiguration "configLocation"

YAML-Schema

```yaml
type: specs.openrewrite.org/v1beta/recipe
name: com.gepardec.EAP8Upgrade
displayName: Migrations for EAP-8 Upgrade
recipeList:
    - org.openrewrite.maven.ChangeParentPom:
          oldGroupId: com.gepardec.parent
          oldArtifactId: parent-javax
          newArtifactId: parent-jakarta
          newVersion: 2025.12.0
...
```

Dependency Version erhöhen

```
- org.openrewrite.maven.UpgradeDependencyVersion:
    groupId: org.apache.cxf
    artifactId: cxf-rt-ws-security
    newVersion: 4.1.2
```

https://docs.openrewrite.org/recipes/maven/upgradedependencyversion

Dependency ändern

```
- org.openrewrite.java.dependencies.ChangeDependency:
    oldGroupId: org.jboss.spec.javax.jms
    oldArtifactId: jboss-jms-api_2.0_spec
    newGroupId: jakarta.jms
    newArtifactId: jakarta.jms-api
    newVersion: latest.release
```

https://docs.openrewrite.org/recipes/java/dependencies/changedependency

Maven Property ändern

```
- org.openrewrite.maven.ChangePropertyValue:
    key: version.infinispan
    newValue: 16.0.2
```

https://docs.openrewrite.org/recipes/maven/changepropertyvalue

gepardec
simplify your business

Java Klasse ändern

```
- org.openrewrite.java.ChangeType:
    oldFullyQualifiedTypeName: org.hibernate.Query
    newFullyQualifiedTypeName: org.hibernate.query.Query
```

https://docs.openrewrite.org/recipes/java/changetype

gepardec
simplify your business

Java Methode umbenennen

```
- org.openrewrite.java.ChangeMethodName:
  methodPattern: >-
    org.reflections.util.FilterBuilder include(..)
  newMethodName: includePattern
```

https://docs.openrewrite.org/recipes/java/changemethodname
https://docs.openrewrite.org/reference/method-patterns

gepardec
simplify your business

Find and Replace

```
- org.openrewrite.text.FindAndReplace:
    find: javax.xml.ws.client.receiveTimeout
    replace: jakarta.xml.ws.client.receiveTimeout
    filePattern: '**/*.java'
```

https://docs.openrewrite.org/recipes/text/findandreplace

Übung #2 Composite Recipe

// Unit-Test "CompositeRecipeExampleTest"
// Tipp: Aufteilen der Migration in 3 Aufgaben:
    // HelloWorldPrinter mit GepardecPrinter ersetzen
    // Dependency com.ext:printer zu com.gepardec.printer:1.0.0 umändern
    // Dependency com.gepardec:printer:1.0.0 ins DependencyManagement übernehmen
// Pro Aufgabe gibt es ein passendes Rezept ;)

Gepardec Rezepte

// Eigene, interne Rezeptsammlung
// https://github.com/Gepardec/openrewrite-recipes

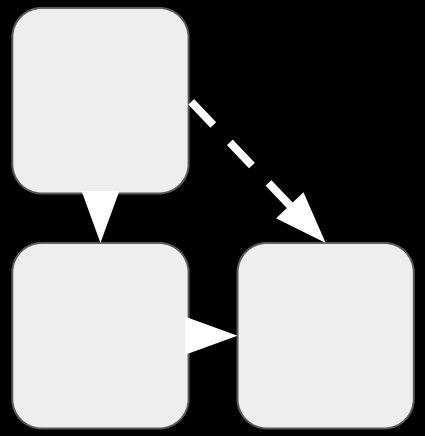
// `com.gepardec.common.recipes.maven.dependencies.MakeManagedVersionExplicit`
// `com.gepardec.common.recipes.maven.dependencies.AddDeclarationsOfUsedTransitiveDependencies`
// `com.gepardec.common.recipes.maven.modules.AddDependencyToModule`

gepardec
simplify your business

Vorbereitungen

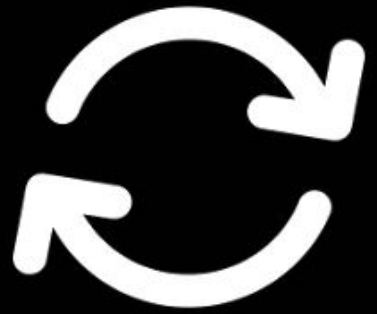
gepardec
simplify your business

Vorbereitungen



// Transitive Dependencies

// Gute Testabdeckung



// Automatisierung unterstützen

gepardec
simplify your business

Transitive Dependencies sichtbar

```
[INFO] --- dependency:3.8.1:analyze-only (default-cli) @ util ---
[WARNING] Used undeclared dependencies found:
[WARNING]    jakarta.inject:jakarta.inject-api:jar:2.0.1:provided
[WARNING] Unused declared dependencies found:
[WARNING]    org.slf4j:slf4j-ext:jar:2.0.17:compile
```

gepardec
simplify your business

Saubere Trennung von Modulen

// `mvn dependency:tree`

// Längerfristiges Sicherstellen?
    // ArchUnit
    // maven-enforcer-plugin

gepardec
simplify your business

DataTables

// Manche Rezepte erstellen tabellarische Daten

// z.B. Usages von Methoden, Git-Metadaten, uvm.

// Werden pro Ausführung mit timestamp unter target/rewrite abgelegt

// Aktivieren mit <exportDataTables>true</exportDataTables>

Standard-DataTables

// Standardmäßig exponiert ein Rezept 2-3 DataTables:

// SourceFileResults: Modifizierte Dateien

// RecipeRunStats: Perfomancestatistiken
    // Welches Rezept hat wie lange gebraucht?

// SourceFileErrors: Falls Fehler vorhanden, wo sind sie aufgetreten?

Übung #3: DataTables

// Aus der ticket-monster Applikation soll ein DataTable erzeugt werden

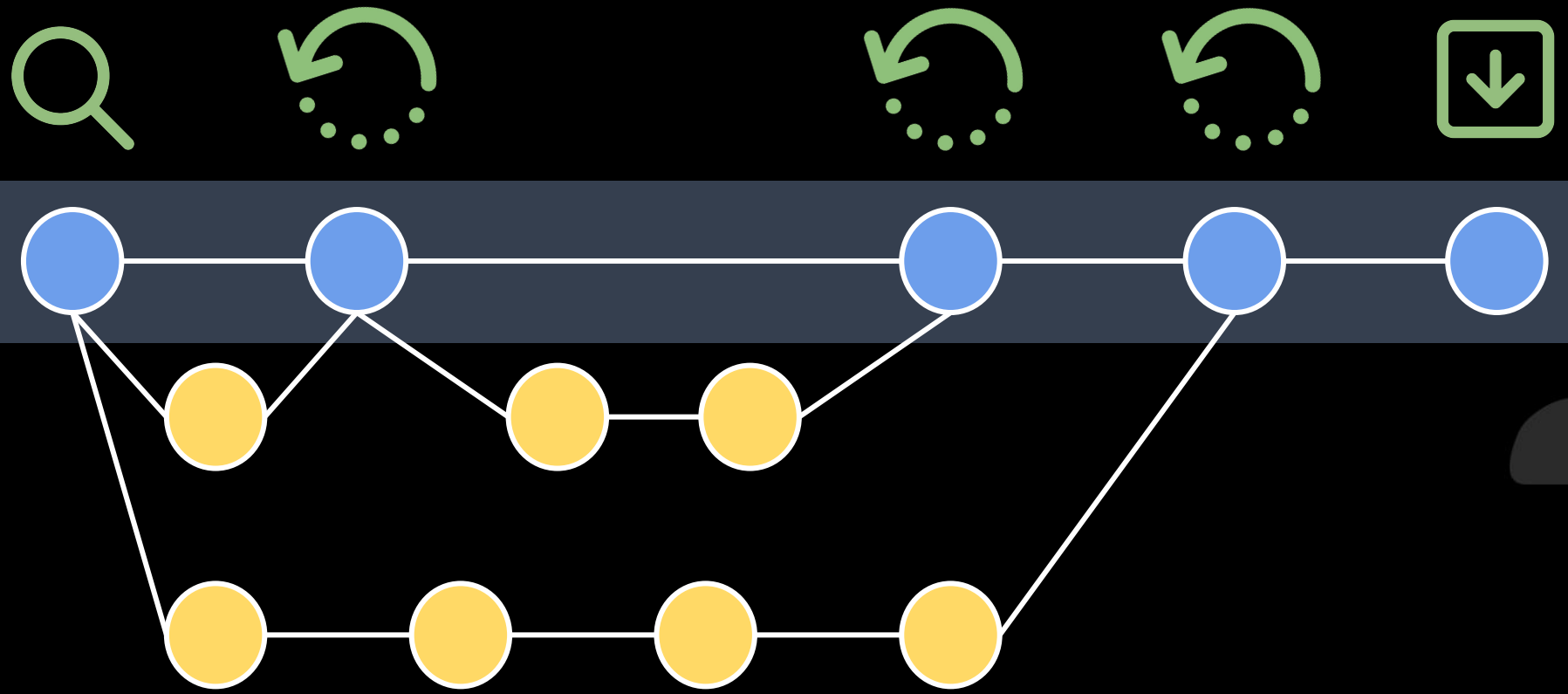
// DataTables im rewrite-maven-plugin aktivieren

// org.openrewrite.java.dependencies.RelocatedDependencyCheck

Vorgehensweise

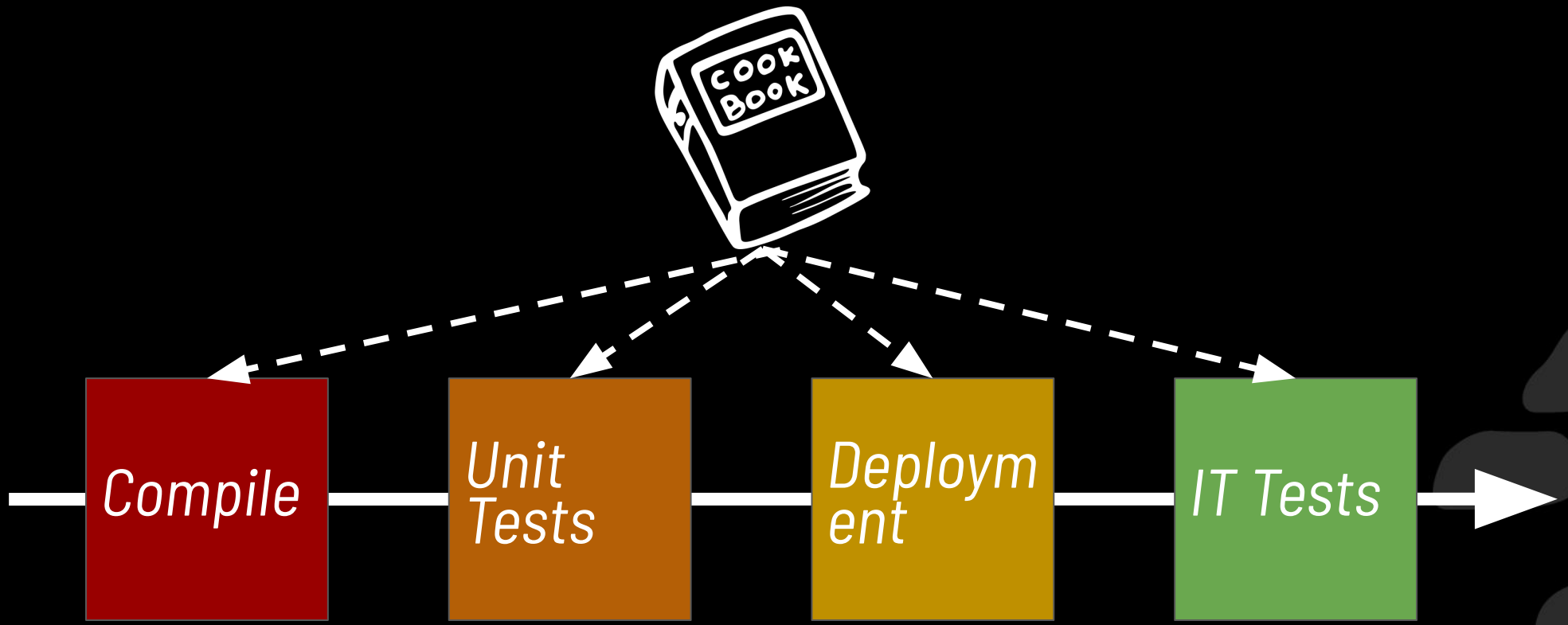
gepardec
simplify your business

Transformer



gepardec
simplify your business

Meilensteine



gepardec
simplify your business

Einzelfälle

Q: Sollte ich spezifische Fälle, die nur einzeln Vorkommen, in mein Rezept einbinden?

A: Ja, aber halte es simpel!

- *FindAndReplace*
- *Replace entire files*

Recipes anwenden (Advanced)

gepardec
simplify your business

Preconditions

```
type: specs.openrewrite.org/v1beta/recipe
name: com.gepardec.RemoveHibernateEntityManagerFromCommons
displayName: Removes Hibernate EntityManager in commons module
preconditions:
   - com.gepardec.common.recipes.maven.modules.FindModule:
        groupId: com.gepardec
        artifactId: commons
recipeList:
   - org.openrewrite.maven.RemoveDependency:
        groupId: org.hibernate
        artifactId: hibernate-entitymanager
```

Preconditions OR

```
type: specs.openrewrite.org/v1beta/recipe
name: com.gepardec.SelectCommonsAndService
displayName: Removes Hibernate EntityManager in commons module
recipeList:
   - com.gepardec.common.recipes.maven.modules.FindModule:
        groupId: com.gepardec
        artifactId: commons
   - com.gepardec.common.recipes.maven.modules.FindModule:
        groupId: com.gepardec
        artifactId: service
```

Gliederung von Composite Recipes

// Mehrere Rezepte innerhalb eines YAML-Files umsetzbar

// Gliederung z.B. nach:
    // Phase
    // Technologie
    // Precondition

Gliederung von Composite Recipes

```yaml
recipeList:
    - org.openrewrite.java.migrate.jakarta.JakartaEE10
    - com.gepardec.eap8.Build
    - com.gepardec.eap8.Tests
    - com.gepardec.eap8.Deployment
    - com.gepardec.eap8.IntegrationTests
    - com.gepardec.eap8.TextReplacements

---
type: specs.openrewrite.org/v1beta/recipe
name: at.sozvers.stp.lgkk.eap8.Build
displayName: Migrations for a successful build
recipeList: ...
```

Cycles

// Mehrere Zyklen möglich (Rezept wird öfters ausgeführt!)

// Bis sich die Ausführung stabilisiert

// canCauseAnotherCycle: true

// Es wird stark davon abgeraten -> Performance, Fehleranfälligkeit

Übung #4: Preconditions

// Zugehöriger Unit-Test "PreconditionsExampleTest"
// jakarta.inject:jakarta.inject-api:2.0.1:provided hinzufügen
// Aber nicht wenn jakarta.jakartaee-api
oder jakarta.jakartaee-core-api vorhanden sind!

Recipes schreiben

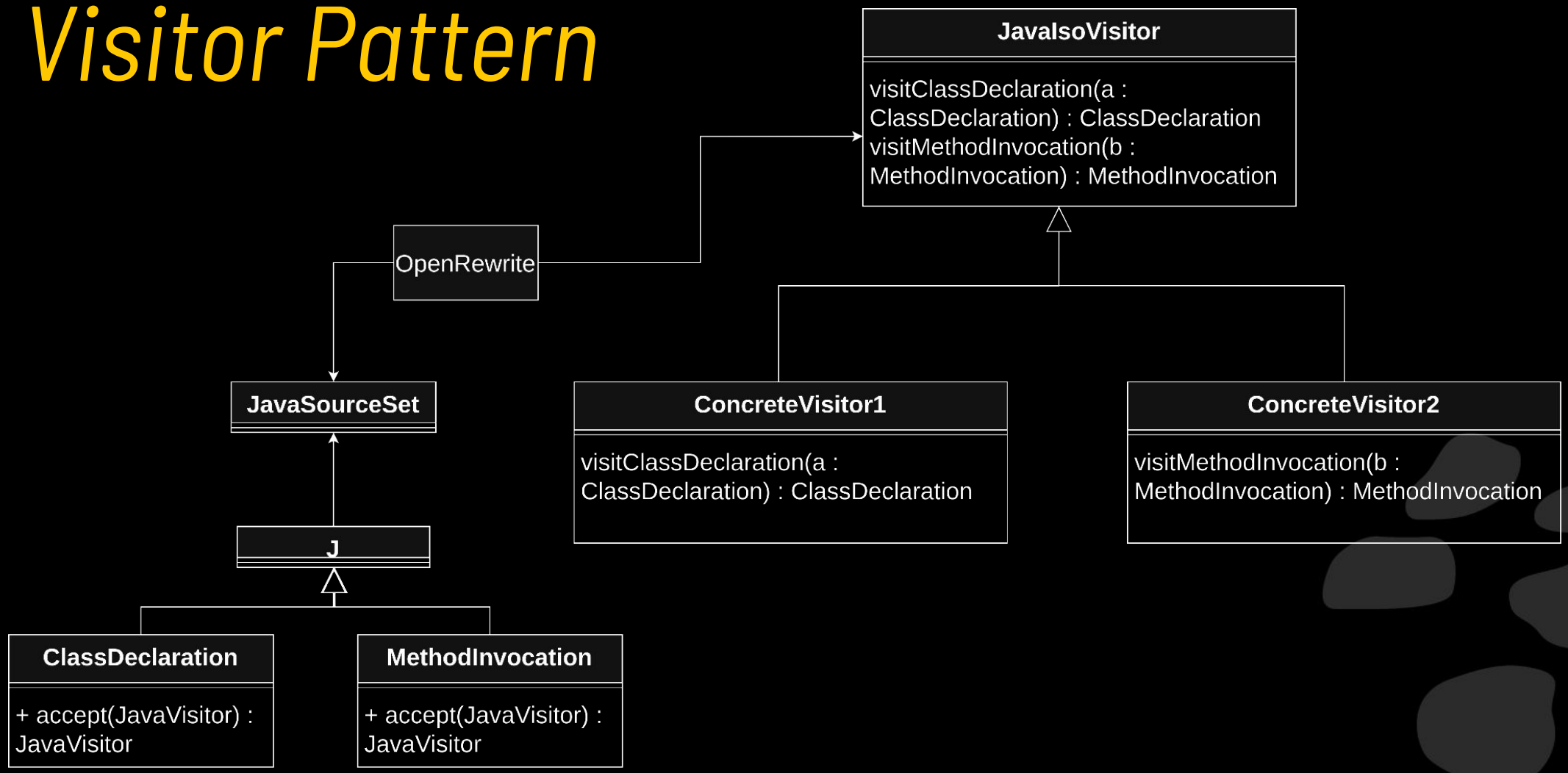
gepardec
simplify your business

Lossless Semantics Tree (LST)

```
\--J.ClassDeclaration
   |--J.Modifier | "public"
   |--J.Identifier | "Main"
   \--J.Block
      \----J.MethodDeclaration | "MethodDeclaration{com.gepardec.Main}"
           |--J.Modifier | "public"
           |--J.Modifier | "static"
           |--J.Primitive | "void"
           |--J.Identifier | "main"
           |------J.VariableDeclarations | "String[ ] args"
           \--J.Block
              \----J.MethodInvocation | "System.out.println("Hello world!")"
                   |----J.FieldAccess | "System.out"
                   |--J.Identifier | "println"
                   \------J.Literal | ""Hello world!""
```

gepardec
simplify your business

Visitor Pattern



gepardec
simplify your business

Rezepte schreiben - Ressourcen

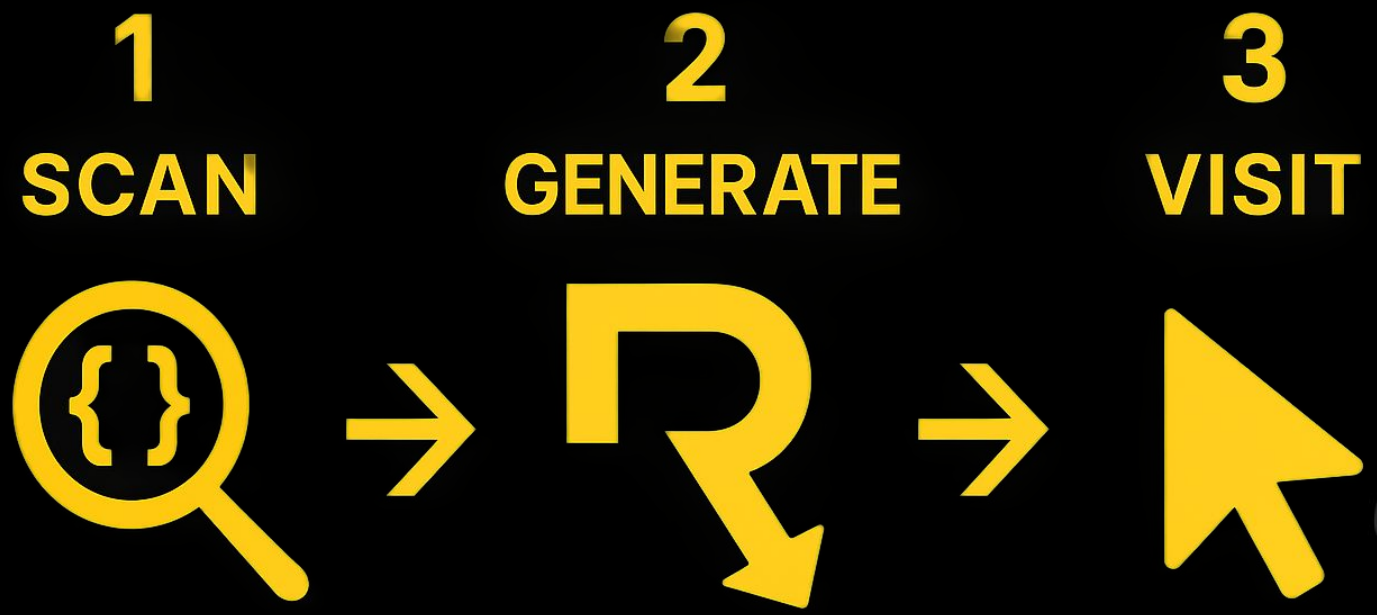
// https://docs.openrewrite.org/authoring-recipes

// https://docs.moderne.io/user-documentation/community-office-hours/

// https://app.moderne.io/recipes/org.openrewrite.java.search.FindMethods

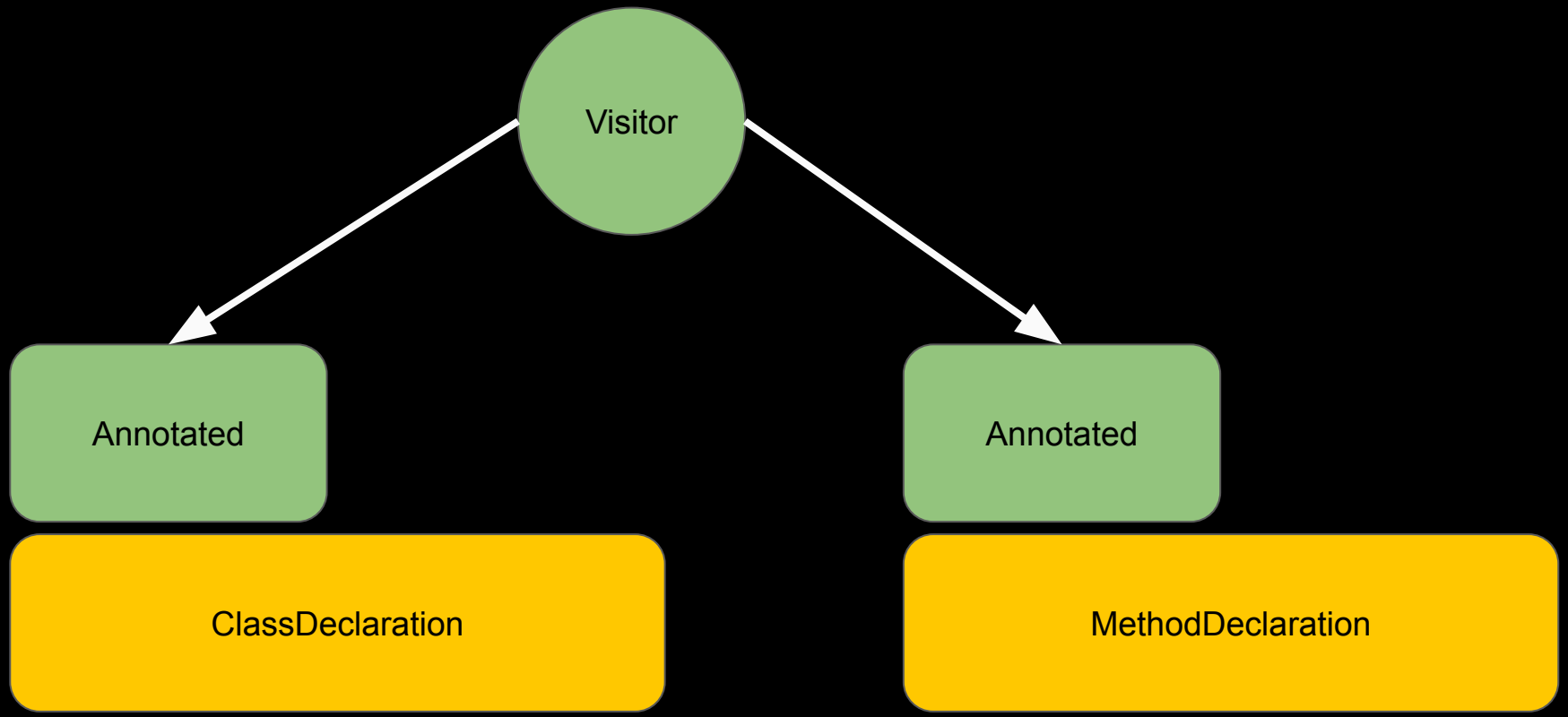
// https://join.slack.com/t/rewriteoss/shared_invite/zt-1ihfgqp2a-gllit_aXJnhHAdv_0uzwow

Scanning Recipe



gepardec
simplify your business

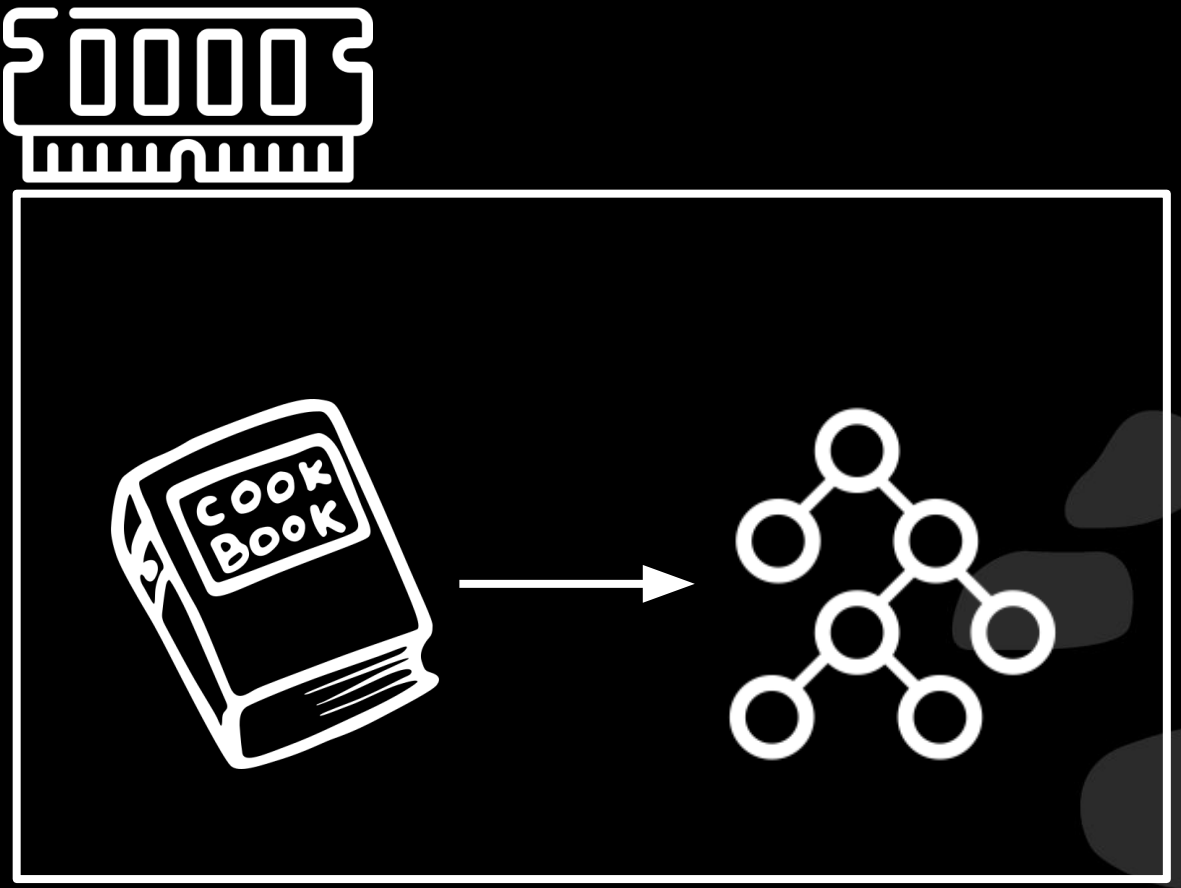
Traits



Multi-Module Setups

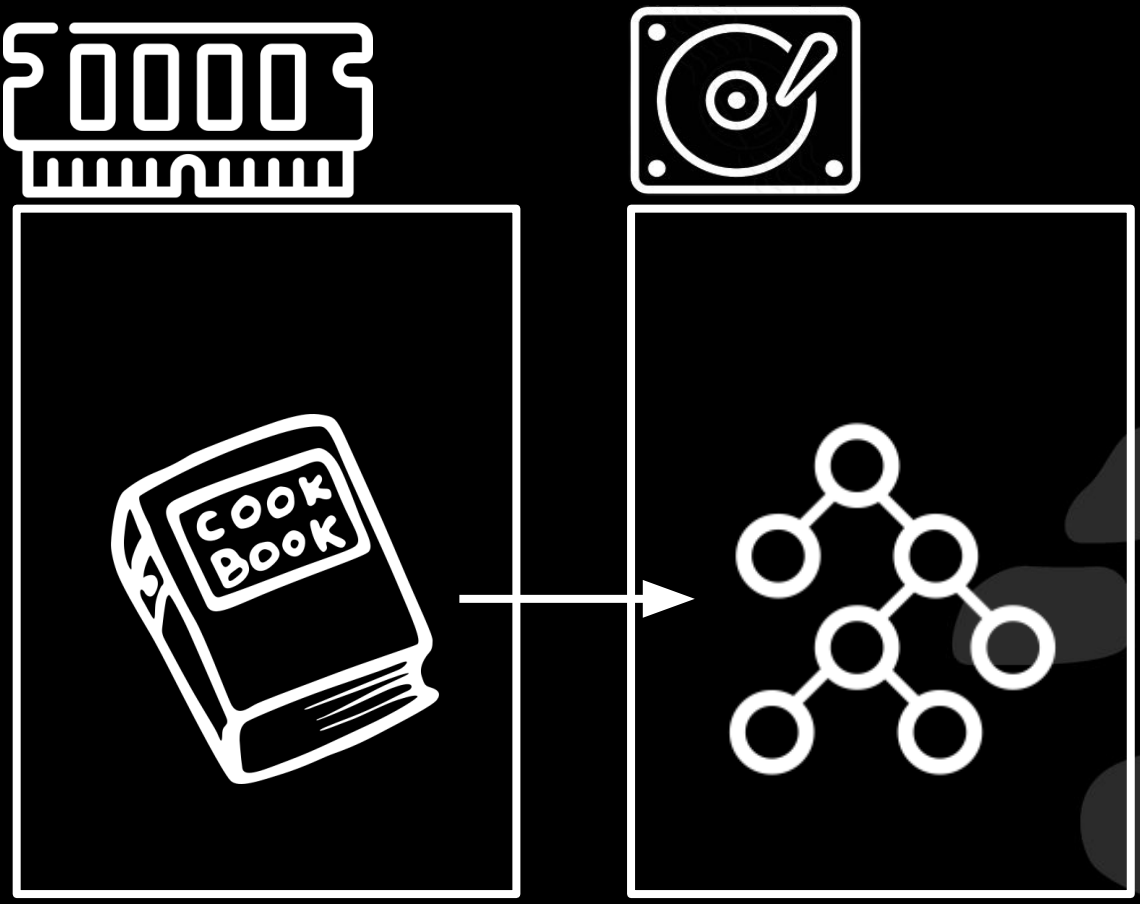
gepardec
simplify your business

OpenRewrite



gepardec
simplify your business

Moderne CLI



Iterationsdauer

// 2 Mio LOC ->
// rewrite:run 2h + OOM
// Moderne CLI <6h

// 500k LOC
// rewrite:run 10min
// rewrite:dryRun 15min

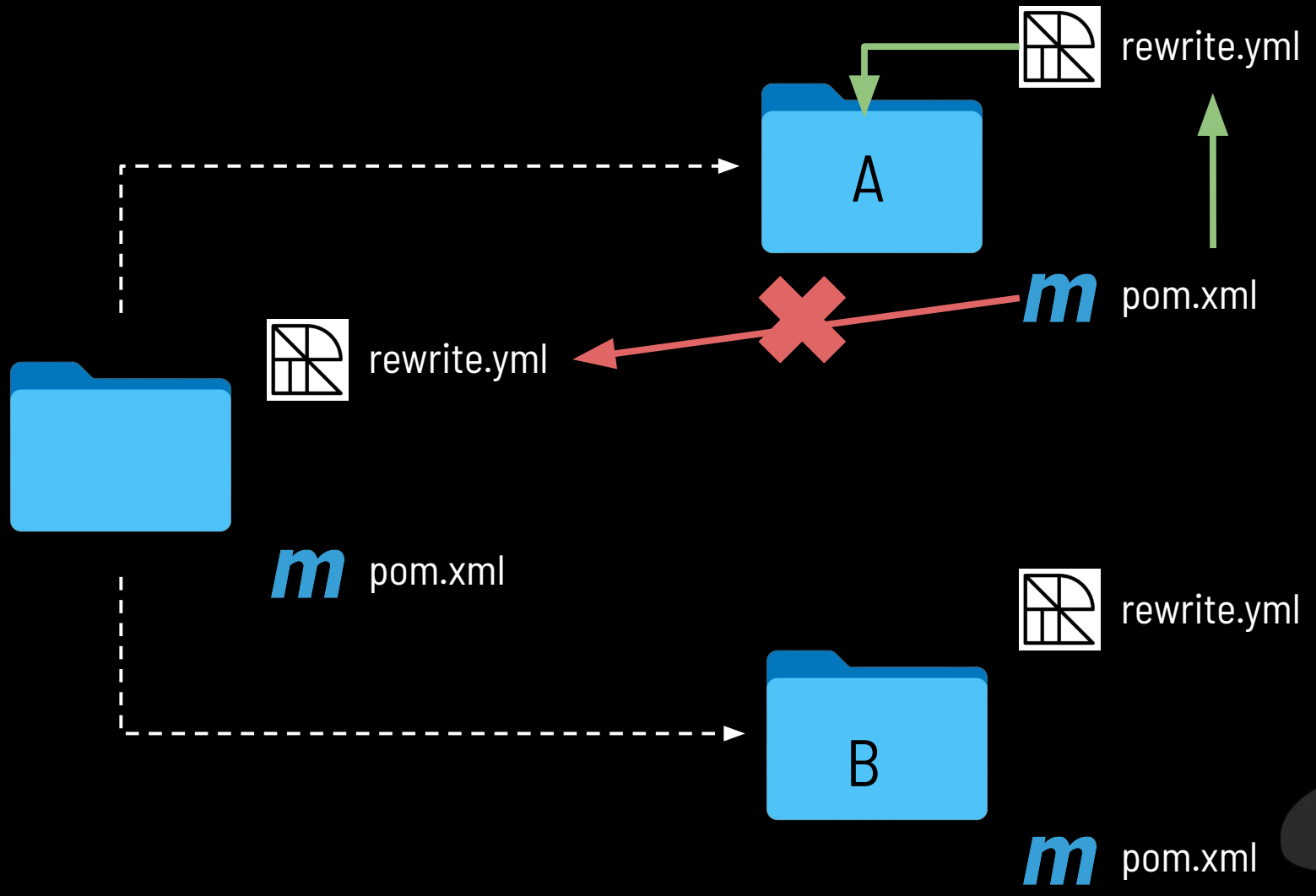
gepardec
simplify your business

Maven Modul-Selektoren

// -N: Nicht-rekursiv; Ignoriert POMs in Submodulen

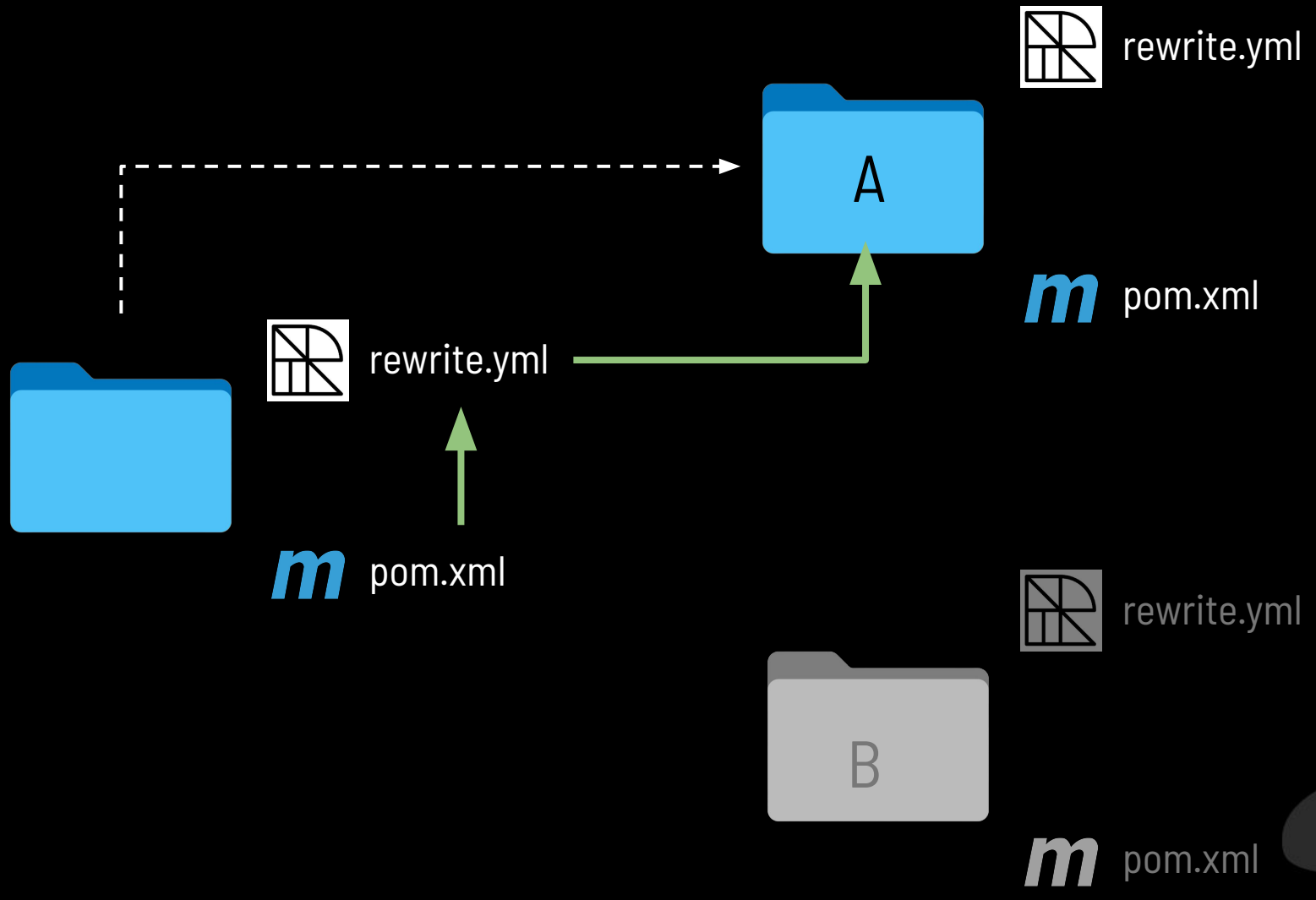
// -pl: Modulliste
// -am: Inkludiere Module, die von der Liste (-pl) benötigt werden
// -amd: Inkludiere Module, die Module der Liste (-pl) benötigen

rewrite.yml aus Parent nicht nutzbar



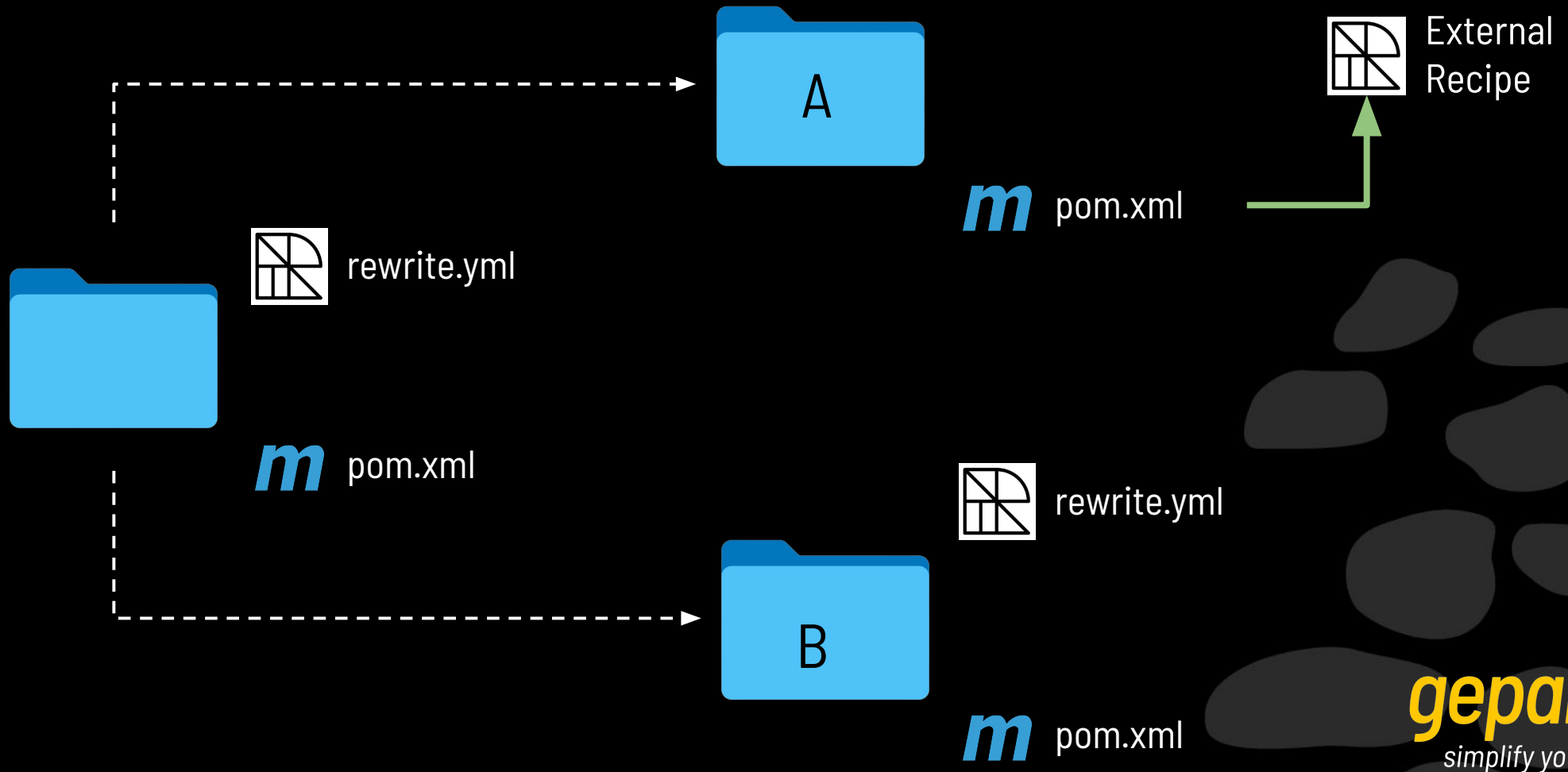
gepardec
simplify your business

-pl vom Parent aus



gepardec
simplify your business

Externes Rezept



gepardec
simplify your business

Nur Parent POM

```
<exclusions>
  <exclude>**/src/**</exclude>
  <exclude>**/*.xml</exclude>
  <exclude>**/*.java</exclude>
  <exclude>**/Jenkinsfile</exclude>
</exclusions>
```

```
-> mvn -N rewrite:run
```

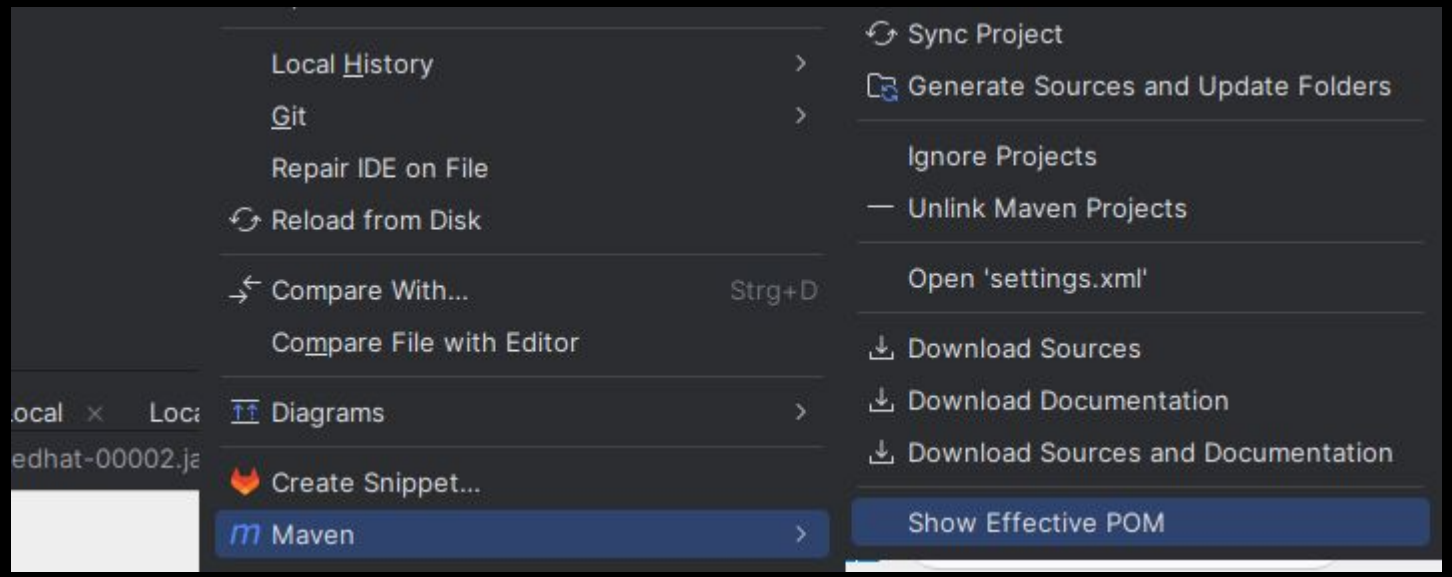
Hilfsmittel

gepardec
simplify your business

Effective POM

// Ist eine Dependency in diesem Modul?

// Welche Version hat diese Dependency wirklich?



Verbose Effective POM

// Von wo kommt dieses Feld im POM?

// Aus welchem BOM kommt diese Version?

```
mvn help:effective-pom -Dverbose
```

gepardec
simplify your business

Dependency Tree

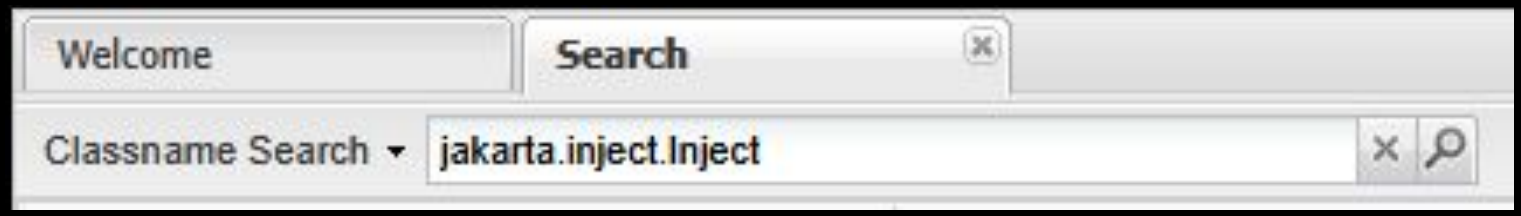
// Über welchen Weg erhalte ich eine transitive Dependency?

```
mvn dependency:tree
```

gepardec
simplify your business

Nexus Classname Search

// Woher kommt eine Klasse?
// -> ClassNotFoundException



gepardec
simplify your business

Übung #5: Ticket Monster fertig migrieren

// Das Rezept MigrateTicketMonsterAdvancedExample soll vervollständigt werden

// Jedes TODO ist eine Aufgabe (auch als Compile-Error sichtbar):
// Rezeptaufruf muss in ticket-monster eingebunden werden (Das uebungen-Modul ist daher eine Dependency des rewrite-plugins)
// Noch nicht alle Dependecies sind migriert
// Gewisse Imports funktionieren nicht mehr

Gängige Probleme beim EAP-8 Upgrade

HIBERNATE

`ClassNotFoundException`

`UnsatisfiedResolutionException`

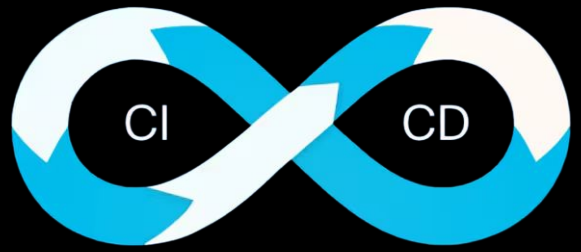
`IllegalAccessException` *(Java 17)*

JCache

`/subsystem=elytron`

Code-Generation

JSF



gepardec

simplify your business

gepardec

simplify your business