



# *OpenRewrite in a Nutshell*

*Scaling Upgrades with  
Practical Insights*

*Simon Gartner*

*02.06.2025*

**gepard**ec  
simplify your business

# Frameworks we've used



2

JUnit



1

**gepard**ec  
simplify your business

*Upgraded to...*



JAKARTA® EE



spring® 6



QUARKUS

3

JUnit



LOG4J



2

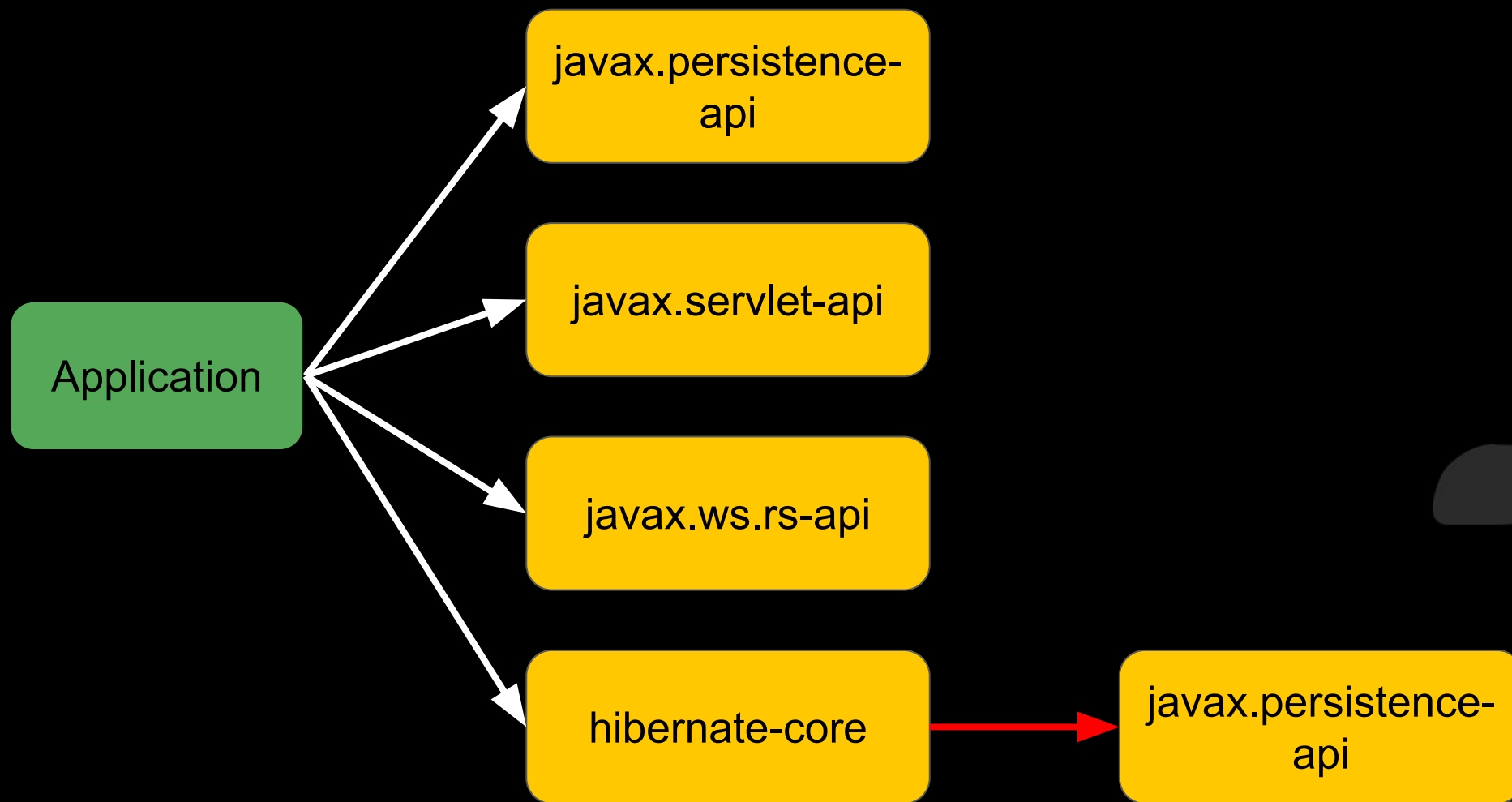
**gepard**ec  
simplify your business

# A large upgrade: Jakarta EE 9

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

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

# *A large upgrade: Jakarta EE 9*



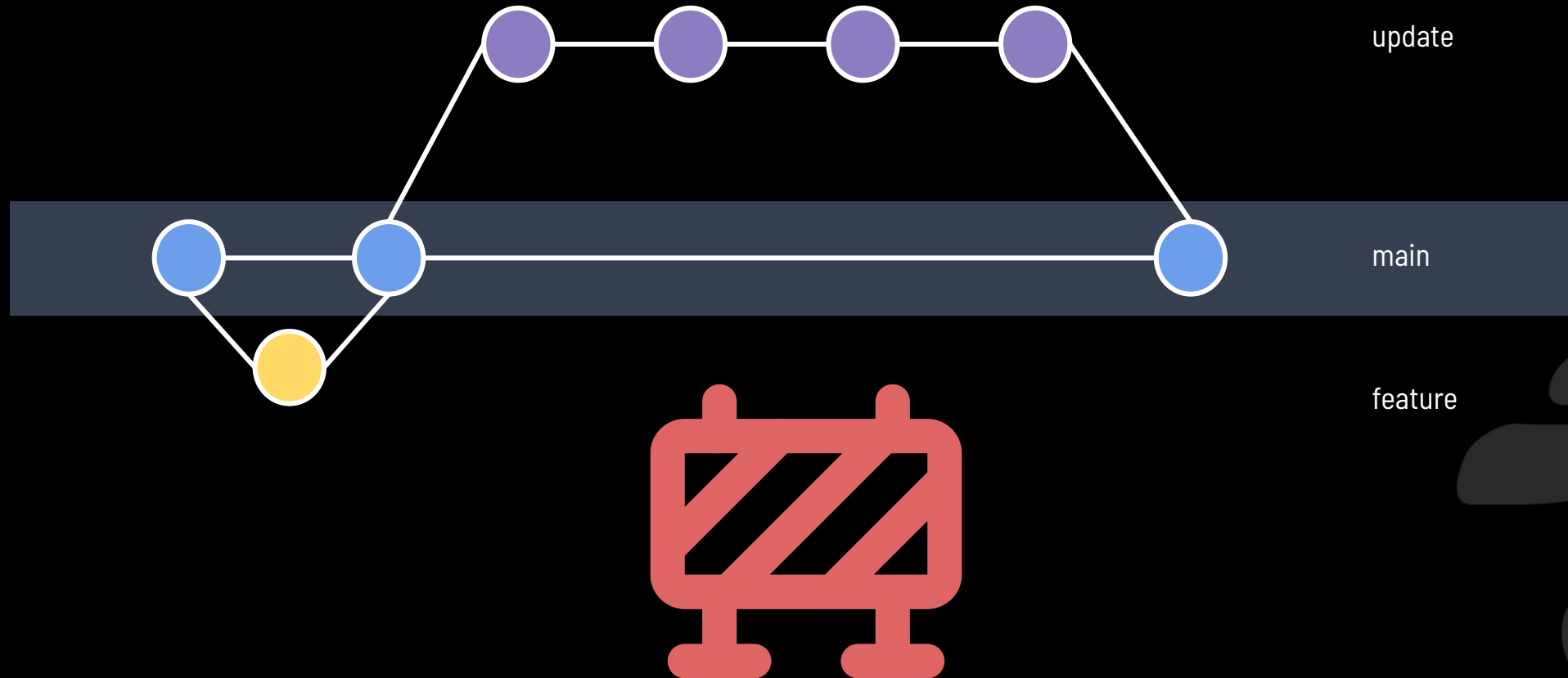
# *Upgrading manually - A tedious task*

*// Time consuming*

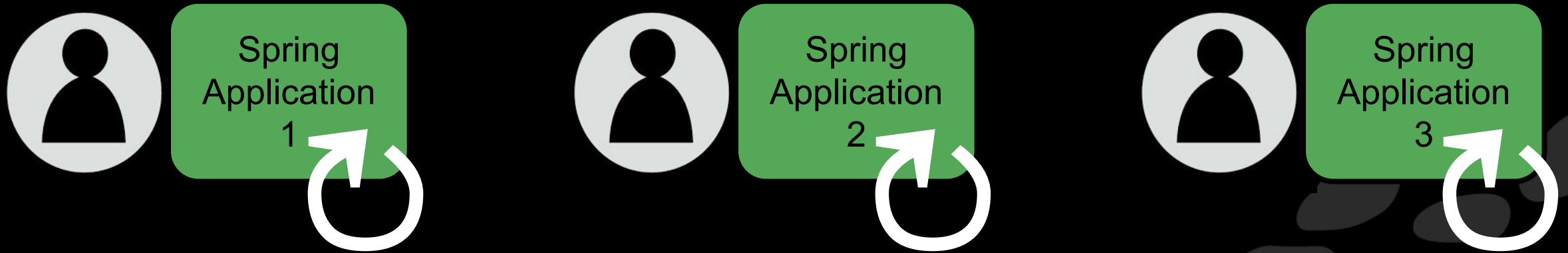
*// Error prone*

*// Not fun*

# Upgrading manually - Stop the world

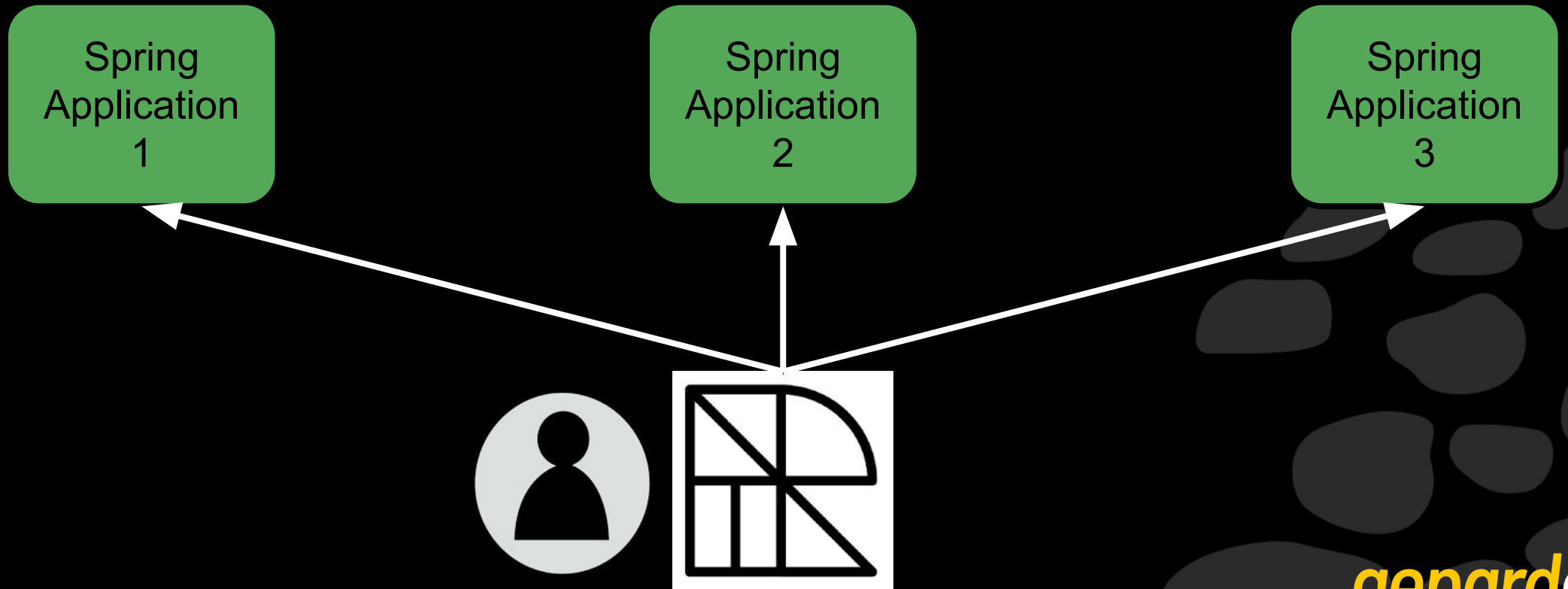


# *Upgrading manually - Repeated effort*





# *And that's how OpenRewrite got born!*



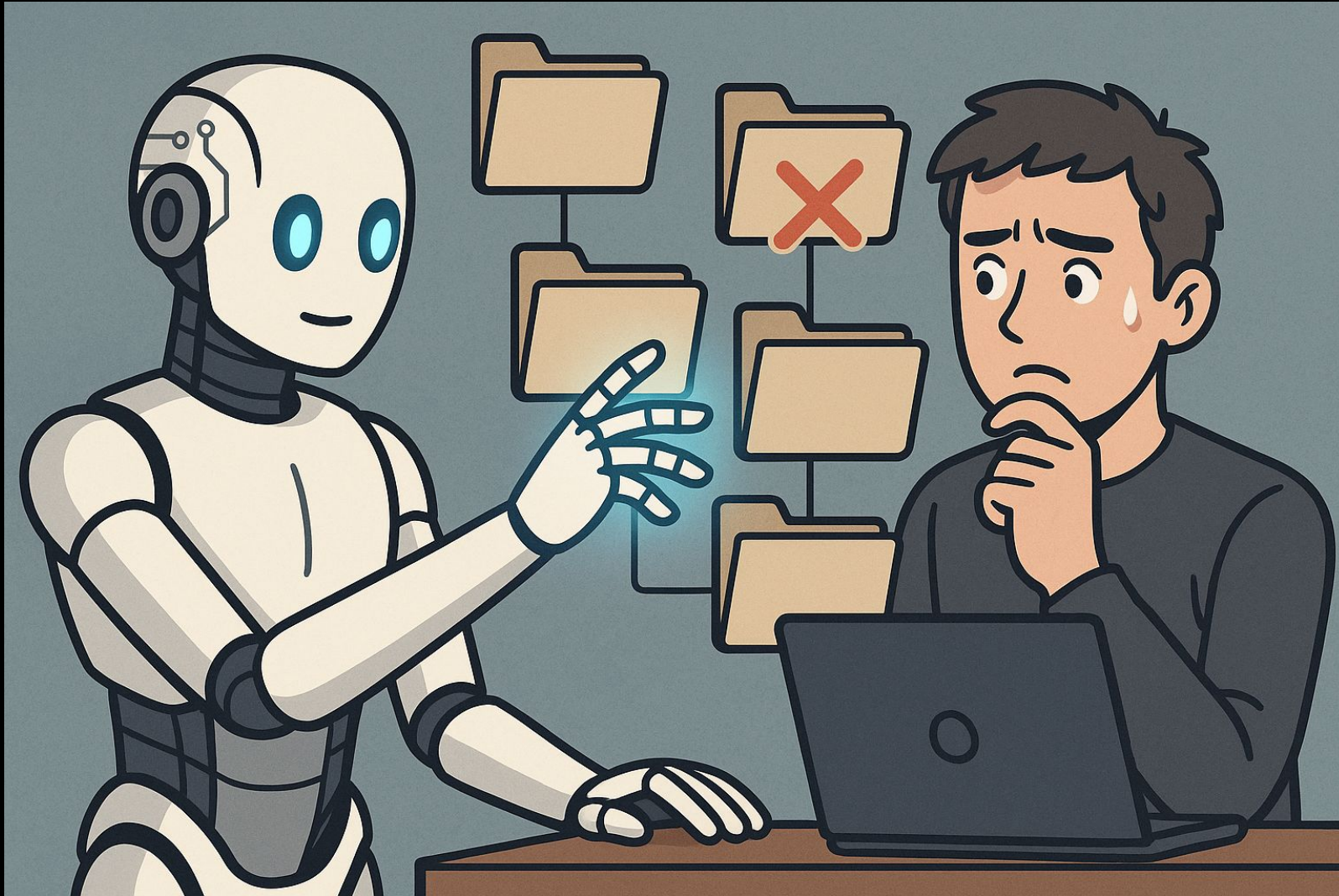
# About text replacements

```
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.doSomething();  
}
```

# About LLMs



# *Formats supported by OpenRewrite*

*// Java*

*// XML / Maven*

*// Groovy / Gradle*

*// JSON*

*// YAML*

*// Properties*

*// ...*

# Recipe Catalog

*// List of official Recipes by Moderne / Open Source*

<https://docs.openrewrite.org/recipes/java/migrate/jakarta/jakartaee11>





# *Composite Recipes*

# Another Recipe

[https://docs.openrewrite.org/recipes/maven/changedependencygroup  
idandartifactid](https://docs.openrewrite.org/recipes/maven/changedependencygroupidandartifactid)

# Other Recipes

// UpgradeDependency

// ChangePropertyValue

// ChangeType

// ReplaceAnnotation

// JakartaEE10

// Slf4jToLog4j





# *Impact analysis*

# DataTables

*// List of Recipes that produce DataTables:*

*<https://docs.openrewrite.org/reference/recipes-with-data-tables>*

*// (includes mutating recipes that also produce custom DataTables)*

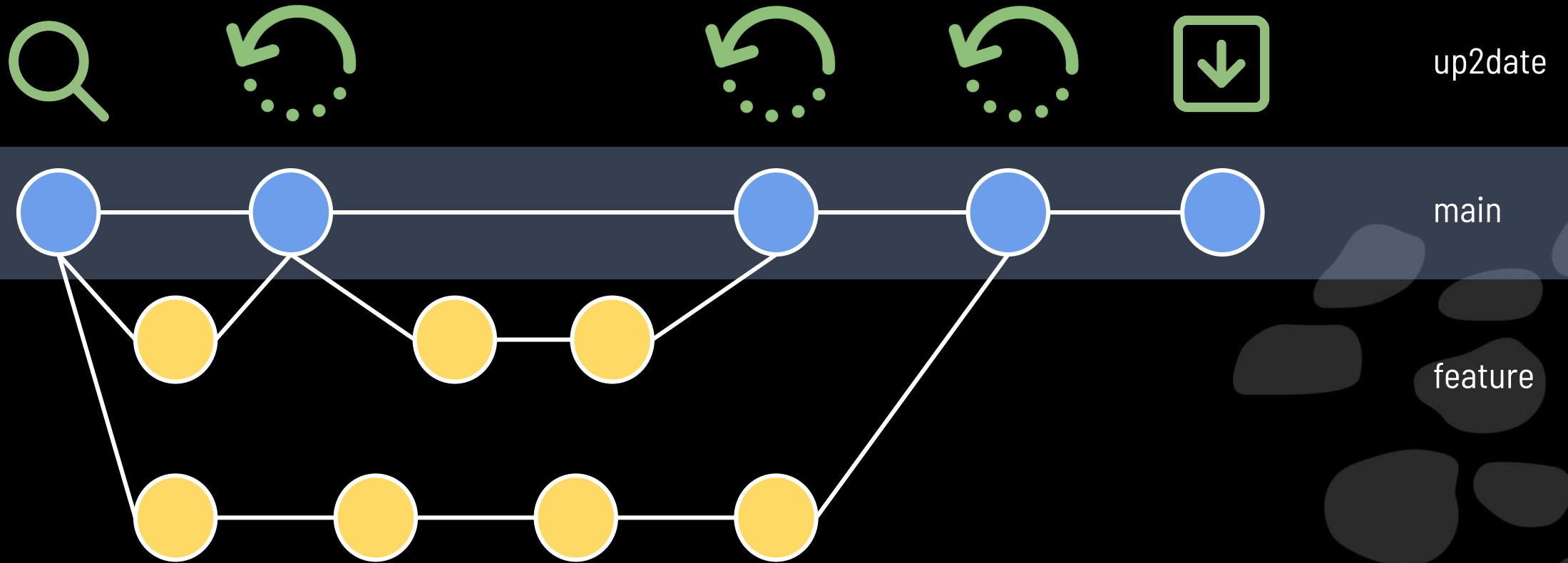
*// DataTables example:*

*<https://docs.openrewrite.org/recipes/core/languagecomposition>*

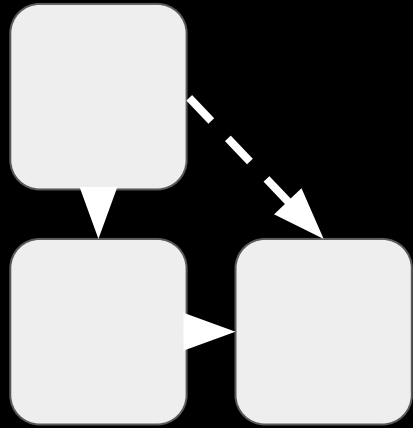


# *Framework Migrations*

# Transformer



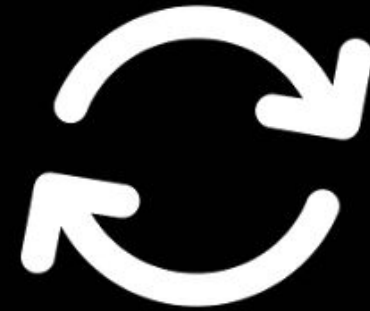
# *Prerequisites for Framework Migrations*



// Transitive dependencies in controlled code?

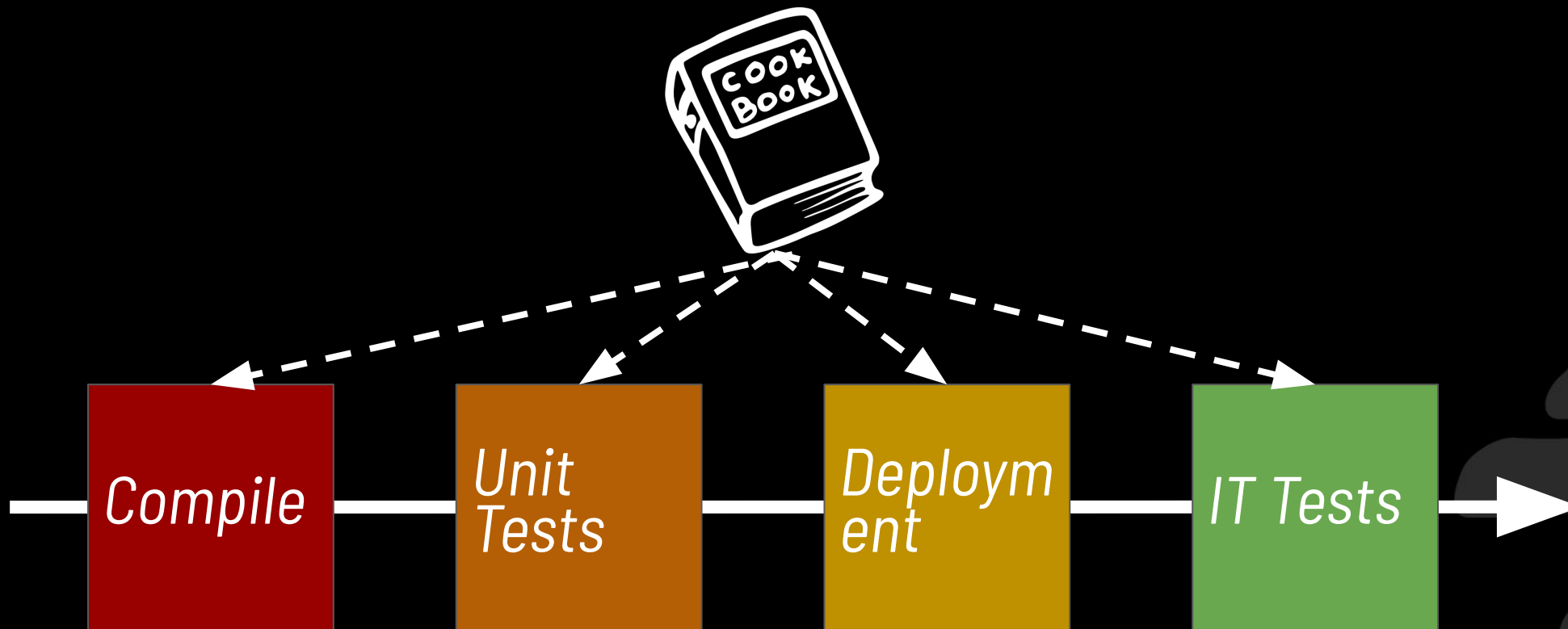


// Ensure a good test coverage



// Clean up to ease automation

# *Automating Framework Migrations*



# Single Cases

*Q: Should I make the effort to integrate specific cases even if they only have one occurrence?*

*A: Yes! But keep it simple! For example, you could do:*

- FindAndReplace*
- Replace entire files*





# *Custom Recipes*



# Interface transformation

```
request  
    .getMetadata()  
    .setTenantId("1");
```

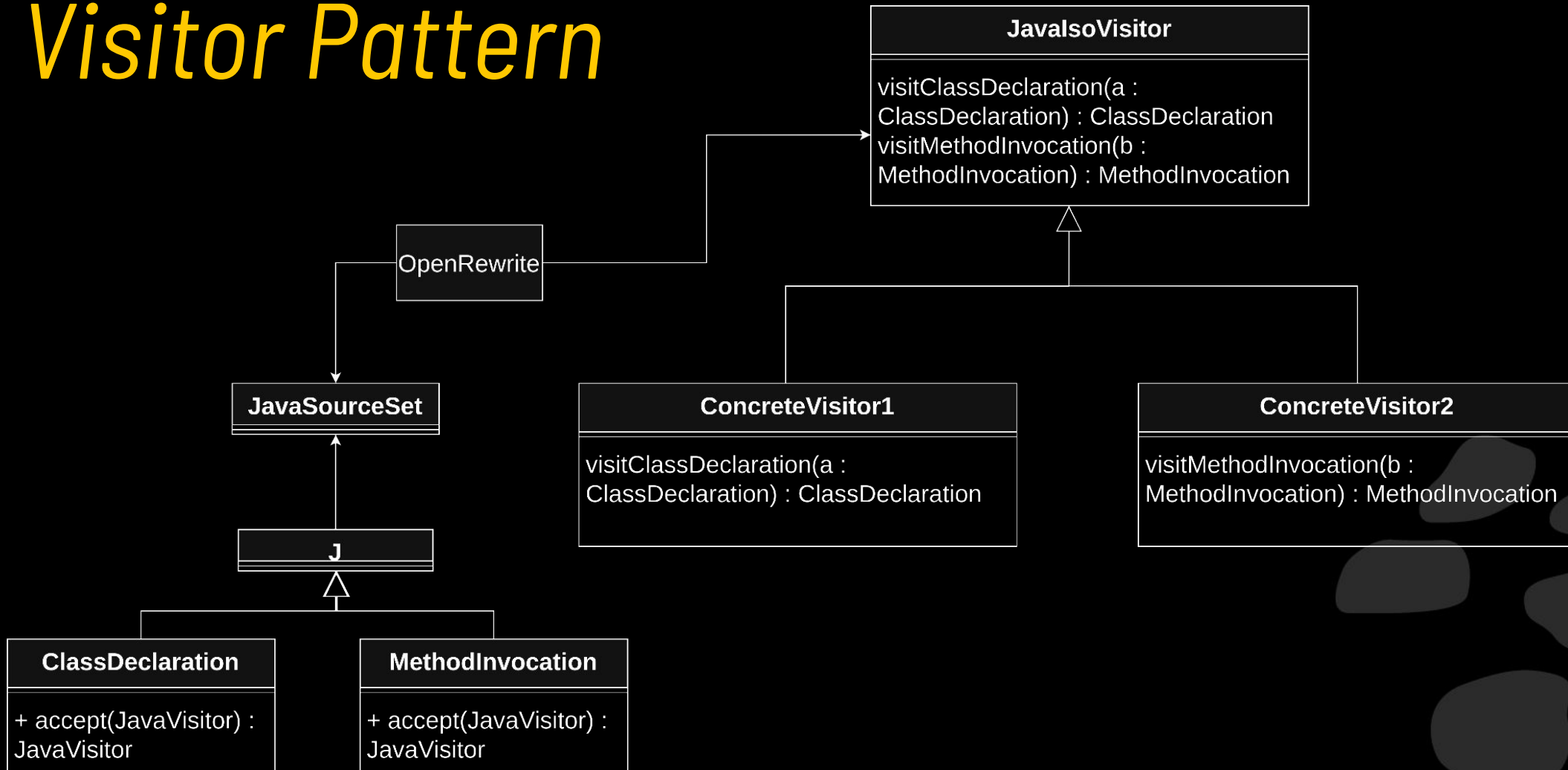
```
Metadata meta =  
    ObjectFactory.createMetadata();  
JaxbElement<String> tenantId =  
    ObjectFactory  
        .createMetadataTenantId("1");  
meta.setTenantId(tenantId);  
request.setMetadata(meta);
```



# 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!""
```

# Visitor Pattern





# *Example for custom recipes*

# Interface transformation

```
request  
    .getMetadata()  
    .setTenantId("1");
```

```
Metadata meta =  
    ObjectFactory.createMetadata();  
JaxbElement<String> tenantId =  
    ObjectFactory  
        .createMetadataTenantId("1");  
meta.setTenantId(tenantId);  
request.setMetadata(meta);
```



# Scanning Recipe

**1**  
**SCAN**



// Scan DTO  
hierarchies and store  
as Tree

**2**  
**GENERATE**



**3**  
**VISIT**



// Lookup DTO in  
tree and add  
initialisations +  
nested setters

# *Quality Criterias for Recipes*

*// Type attribution*

*// Idempotency*

*// Atomicity*

# *Learning to write Recipes*

// <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>

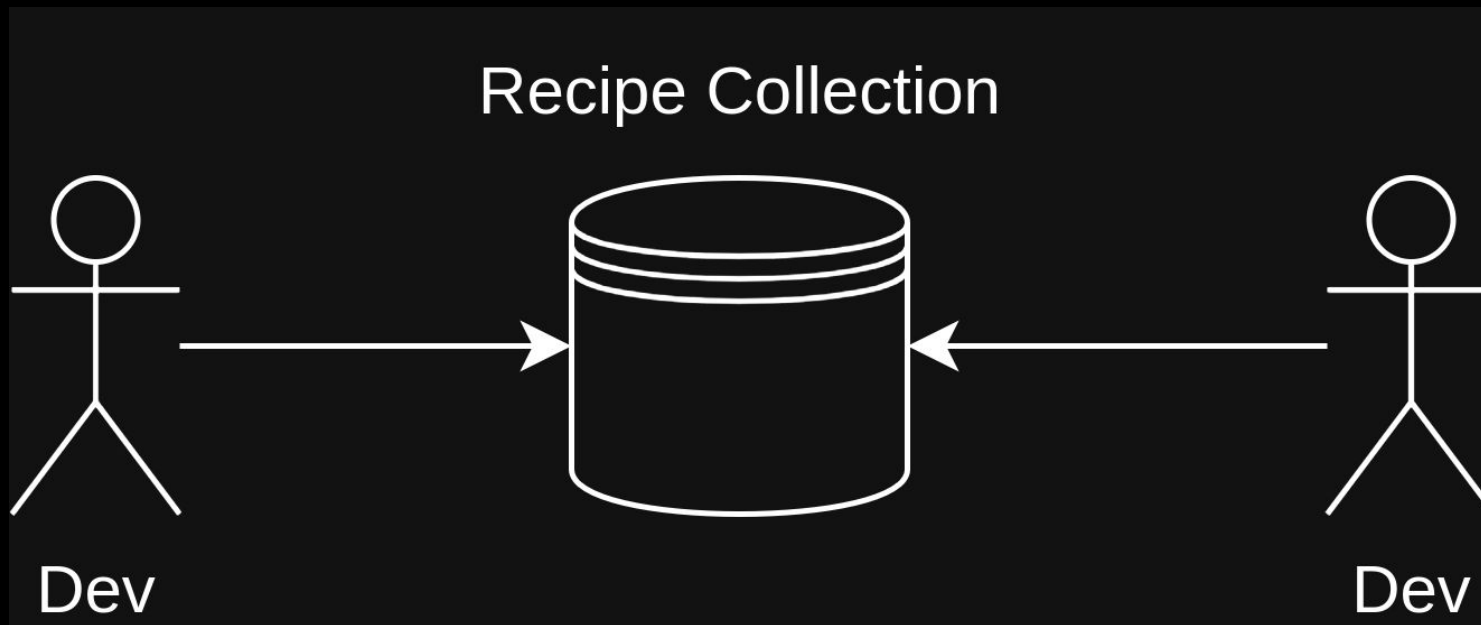
// [https://join.slack.com/t/rewriteoss/shared\\_invite/zt-1ihfggp2a-gllit\\_aXJnhHAdv\\_0uzwow](https://join.slack.com/t/rewriteoss/shared_invite/zt-1ihfggp2a-gllit_aXJnhHAdv_0uzwow)



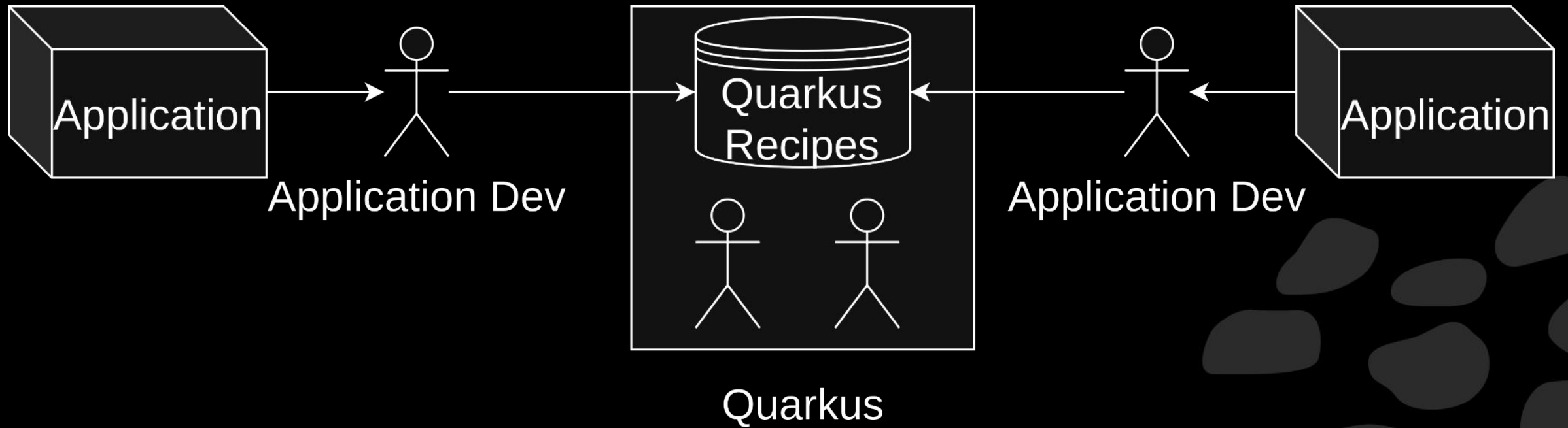


# *Sharing Recipes*

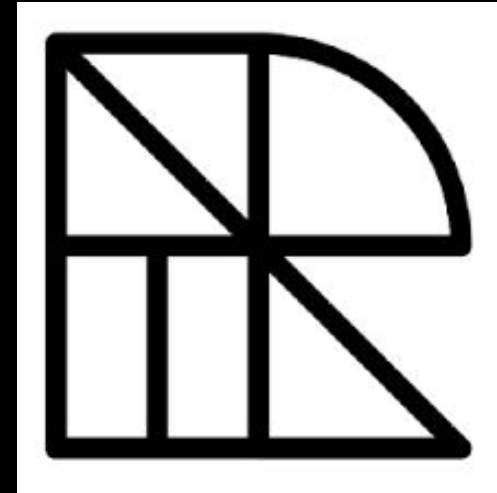
# *In-House Toolbox*



# *Library-oriented*



# Auto Update Service



*// Renovate finds new Updates*

*// OpenRewrite migrates to a new update*



**gepard**ec

*simplify your business*

