

Migrating Speedment to Java 9

Dan Lawesson, [@dan_lawesson](#)
CSO, Speedment, Inc

About Us



About Us



Dan Lawesson, PhD

- AI, model-based diagnosis
- CSO with S as in Science
- 20 years of Java experience
- Previous lives: telecom and automotive IoT

About Us



Dan Lawesson, PhD

- AI, model-based diagnosis
- CSO with S as in Science
- 20 years of Java experience
- Previous lives: telecom and automotive IoT



Spire

- Speedment Open Source mascot
- Lives on GitHub
- 2 years of mascot experience

Outline

Outline

- **Speedment**
- **Jigsaw**
- **Jigsawing Speedment**

Speedment

Speedment

Speedment is

Speedment

Speedment is

- a Streams API ORM

Speedment

Speedment is

- a Streams API ORM
- using in-JVM memory acceleration

Speedment

Speedment is

- a Streams API ORM
- using in-JVM memory acceleration
- and Code Generation,

Speedment

Speedment is

- a Streams API ORM
- using in-JVM memory acceleration
- and Code Generation,
- with modular design

Speedment

Speedment is

- a Streams API ORM
- using in-JVM memory acceleration
- and Code Generation,
- with modular design

Oracle Java Magazine May-June pages 34-40

Speedment

Speedment is

- a Streams API ORM
- using in-JVM memory acceleration
- and Code Generation,
- with modular design

Oracle Java Magazine May-June pages 34-40

//databases /



PER MINBORG

Database Actions Using Java 8 Stream Syntax Instead of SQL

Speedment 3.0 enables Java developers to stay in Java when writing database applications.

Why should you need to use SQL when the same semantics can be derived directly from Java 8 streams? If you take a closer look at this objective, it turns out there is a remarkable resemblance between the verbs of Java 8 streams and SQL commands, as summarized in [Table 1](#).

Streams and SQL queries have similar syntax in part because both are declarative constructs, meaning they describe a result rather than state instructions on how to compute the result. Just as a SQL query describes a result set rather than the operations needed to compute the result, a Java stream describes the result of a sequence of abstract functions without dictating the properties of the actual computation.

The open source project Speedment capitalizes on this similarity to enable you to perform database actions using Java 8 stream syntax instead of SQL. It is available on [GitHub](#) under the business-friendly Apache 2 license for open source databases. (A license fee is required for commercial databases.) Feel free to clone the entire project.

About Speedment

Speedment allows you to write pure Java code for entire database applications. It uses lazy evaluation of streams, meaning that only a minimum set of data is actually pulled from the database into your application and only as the elements are needed.

In the following example, the objective is to print out all `Film` entities having a rating of PG-13 (meaning “parents are strongly cautioned” in the US). The films are located in a database table represented by a Speedment Manager variable

| SQL COMMAND | JAVA 8 STREAM OPERATIONS |
|-------------|--|
| FROM | <code>stream()</code> |
| SELECT | <code>map()</code> |
| WHERE | <code>filter()</code> (BEFORE COLLECTING) |
| HAVING | <code>filter()</code> (AFTER COLLECTING) |
| JOIN | <code>flatMap()</code> OR <code>map()</code> |
| DISTINCT | <code>distinct()</code> |
| UNION | <code>concat(s0, s1).distinct()</code> |
| ORDER BY | <code>sorted()</code> |
| OFFSET | <code>skip()</code> |
| LIMIT | <code>limit()</code> |
| GROUP BY | <code>collect(groupingBy())</code> |
| COUNT | <code>count()</code> |

Table 1. SQL commands and their counterpart verbs in Java 8 streams



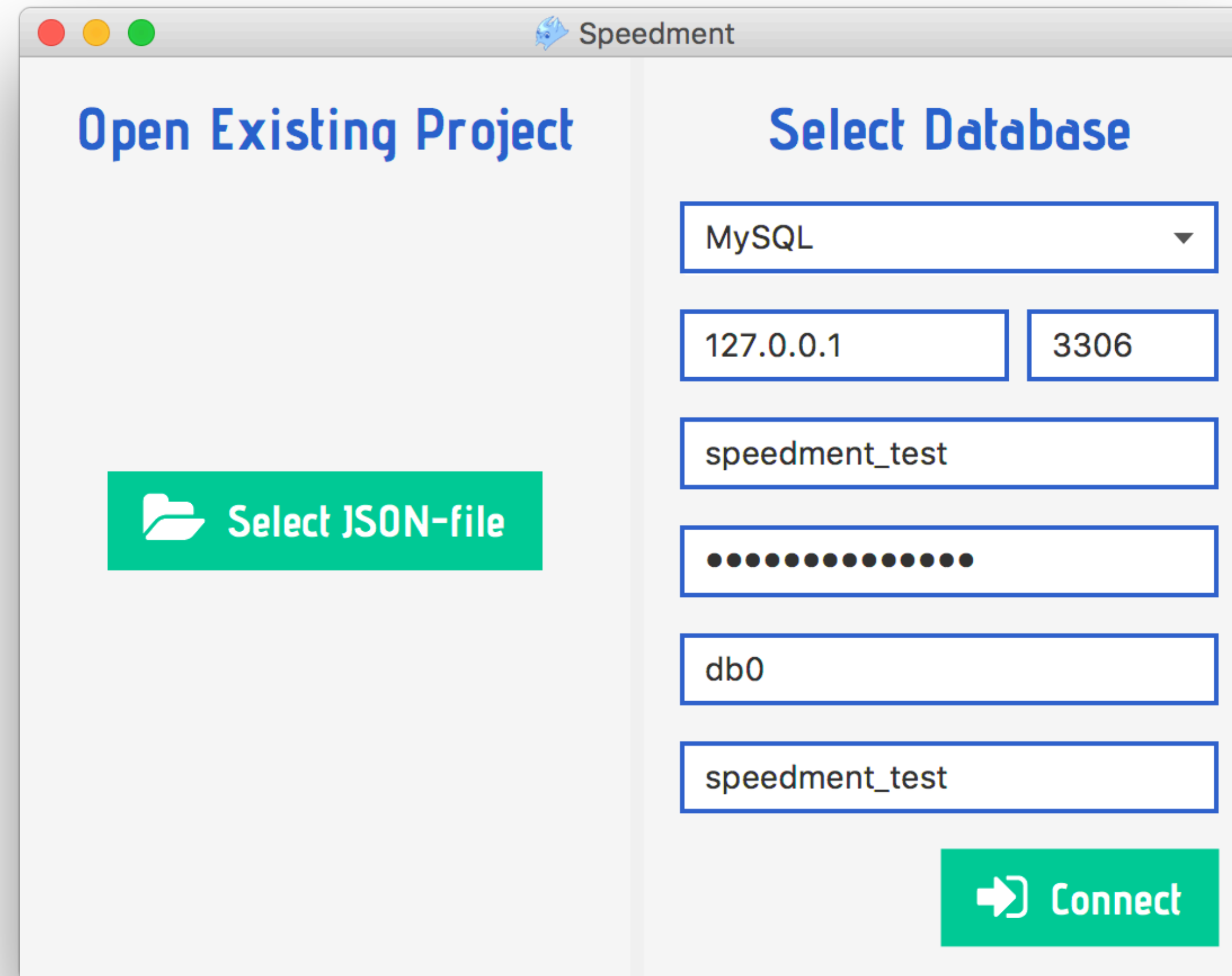
Compile-time Code Generation Tool

The screenshot shows a window titled "Speedment" with two main sections:

- Open Existing Project:** A green button with a folder icon and the text "Select JSON-file".
- Select Database:** A series of input fields for database configuration:
 - Database type: A dropdown menu showing "MySQL".
 - Host: A text input field containing "127.0.0.1".
 - Port: A text input field containing "3306".
 - Username: A text input field containing "speedment_test".
 - Password: A text input field containing ten dots ".....".
 - Schema: A text input field containing "db0".
 - Table: A text input field containing "speedment_test".

At the bottom right of the "Select Database" section is a green button with a right-pointing arrow and the text "Connect".

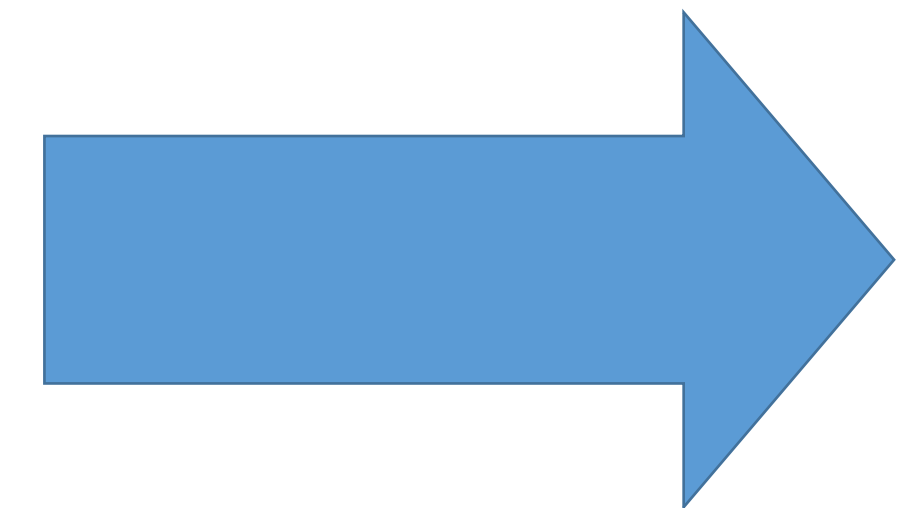
Compile-time Code Generation Tool



The screenshot shows a window titled "Speedment" with two main sections: "Open Existing Project" and "Select Database".

- Open Existing Project:** Contains a green button with a folder icon and the text "Select JSON-file".
- Select Database:** Contains several input fields:
 - A dropdown menu with "MySQL" selected.
 - Two input fields for IP address: "127.0.0.1" and "3306".
 - An input field for database name: "speedment_test".
 - A password field with 12 dots.
 - An input field for username: "db0".
 - An input field for schema: "speedment_test".

At the bottom right of the "Select Database" section is a green button with a right-pointing arrow and the text "Connect".





Reload



Generate



Node hierarchy

- ▼ speedment_test
 - ▼ db0
 - ▼ speedment_test
 - ▶ S_P
 - ▶ TRIGGERLOG
 - ▶ all_type
 - ▼ country
 - id**
 - abbreviation
 - name
 - full_name
 - iso3
 - number

Editing Column 'id'

Enabled

Column Name

Java Alias Auto

Is Nullable

Is Auto Incrementing

JDBC Type to Java

Querying the Database using Streams



Querying the Database using Streams

- Queries are expressed using the standard Java 8 Stream API



Querying the Database using Streams

- Queries are expressed using the standard Java 8 Stream API
- Streams are analyzed to produce high-performance queries



Expressing Queries as Streams

```
customers.stream()  
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
    .count();
```

Expressing Queries as Streams

Standard Stream API

```
customers.stream()  
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
    .count();
```

Expressing Queries as Streams

Standard Stream API

Full Type-Safety

```
customers.stream()  
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
    .count();
```


Expressing Queries as Streams

Standard Stream API

Generated Enum Constants

Full Type-Safety

```
customers.stream()  
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
    .count();
```

Expressing Queries as Streams

Standard Stream API

Generated Enum Constants

Full Type-Safety

```
customers.stream()  
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
    .count();
```

Only 1 value is loaded from DB

Expressing Queries as Streams

Standard Stream API

```
customers.stream()  
  .filter(Customer.REGION.equal(Region.NORTH_AMERICA))  
  .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))  
  .count();
```

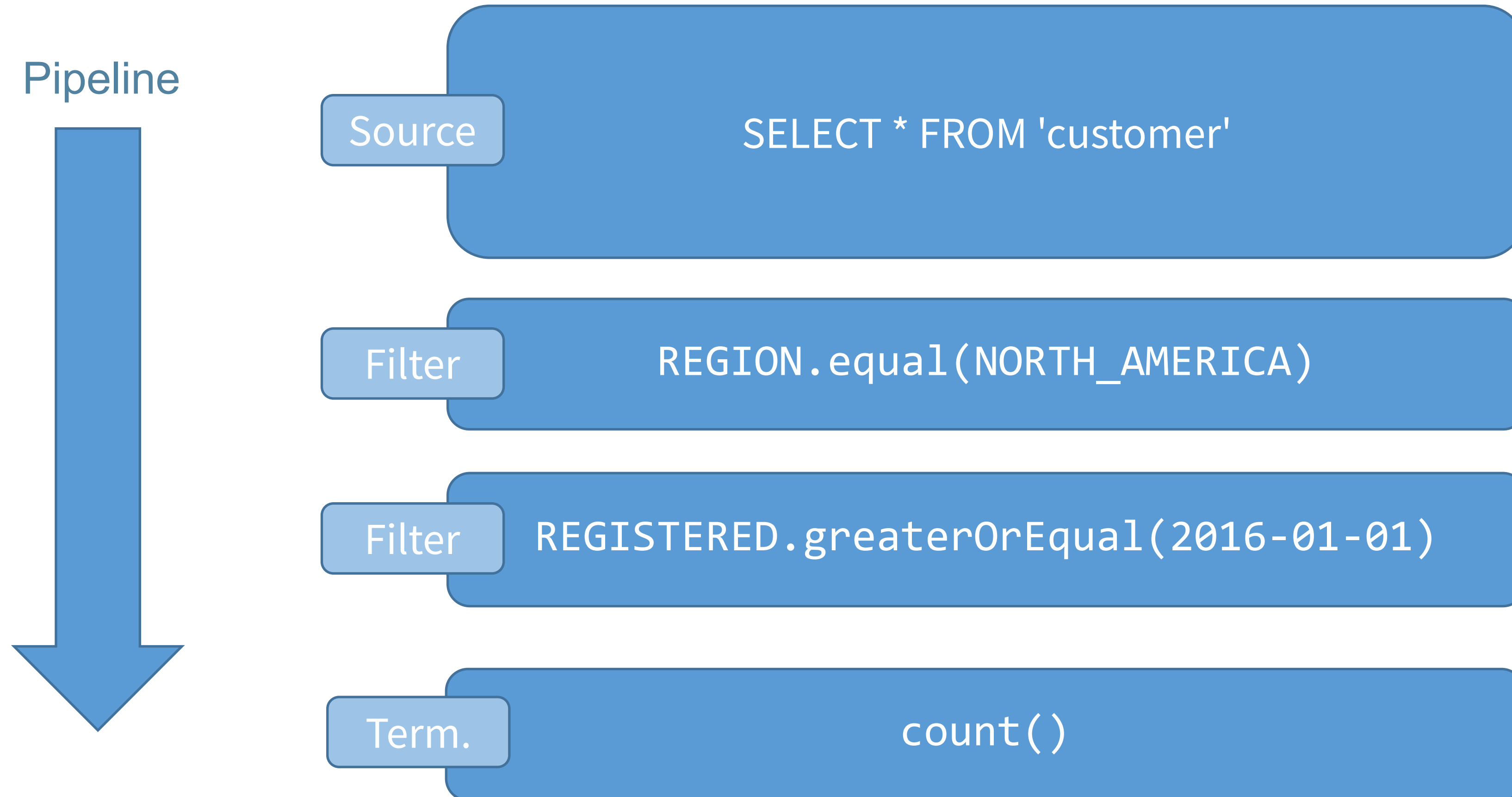
Only 1 value is loaded from DB

Full Type-Safety

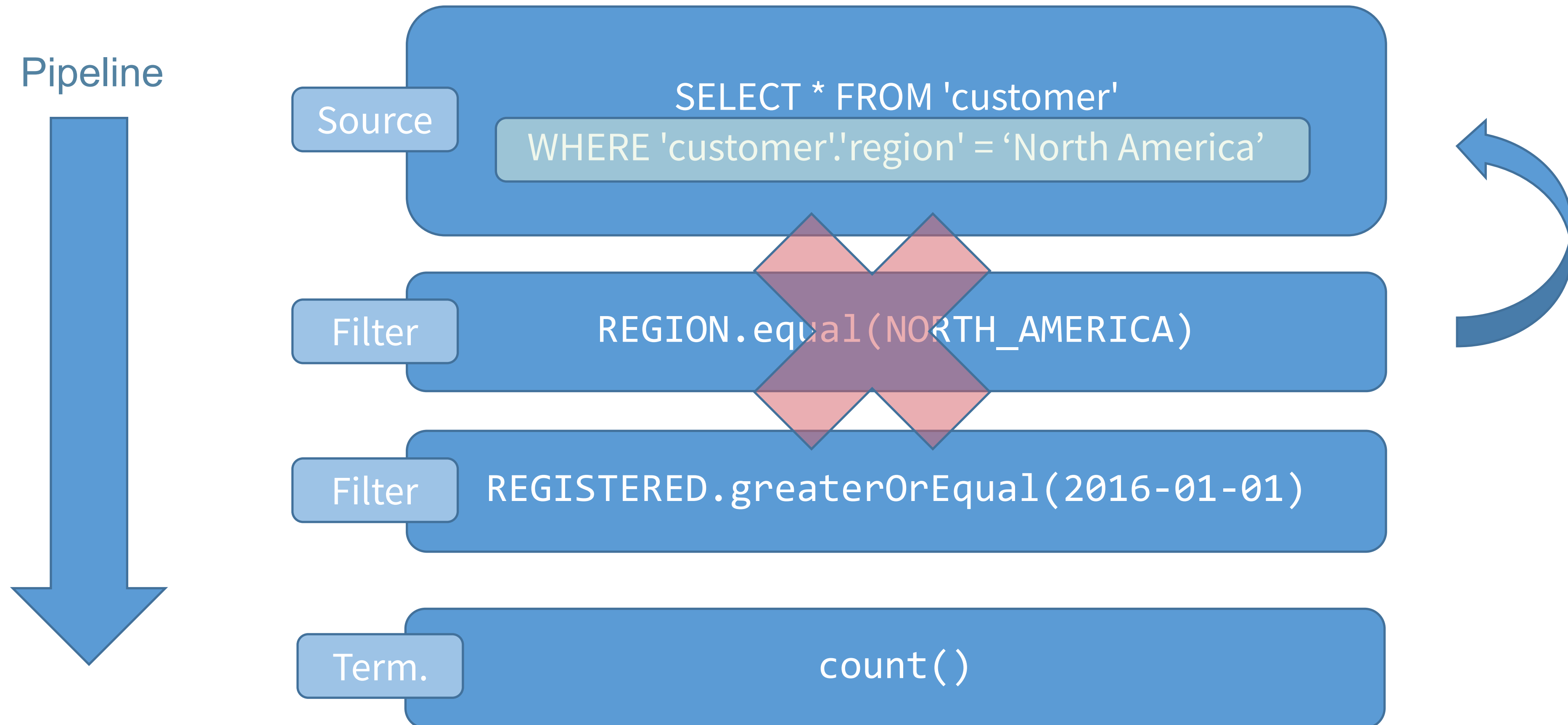
Generated Enum Constants

```
SELECT COUNT('id') FROM 'customer'  
WHERE 'customer'.'region' = 'North America'  
AND 'customer'.'registered' >= '2016-01-01';
```

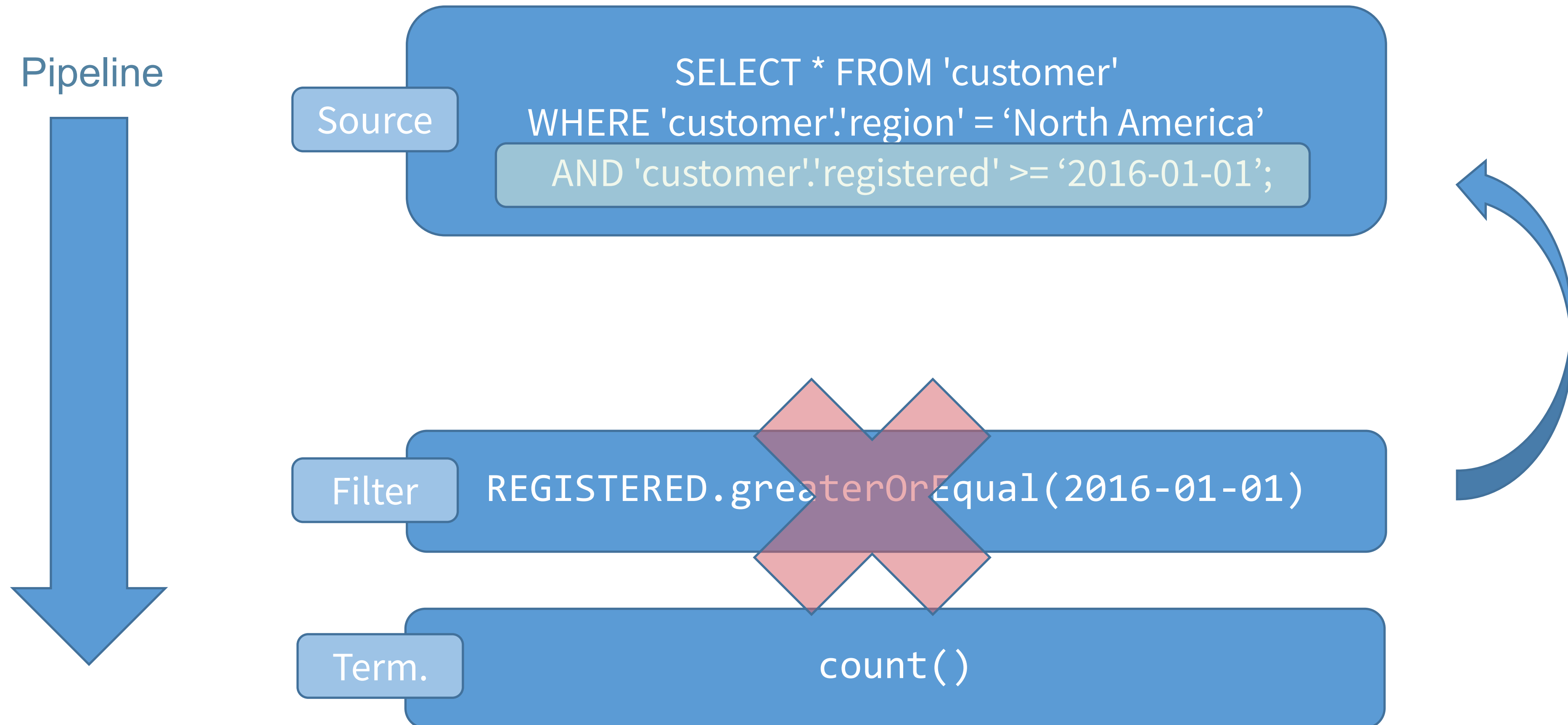
Querying the Database using Streams



Querying the Database using Streams

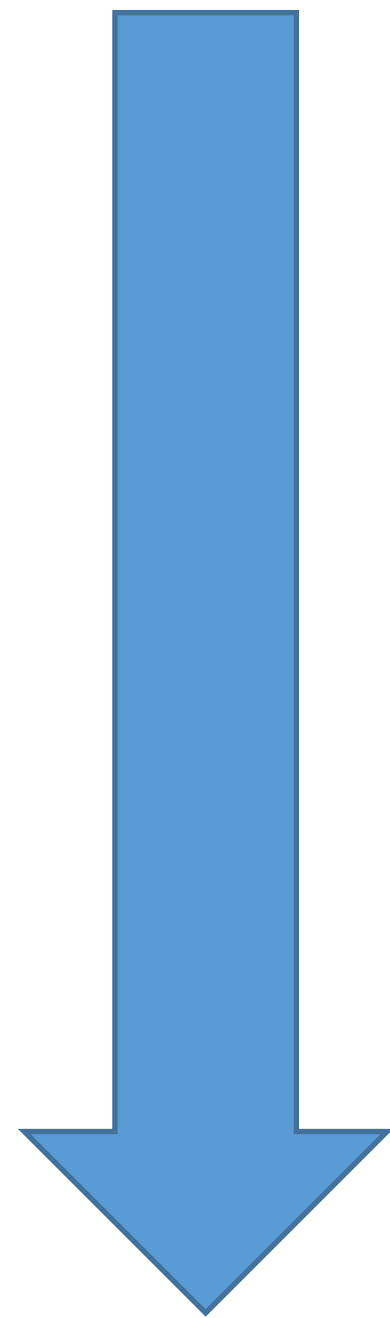


Querying the Database using Streams



Querying the Database using Streams

Pipeline



Source

```
SELECT COUNT('id') FROM 'customer'  
WHERE 'customer'.region = 'North America'  
AND 'customer'.registered >= '2016-01-01';
```

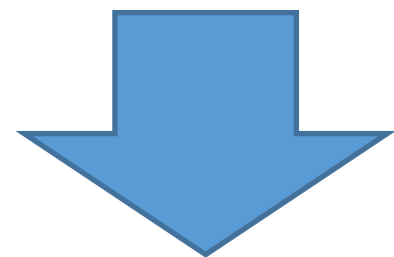
Term.

count()



Querying the Database using Streams

Pipeline



Source

```
SELECT COUNT('id') FROM 'customer'  
WHERE 'customer'.region = 'North America'  
AND 'customer'.registered >= '2016-01-01';
```


Expressing Queries as Streams

```
// Gets the second page of customers in North America  
// sorted by name in the form of a JSON array
```



Expressing Queries as Streams

```
// Gets the second page of customers in North America  
// sorted by name in the form of a JSON array
```

```
customers.stream()  
  .filter(REGION.equal(Region.NORTH_AMERICA))  
  .sorted(NAME.comparator())  
  .skip(10)  
  .limit(10) // JVM from here..  
  .collect(toJson(encode.allOf(customers)))
```

```
[  
  {"id":11, "name": ...},  
  {...},  
  ...  
]
```



Parallelism as Expected for a Stream

```
// Supports parallelism on custom executors  
// with full control of thread work item layout
```



Parallelism as Expected for a Stream

```
// Supports parallelism on custom executors  
// with full control of thread work item layout  
  
customers.stream()  
    .parallel()  
    .filter(REGION.equal(Region.NORTH_AMERICA))  
    .forEach(expensiveOperation());
```



Querying

```
Optional<Hare> oldHare = hares.stream()  
    .filter(Hare.AGE.greaterThan(5))  
    .findAny();
```

Querying

```
Optional<Hare> oldHare = hares.stream()  
    .filter(Hare.AGE.greaterThan(5))  
    .findAny();
```

Predicate Builder

Querying

```
Optional<Hare> oldHare = hares.stream()  
    .filter(Hare.AGE.greaterThan(5))  
    .findAny();
```

Predicate Builder

```
SELECT id, name, color, age  
FROM hare  
WHERE (age > 5) LIMIT 1;
```

Entities are Linked

```
// Find the owner of the orange carrot
Optional<Hare> hare = carrots.stream()
    .filter(Carrot.NAME.equal("Orange"))
    .map(hares.finderBy(Carrot.OWNER))
    .findAny();
```

```
// Find one carrot owned by Harry
// Carrot is a foreign key table
Optional<Carrot> carrot = hares.stream()
    .filter(Hare.NAME.equal("Harry"))
    .flatMap(carrots.finderBackwardsBy(Carrot.OWNER))
    .findAny();
```


Joins

```
Map<Hare, List<Carrot>> join = carrots.stream()
    .collect(
        groupingBy(hares.finderBy(Carrot.OWNER))
    );
```

Joins

```
Map<Hare, List<Carrot>> join = carrots.stream()
    .collect(
        groupingBy(hares.finderBy(Carrot.OWNER))
    );
```

Standard Java 8

Joins

```
Map<Hare, List<Carrot>> join = carrots.stream()  
    .collect(  
        groupingBy(hares.finderBy(Carrot.OWNER))  
    );
```

Standard Java 8

Imperative vs Declarative

Imperative vs Declarative

SQL describes *what* rather than *how*

Imperative vs Declarative

SQL describes *what* rather than *how*

database engine figures out the how

Imperative vs Declarative

SQL describes *what* rather than *how*

database engine figures out the *how*

Streams are also declarative - a result is described

Imperative vs Declarative

SQL describes *what* rather than *how*

database engine figures out the *how*

Streams are also declarative - a result is described

framework takes pipeline as input and determines the *how*

Typical ORM + Java Streams Application

Typical ORM + Java Streams Application

Explicit SQL + Java Stream is an *imperative* two step program:

Typical ORM + Java Streams Application

Explicit SQL + Java Stream is an *imperative* two step program:

1. Compute Result Set

```
SELECT x FROM y WHERE z
```

Typical ORM + Java Streams Application

Explicit SQL + Java Stream is an *imperative* two step program:

1. Compute Result Set

```
SELECT x FROM y WHERE z
```

2. Stream over Result Set

```
rs.stream().filter...
```

Speedment - Breaking the Language Barrier

Speedment - Breaking the Language Barrier

Java Stream expressing **both** database and JVM operations ->

Speedment - Breaking the Language Barrier

Java Stream expressing **both** database and JVM operations ->

the program is *declarative*,

Speedment - Breaking the Language Barrier

Java Stream expressing **both** database and JVM operations ->

the program is *declarative*,

the runtime determines the DB-JVM work split.

Speedment - Fully Declarative DB Applications

Speedment - Fully Declarative DB Applications

Fully declarative ->

Speedment - Fully Declarative DB Applications

Fully declarative ->

Speedment Enterprise: add in-memory-acceleration

Speedment - Fully Declarative DB Applications

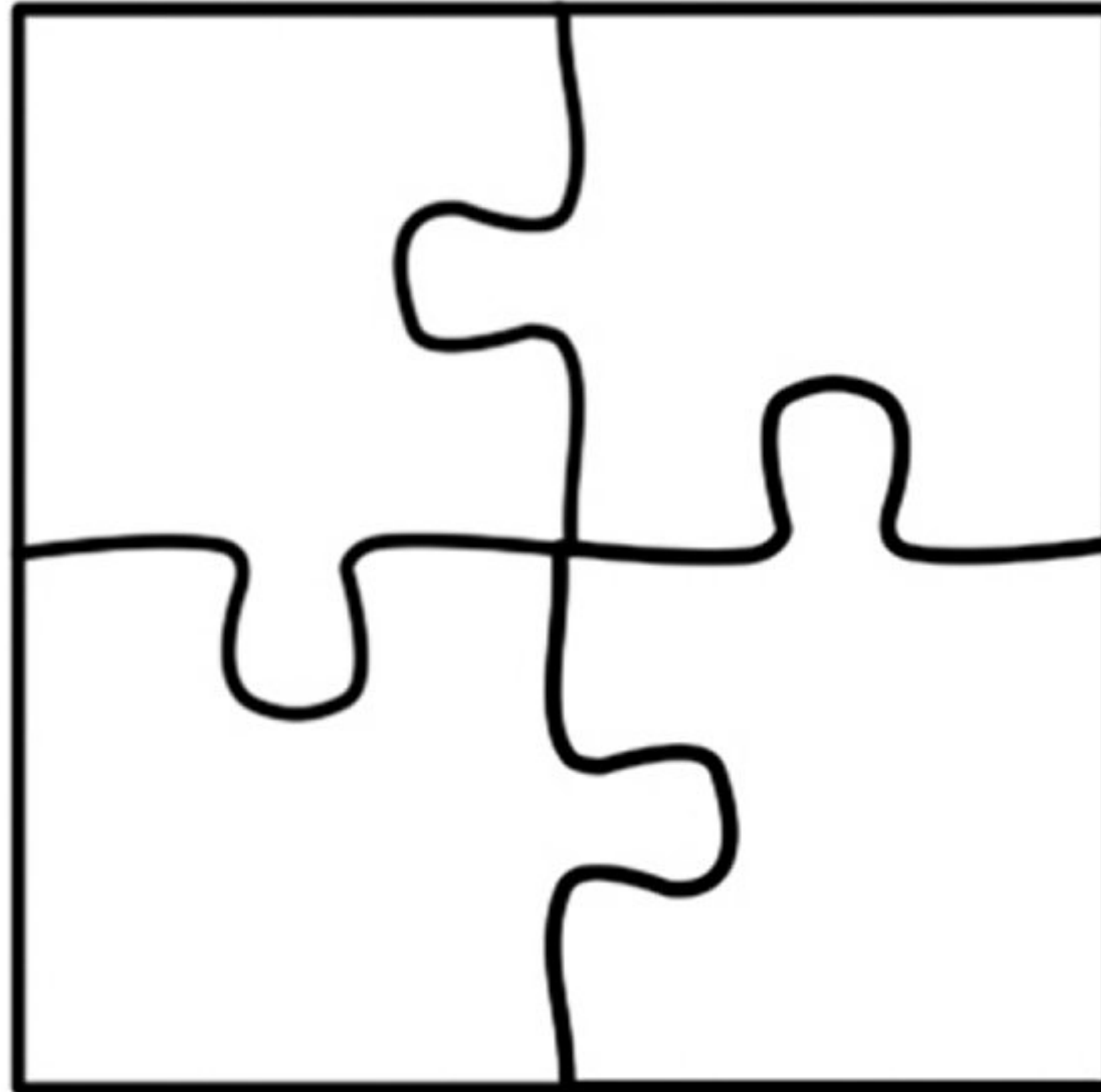
Fully declarative ->

Speedment Enterprise: add in-memory-acceleration

without changing anything in the stream application.

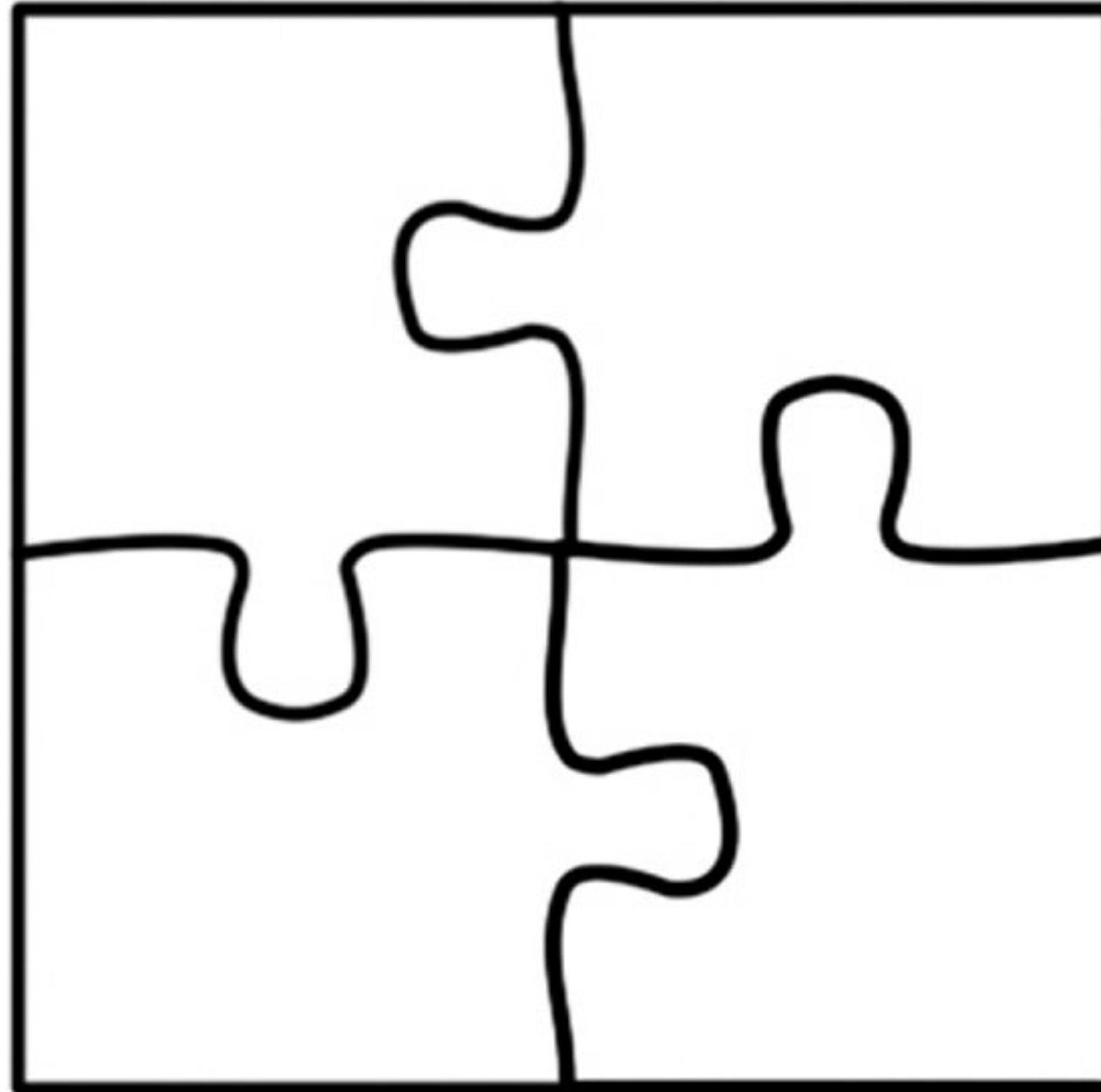
Jigsaw

Jigsaw



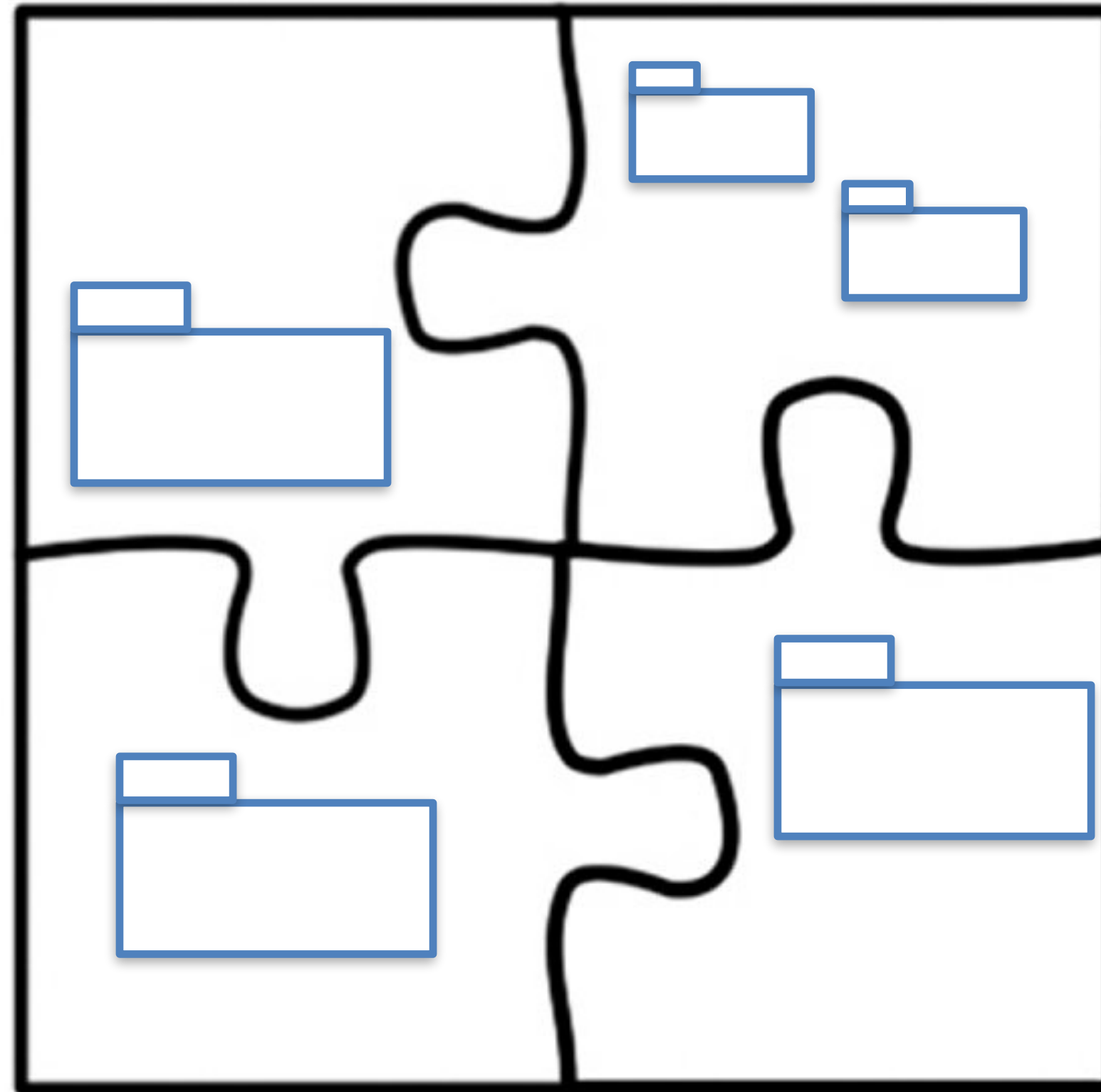
Jigsaw

- Modules - an abstraction on top of packages
 - Strong Encapsulation
 - Reliable Configuration



Jigsaw

- Modules - an abstraction on top of packages
 - Strong Encapsulation
 - Reliable Configuration



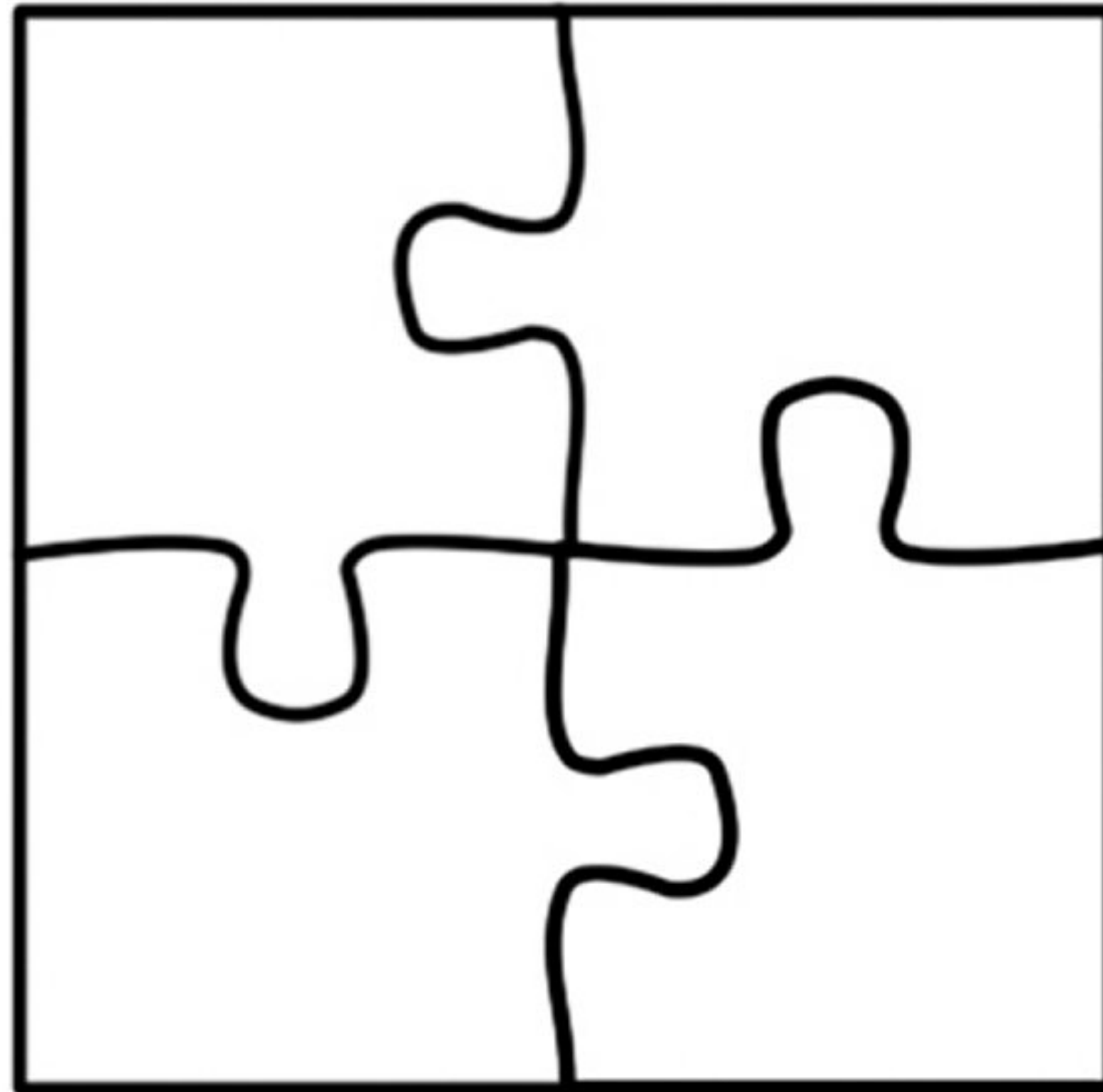


Split Packages

Reliable configuration - a package may only belong to one module

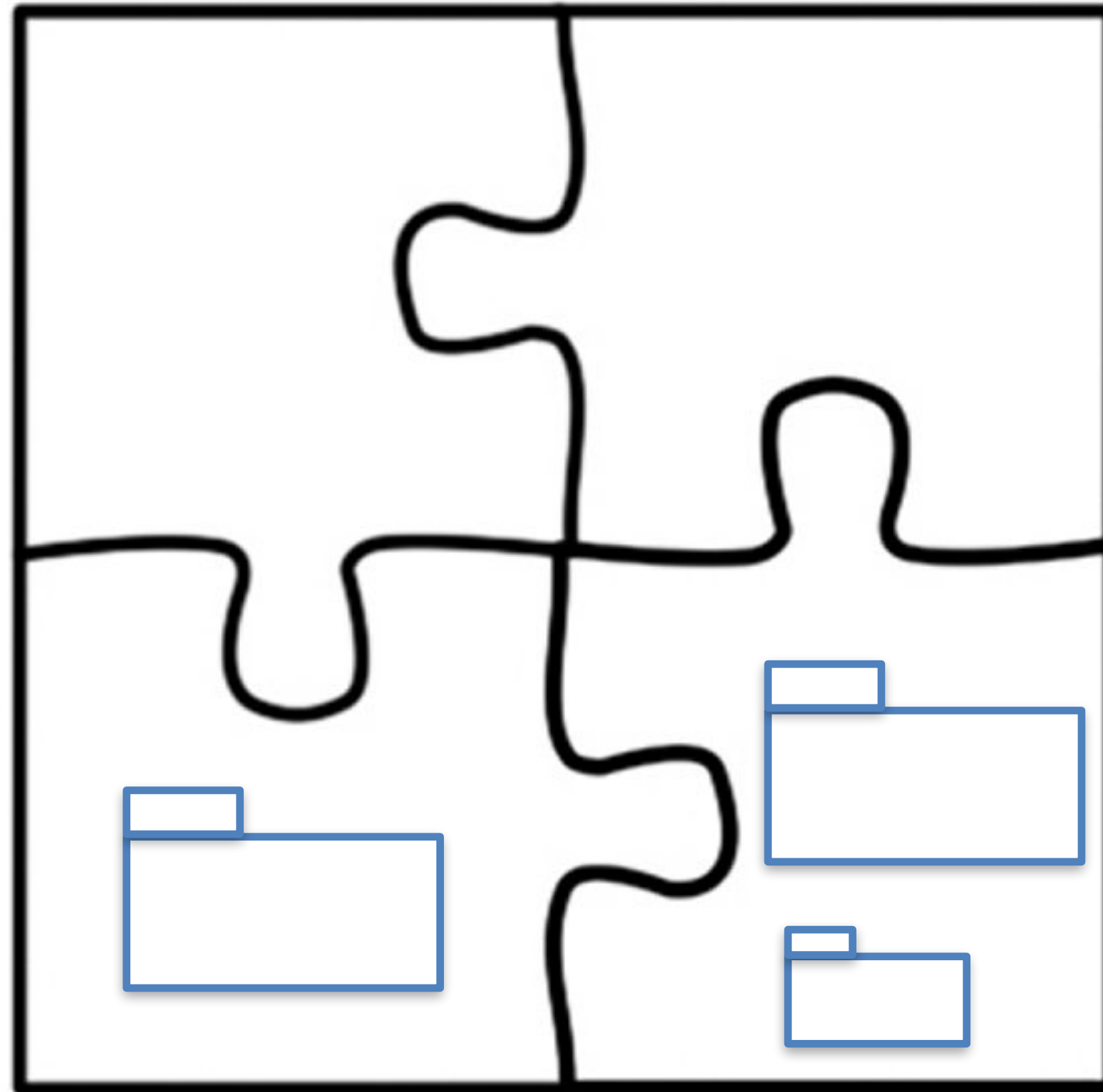
Split Packages

Reliable configuration - a package may only belong to one module



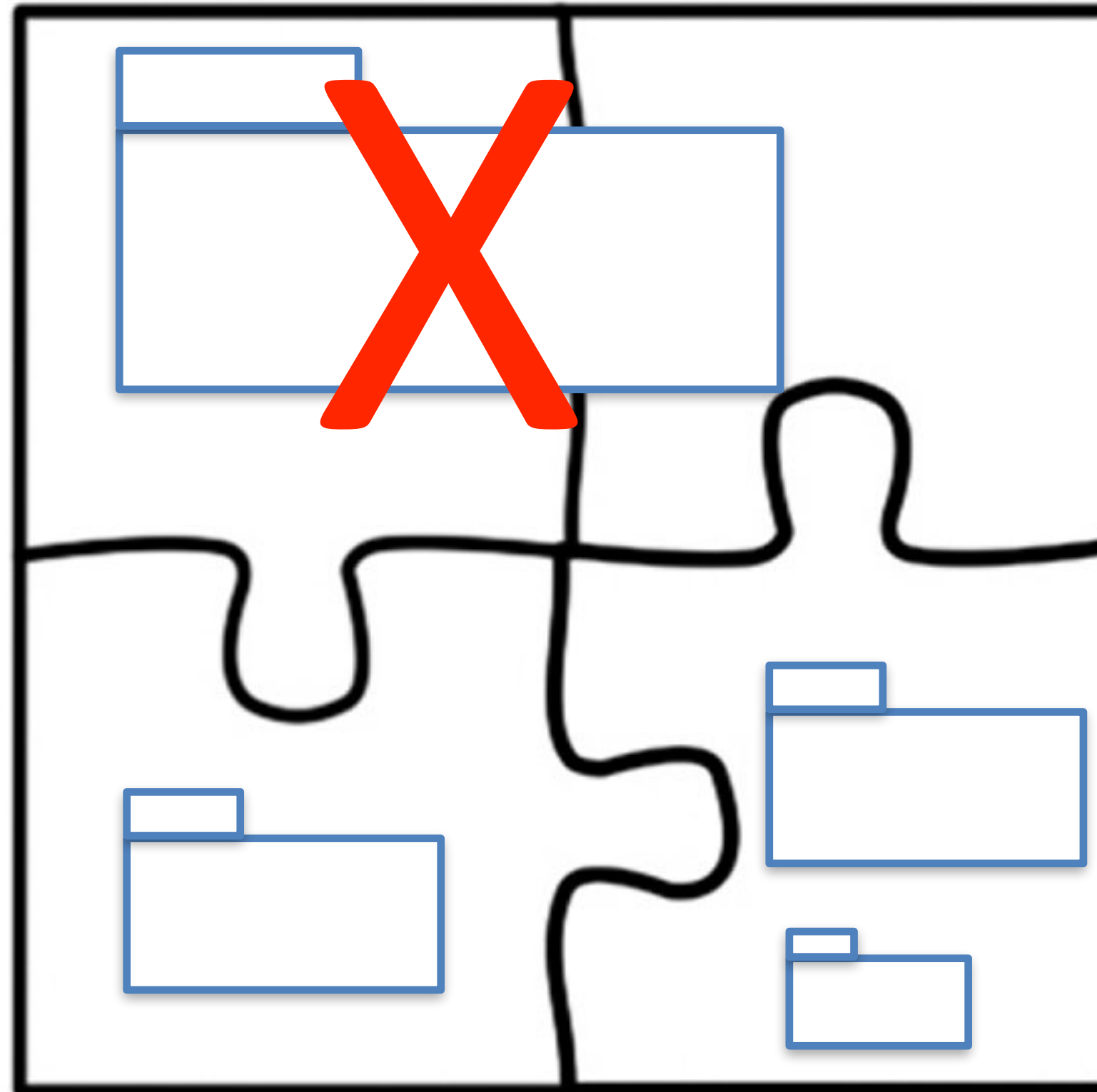
Split Packages

Reliable configuration - a package may only belong to one module



Split Packages

Reliable configuration - a package may only belong to one module





Automatic Modules

Automatic Modules

- Smooth transition to Java 9

Automatic Modules

- Smooth transition to Java 9
- Move the Java 8 JAR from class path to module path

Automatic Modules

- Smooth transition to Java 9
- Move the Java 8 JAR from class path to module path
- The JAR automatically becomes a module



Automatic modules vs split packages

- Maven puts dependencies on the module path

Automatic modules vs split packages

- Maven puts dependencies on the module path
- Java 8 to 9 portability -> automatic modules

Automatic modules vs split packages

- Maven puts dependencies on the module path
- Java 8 to 9 portability -> automatic modules
- Automatic modules give **split packages**

Automatic modules vs split packages

- Maven puts dependencies on the module path
- Java 8 to 9 portability -> automatic modules
- Automatic modules give **split packages**
- But reliable configuration means **no split packages**

Automatic modules vs split packages

- Maven puts dependencies on the module path
- Java 8 to 9 portability -> automatic modules
- Automatic modules give **split packages**
- But reliable configuration means **no split packages**

⊥

Some Other Jigsaw Controversies

Some Other Jigsaw Controversies

- Works for JDK \neq works for applications
 - Restricts current application domain use cases

Some Other Jigsaw Controversies

- Works for JDK \neq works for applications
 - Restricts current application domain use cases
- `sun.misc.Unsafe`
 - "Should not be used" **vs** "A key for Java real-world success"

Some Other Jigsaw Controversies

- Works for JDK \neq works for applications
 - Restricts current application domain use cases
- `sun.misc.Unsafe`
 - "Should not be used" **vs** "A key for Java real-world success"
- Fundamentally different compared to e.g. OSGi
 - lazy loading, dynamic package adding, split packages

Jigsawing the Java 8 Application



Jigsawing the Java 8 Application

- Run Java 8 under Java 9 is super easy



Jigsawing the Java 8 Application

- Run Java 8 under Java 9 is super easy

```
java -cp <...> -jar app.jar
```



Jigsawing the Java 8 Application

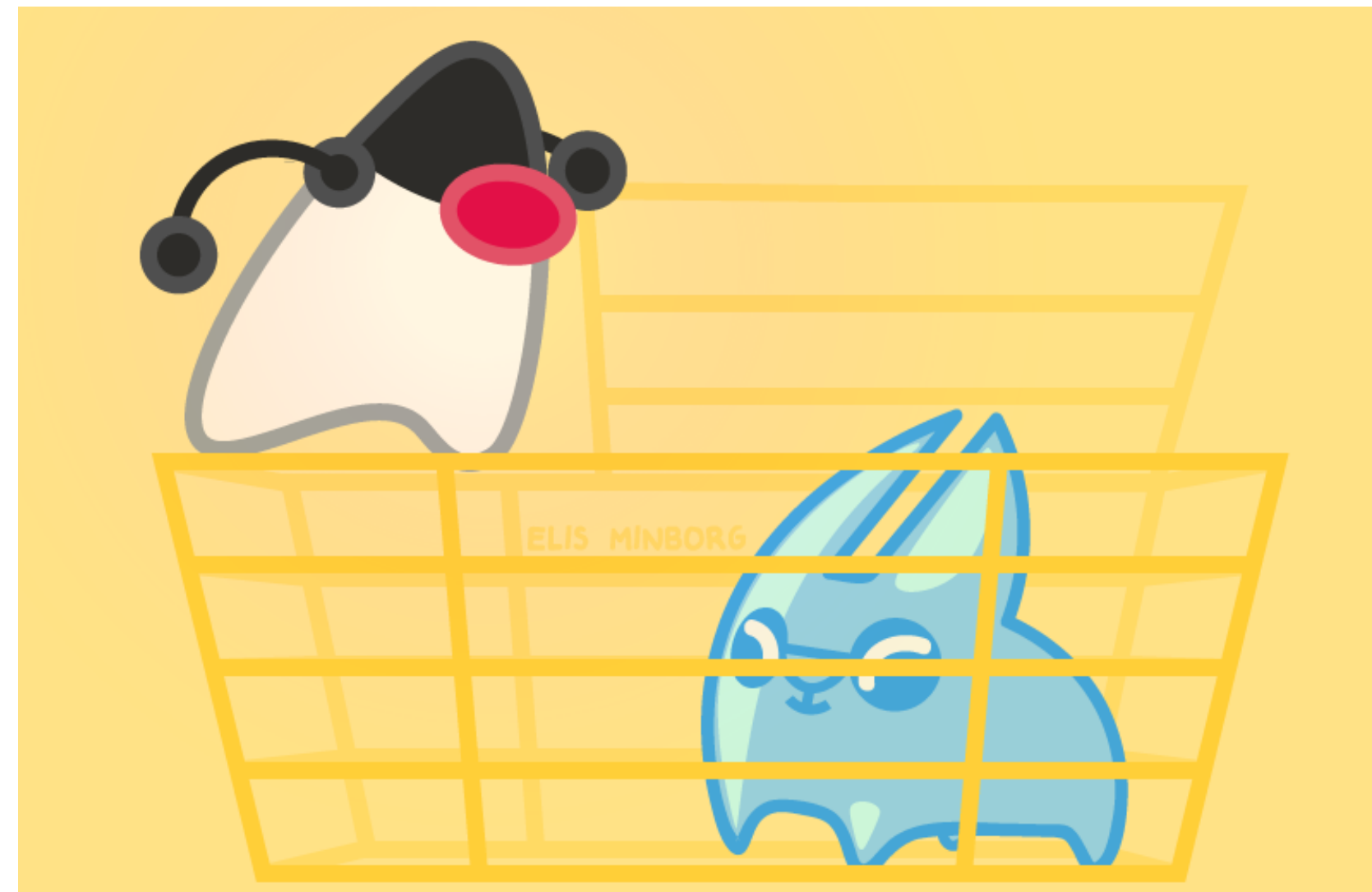
- Run Java 8 under Java 9 is super easy

```
java -cp <...> -jar app.jar
```

- The challenge is to move from cp to mp

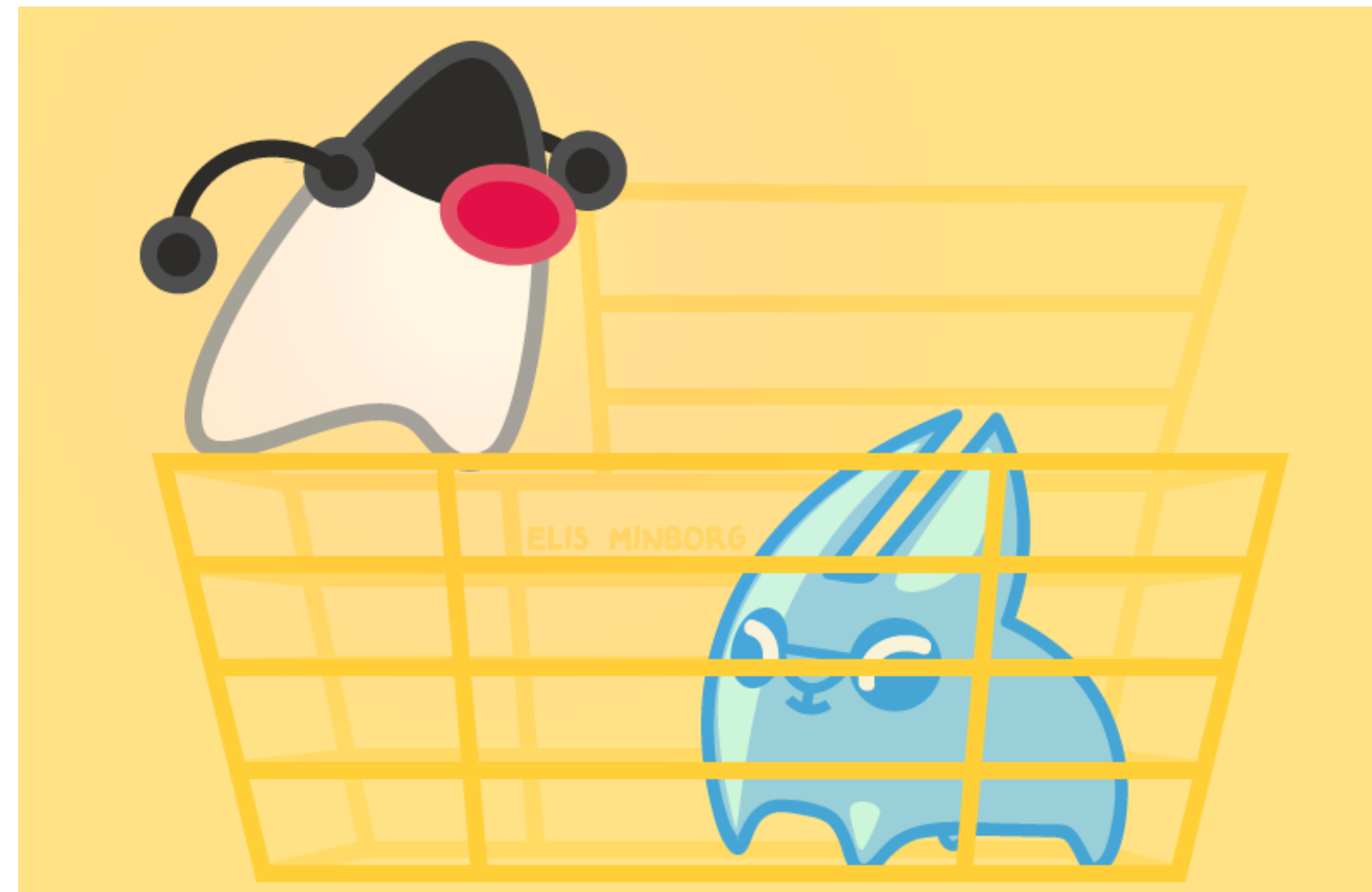


Inter module dependencies



Inter module dependencies

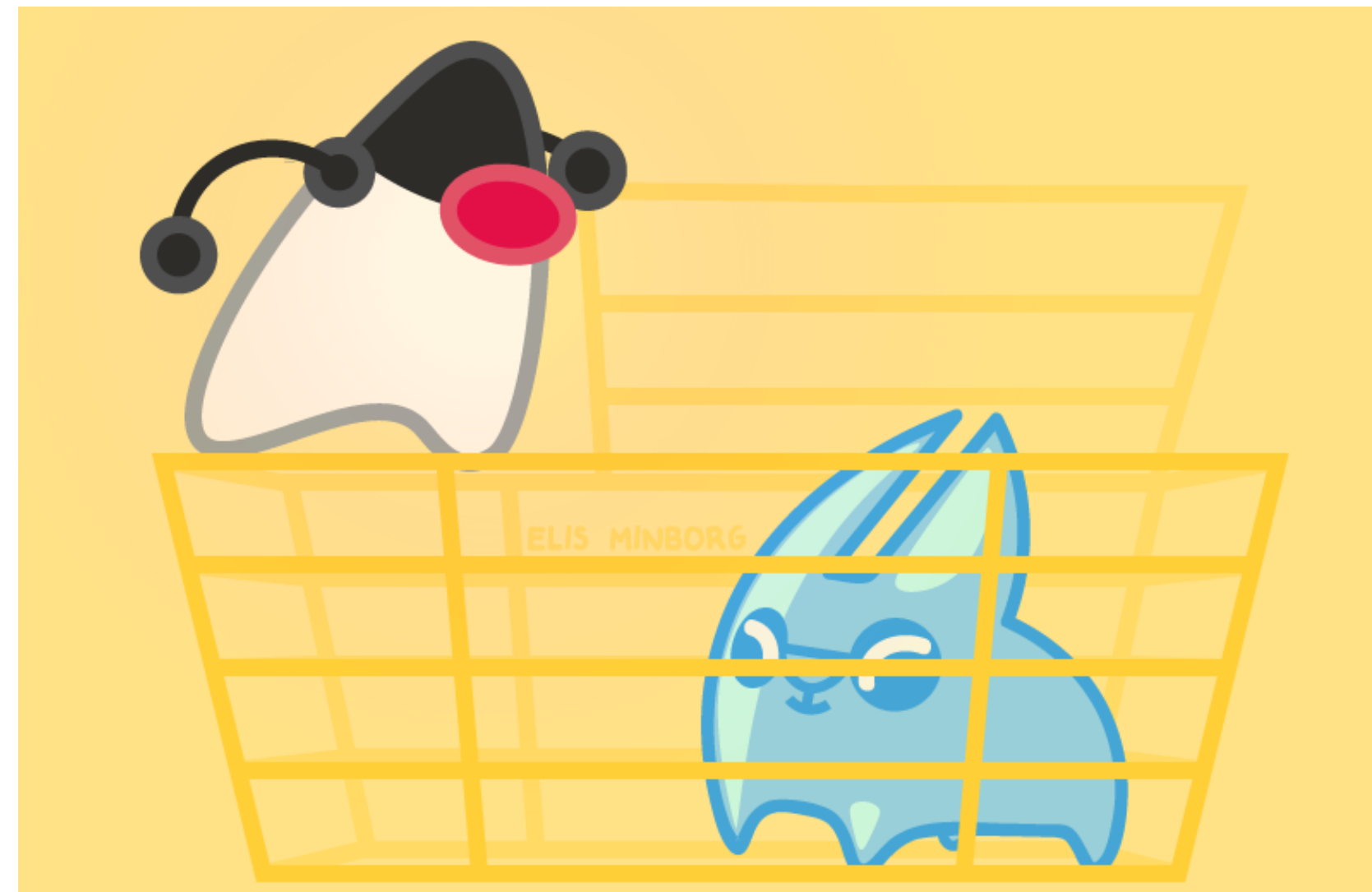
Dependencies explicitly given in module-info.java



Inter module dependencies

Dependencies explicitly given in module-info.java

```
module com.speedment.common {  
    requires com.foo.bar;           // a module we depend on  
    exports com.speedment.common.invariant; // a package we expose to the user  
}
```



A Straight-Forward Modularization Approach

class path

module path

A Straight-Forward Modularization Approach

class path

module path

1. All JARs on class path

<all jars>

A Straight-Forward Modularization Approach

| | class path | module path |
|-------------------------------|------------|-------------|
| 1. All JARs on class path | <all jars> | |
| 2. App as a monolithic module | <all deps> | myapp.jar |

A Straight-Forward Modularization Approach

| | class path | module path |
|-------------------------------|------------|-------------|
| 1. All JARs on class path | <all jars> | |
| 2. App as a monolithic module | <all deps> | myapp.jar |
| 3. Declare dependencies | | |

A Straight-Forward Modularization Approach

| | class path | module path |
|---------------------------------|------------|-------------|
| 1. All JARs on class path | <all jars> | |
| 2. App as a monolithic module | <all deps> | myapp.jar |
| 3. Declare dependencies | | |
| 4. Move some JARs from cp to mp | <deps> | -> |

A Straight-Forward Modularization Approach

| | class path | module path |
|---------------------------------|------------|-------------|
| 1. All JARs on class path | <all jars> | |
| 2. App as a monolithic module | <all deps> | myapp.jar |
| 3. Declare dependencies | | |
| 4. Move some JARs from cp to mp | <deps> | -> |
| 5. Modularize app | | |

The Speedment Java 8 Modules

The Speedment Java 8 Modules

Speedment pom.xml

The Speedment Java 8 Modules

Speedment pom.xml

```
<modules>  
  <module>common-parent</module>  
  <module>runtime-parent</module>  
  <module>generator-parent</module>  
  <module>tool-parent</module>  
  <module>build-parent</module>  
  <module>plugin-parent</module>  
  <module>connector-parent</module>  
  <module>archetype-parent</module>  
  <module>example-parent</module>  
</modules>
```

The Modules of the Speedment common Module

The Modules of the Speedment common Module

Speedment common/pom.xml

The Modules of the Speedment common Module

Speedment common/pom.xml

```
<modules>
  <module>invariant</module>
  <module>function</module>
  <module>json</module>
  <module>tuple</module>
  <module>logger</module>
  <module>codegen</module>
  <module>codegenxml</module>
  <module>injector</module>
  <module>rest</module>
  <module>lazy</module>
  <module>mapstream</module>
  <module>mutablestream</module>
  <module>singletonstream</module>
  <module>common-all</module>
  <module>annotation</module>
  <module>benchmark</module>
  <module>combinatorics</module>
  <module>collection</module>
</modules>
```

Automatic Jigsawing Script Outline

Automatic Jigsawing Script Outline

- Create separate directories for the modules

Automatic Jigsawing Script Outline

- Create separate directories for the modules
- Move source code packages to respective modules

Automatic Jigsawing Script Outline

- Create separate directories for the modules
- Move source code packages to respective modules
- Add empty module-info.java

Automatic Jigsawing Script Outline

- Create separate directories for the modules
- Move source code packages to respective modules
- Add empty module-info.java
- Loop:
 1. Compile

Automatic Jigsawing Script Outline

- Create separate directories for the modules
- Move source code packages to respective modules
- Add empty module-info.java
- Loop:
 1. Compile
 2. Parse and fix errors

Automatic Jigsawing Script Outline

- Create separate directories for the modules
- Move source code packages to respective modules
- Add empty module-info.java
- Loop:
 1. Compile
 2. Parse and fix errors
 3. git add
 4. git commit



Automatic Jigsawing - add requires

Automatic Jigsawing - add requires

Missing Speedment internal dependency

```
\[ERROR\].*\(package (.*) is declared in module (.*),  
but module com.speedment.(.*)\.(.*) does not read it\)
```

Fix:

```
module com.speedment.X {  
+   requires com.speedment.Y;  
}
```


Automatic Jigsawing - add exports

Automatic Jigsawing - add exports

Missing Speedment internal visibility

```
\[ERROR\].*\(package com.speedment.(.*)\.(.*) is declared in  
module com.speedment.(.*)\.(.*), which does not export it\)
```

Fix:

```
module com.speedment.Y {  
+   exports com.speedment.X;  
}
```

Automatic Jigsawing - patch (ab)use of JDK API

Automatic Jigsawing - patch (ab)use of JDK API

Speedment usage of non-public JDK API

```
\[ERROR\].*\(package (.*) is declared in module (.*), which does  
not export it to module com.speedment.(.*)\.(.*)\)
```

Temporary workaround: add exports in the pom file

```
    <artifactId>maven-compiler-plugin</artifactId>  
+    <configuration>  
+        <compilerArgs>  
+            <arg>-add-exports</arg><arg>java.base/sun.nio.ch=com.speedment...  
+        </compilerArgs>  
+    </configuration>
```

Automatic Jigsawing - remove OSGi bundling

Automatic Jigsawing - remove OSGi bundling

Maven does not currently coexist well with OSGi bundling and Jigsaw

```
\[ERROR\].*Manifest com\.speedment\.([\^:]*)\:([\^:]*)\:[\^:]*\:.* \: Invalid  
class file module-info\.class \  
(java\.lang\.ArrayIndexOutOfBoundsException\: 19\)
```

Temporary workaround: comment out the bundling

```
- <packaging>bundle</packaging>  
+ <packaging>jar</packaging>  
...  
    <plugins>  
+     <!--  
        <plugin>  
            <groupId>org.apache.felix</groupId>  
...  
    </plugins>
```

Speedment Open Source - the Easy Case

Speedment Open Source - the Easy Case

- No usage of `sun.misc.Unsafe`

Speedment Open Source - the Easy Case

- No usage of `sun.misc.Unsafe`
- No third party dependencies

Speedment Open Source - the Easy Case

- No usage of `sun.misc.Unsafe`
- No third party dependencies
 - JDK dependencies on module path - no automatic modules

Speedment Open Source - the Easy Case

- No usage of `sun.misc.Unsafe`
- No third party dependencies
 - JDK dependencies on module path - no automatic modules
 - Automatic script creates workarounds that will need revisiting

Speedment Open Source - the Easy Case

- No usage of `sun.misc.Unsafe`
- No third party dependencies
 - JDK dependencies on module path - no automatic modules
 - Automatic script creates workarounds that will need revisiting
- No reflection

No Reflection in Speedment

No Reflection in Speedment

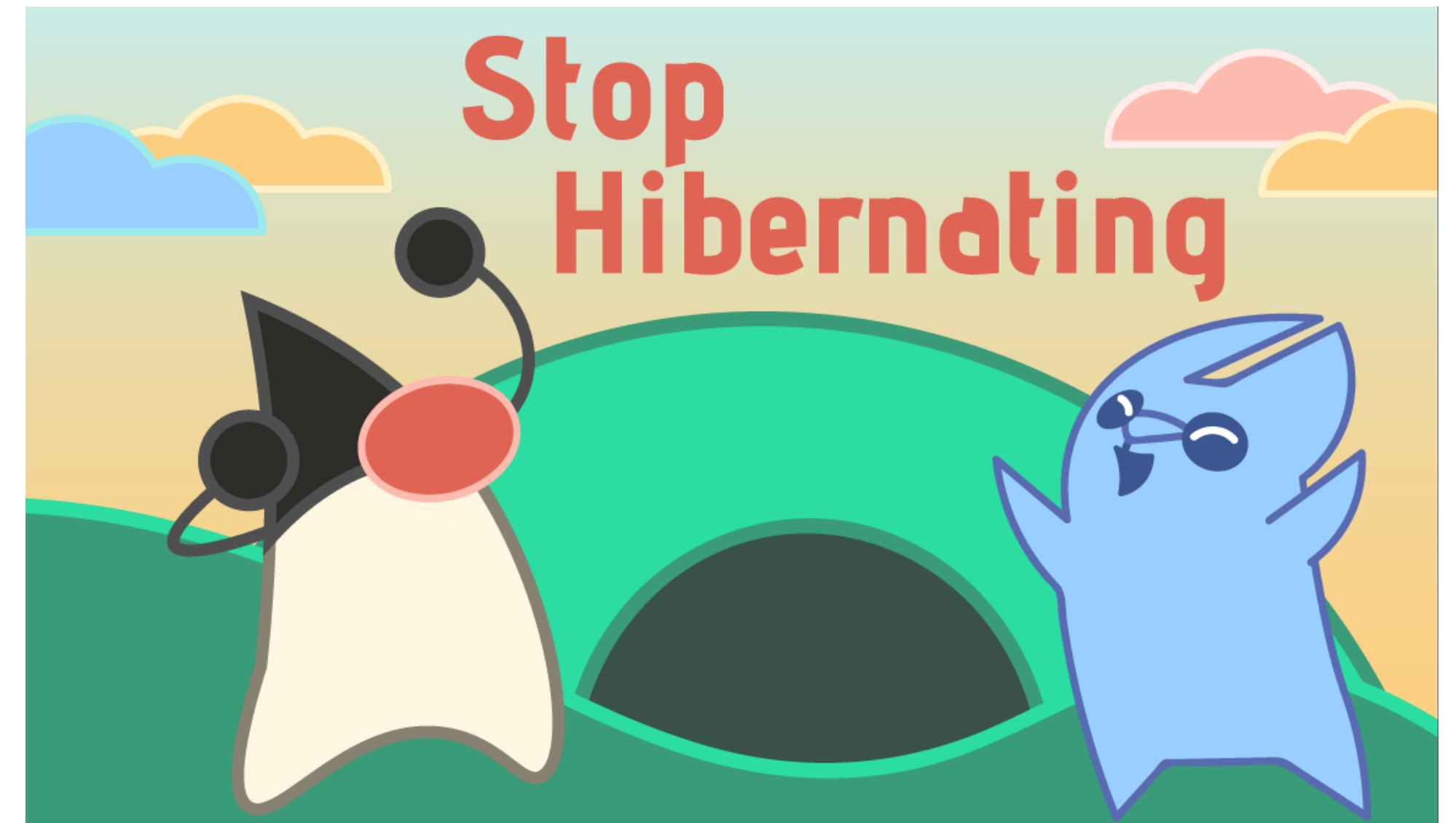
- O/R mapping -> reflection

No Reflection in Speedment

- O/R mapping -> reflection
 - instantiate user code

No Reflection in Speedment

- O/R mapping -> reflection
 - instantiate user code
- Code Generation to the Rescue!



Speedment Enterprise - a Jigsaw Puzzle

Speedment Enterprise - a Jigsaw Puzzle

Uses sun.misc.Unsafe

Speedment Enterprise - a Jigsaw Puzzle

Uses `sun.misc.Unsafe`

- `DirectBuffer.cleaner()` - for predictable cleanup

Speedment Enterprise - a Jigsaw Puzzle

Uses `sun.misc.Unsafe`

- `DirectBuffer.cleaner()` - for predictable cleanup
- `DirectBuffer.address()` - for efficient off-heap byte arrays

Speedment Enterprise - a Jigsaw Puzzle

Uses sun.misc.Unsafe

- `DirectBuffer.cleaner()` - for predictable cleanup
- `DirectBuffer.address()` - for efficient off-heap byte arrays

Third party dependencies

Speedment Enterprise - a Jigsaw Puzzle

Uses sun.misc.Unsafe

- `DirectBuffer.cleaner()` - for predictable cleanup
- `DirectBuffer.address()` - for efficient off-heap byte arrays

Third party dependencies

- Depends on Third Party JARs

Speedment Enterprise - a Jigsaw Puzzle

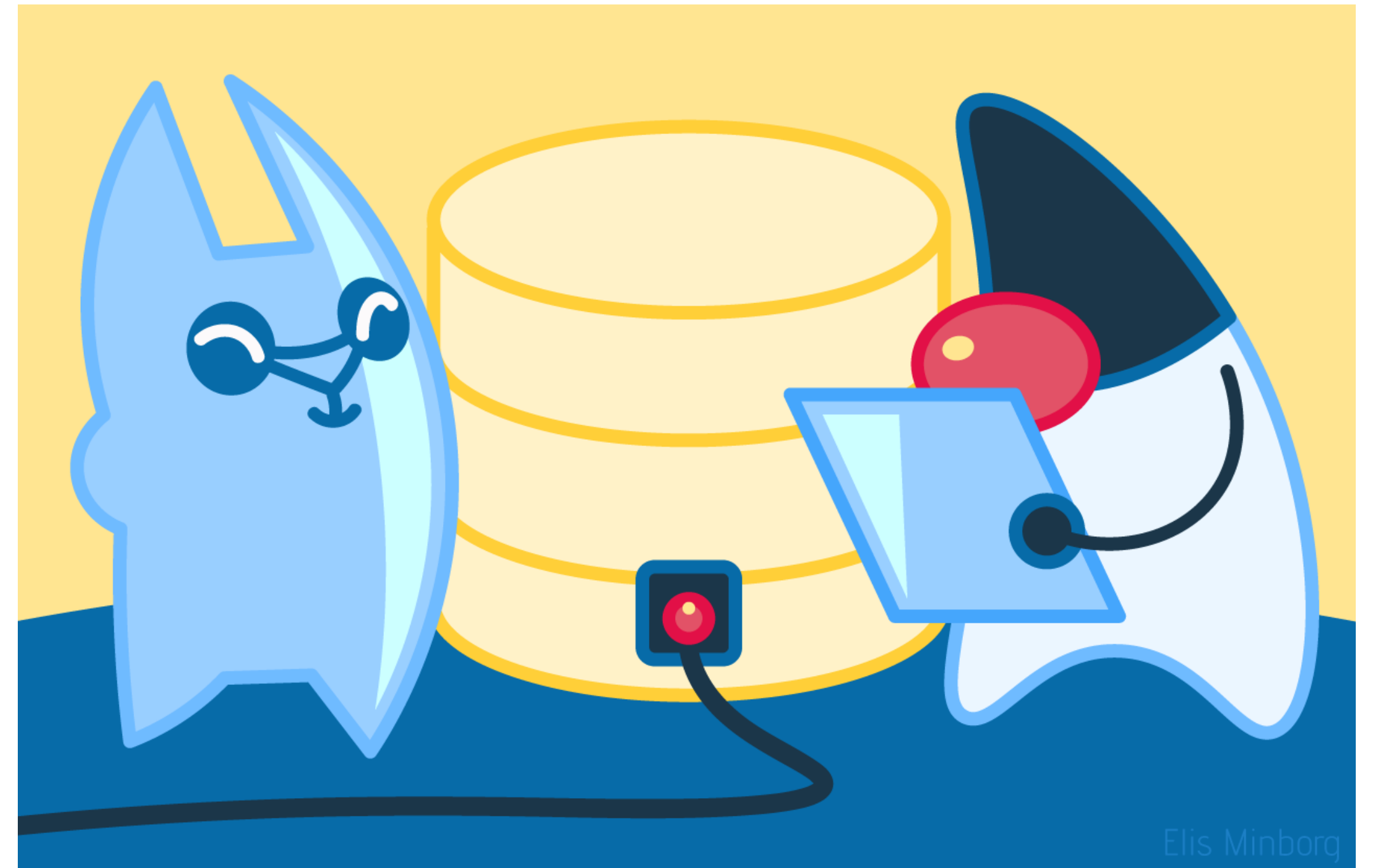
Uses `sun.misc.Unsafe`

- `DirectBuffer.cleaner()` - for predictable cleanup
- `DirectBuffer.address()` - for efficient off-heap byte arrays

Third party dependencies

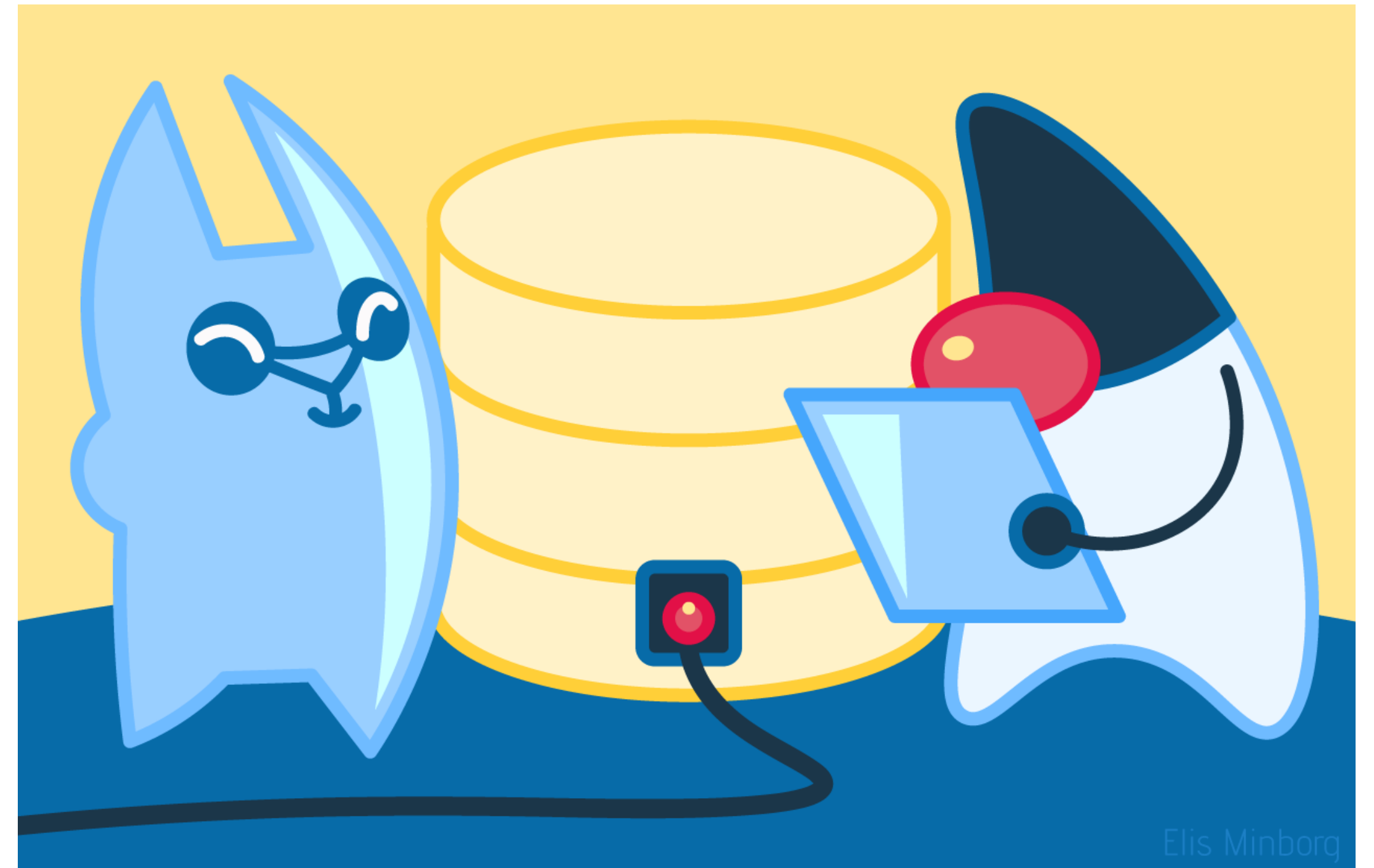
- Depends on Third Party JARs
- Automatic modules with split packages

Awaiting Community Progress



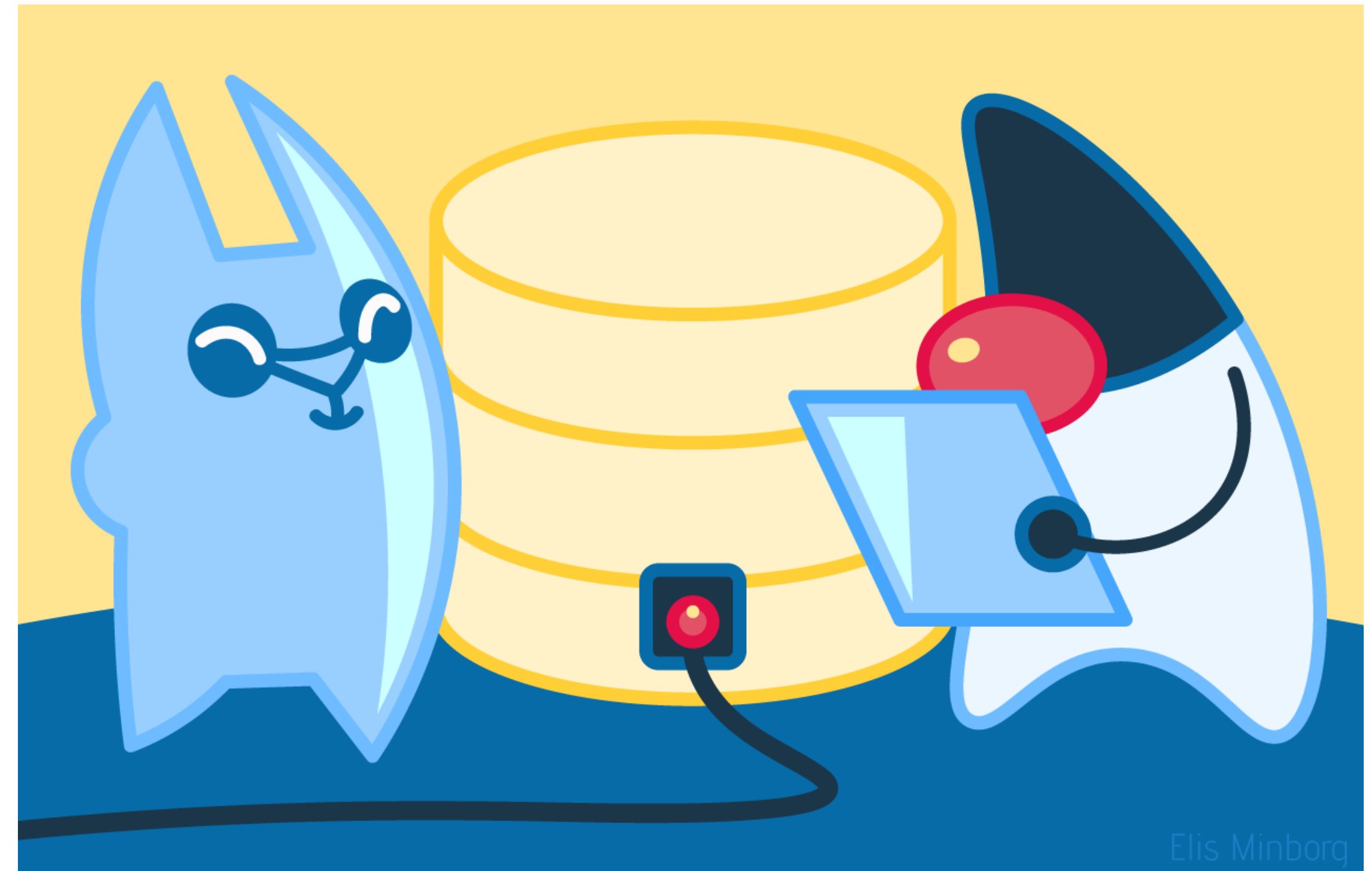
Awaiting Community Progress

- OSGi bundles



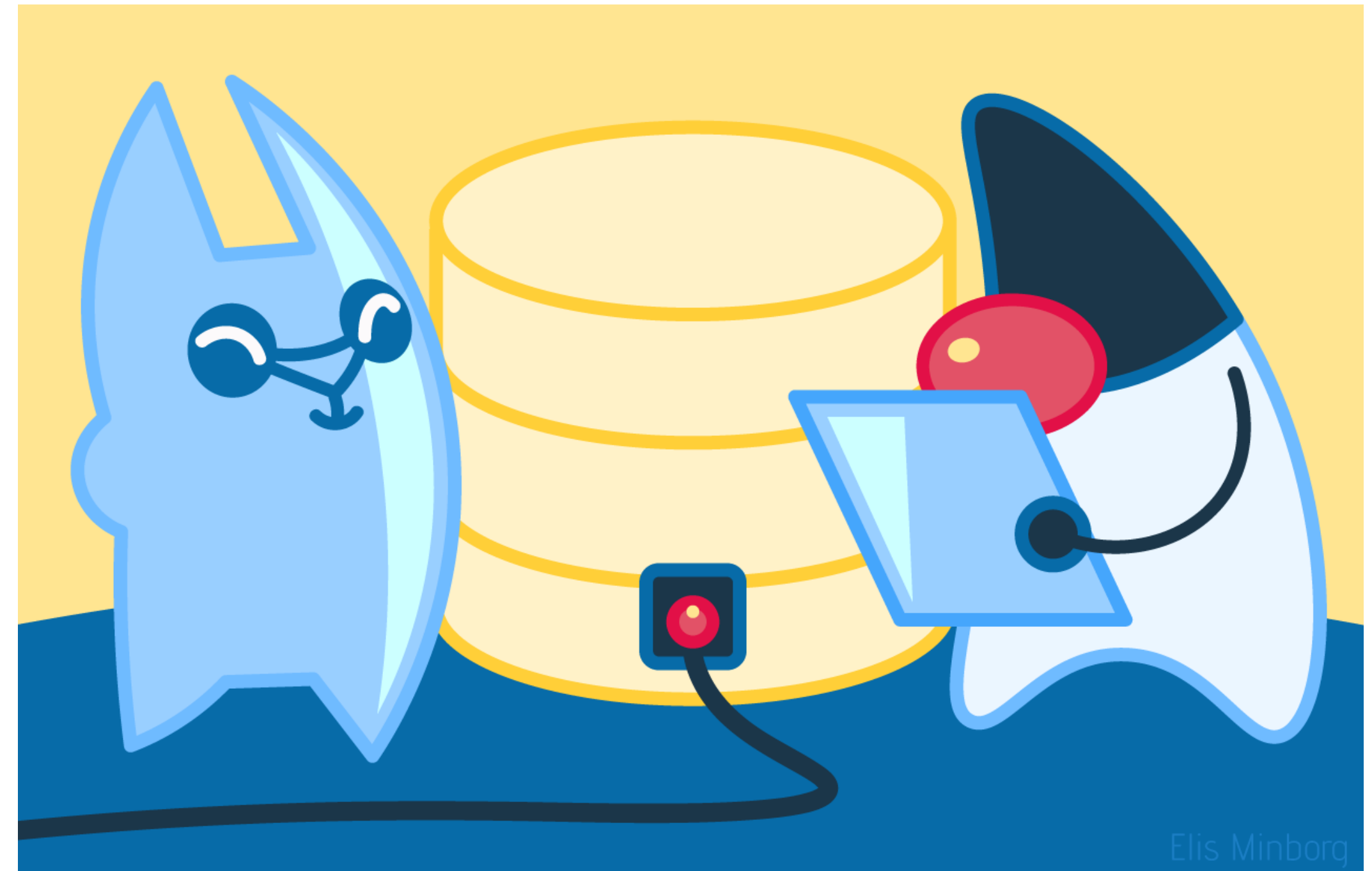
Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies



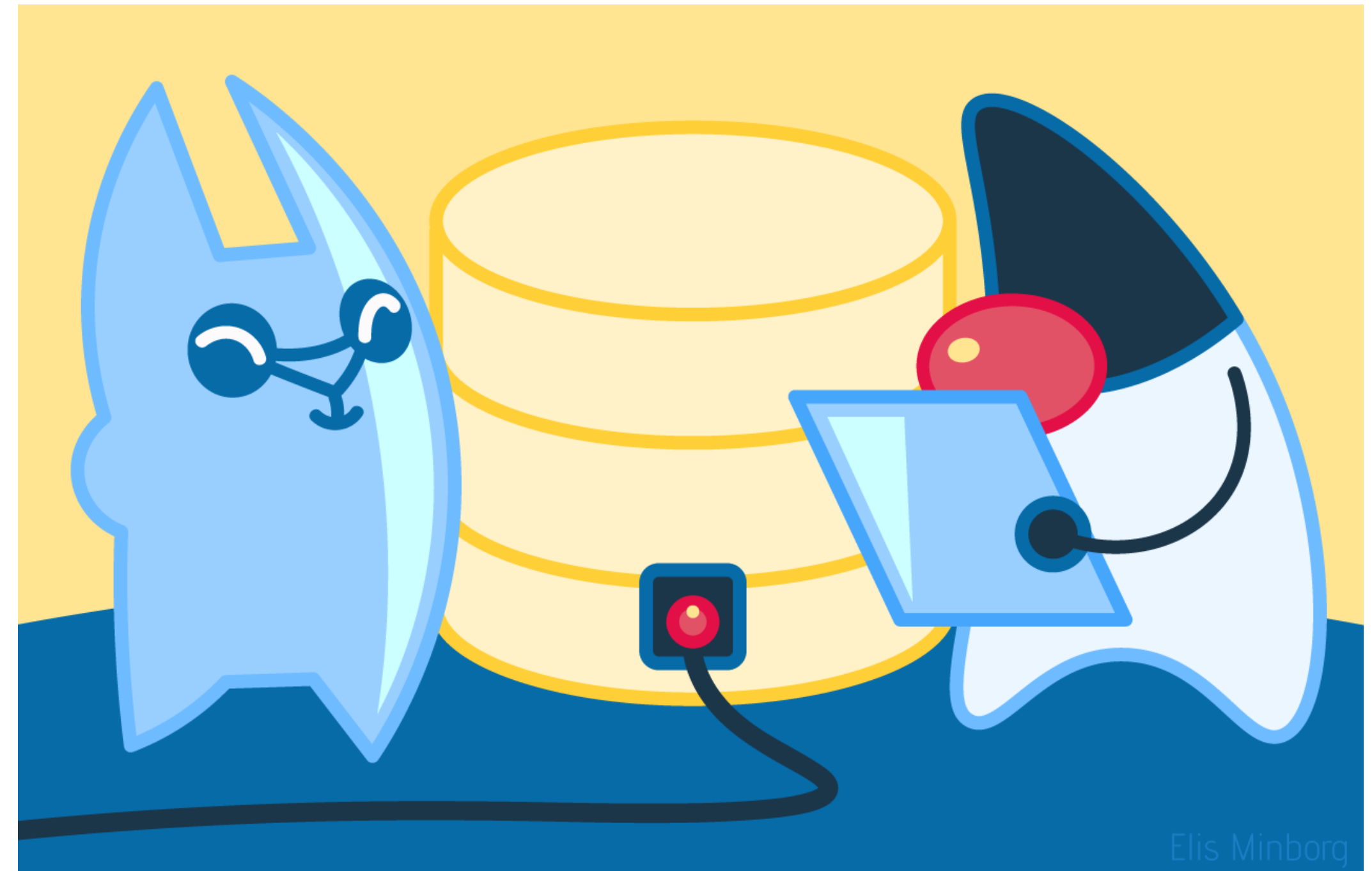
Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
 - Split packages



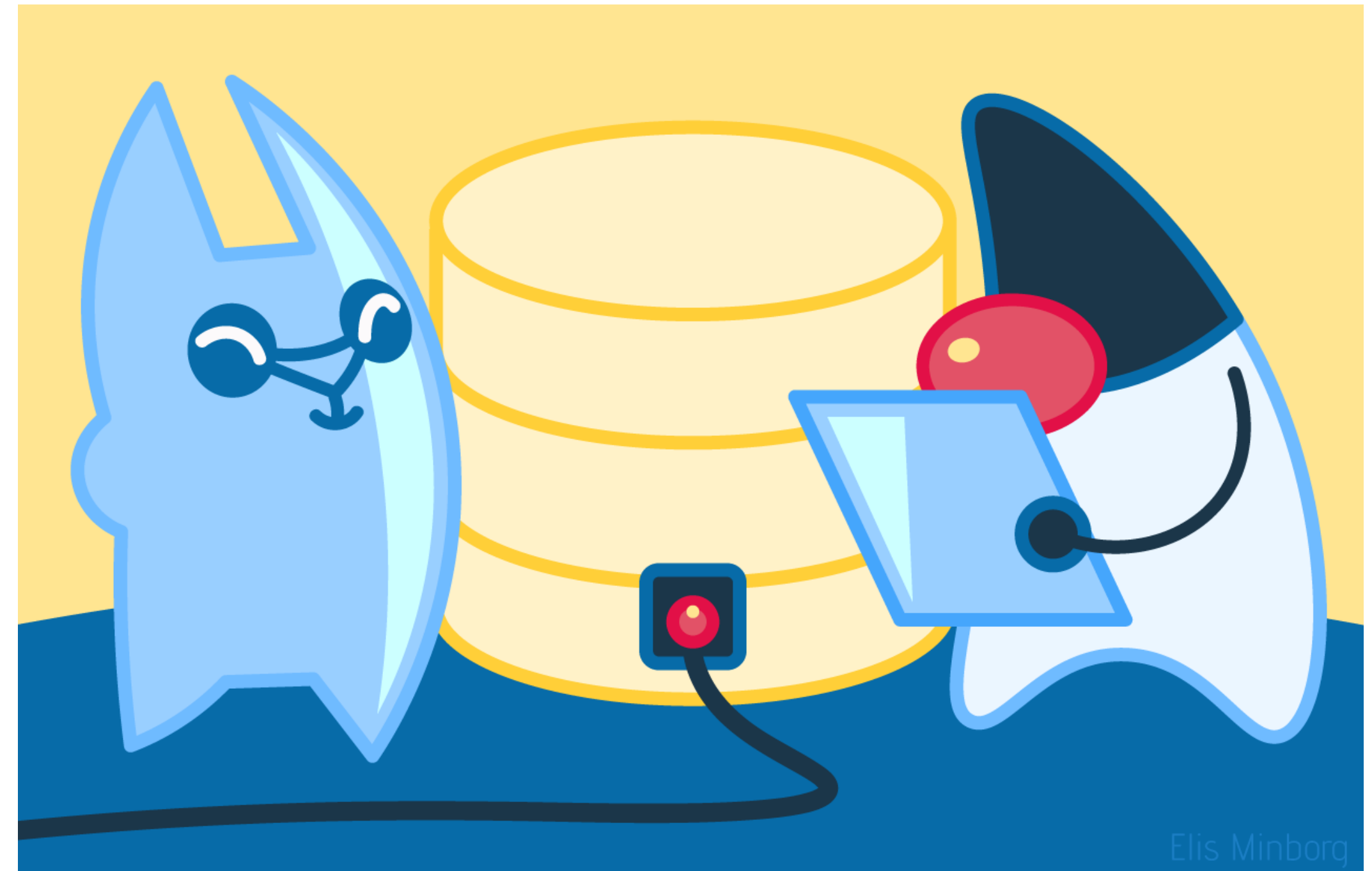
Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
 - Split packages
 - Concealed package conflicts



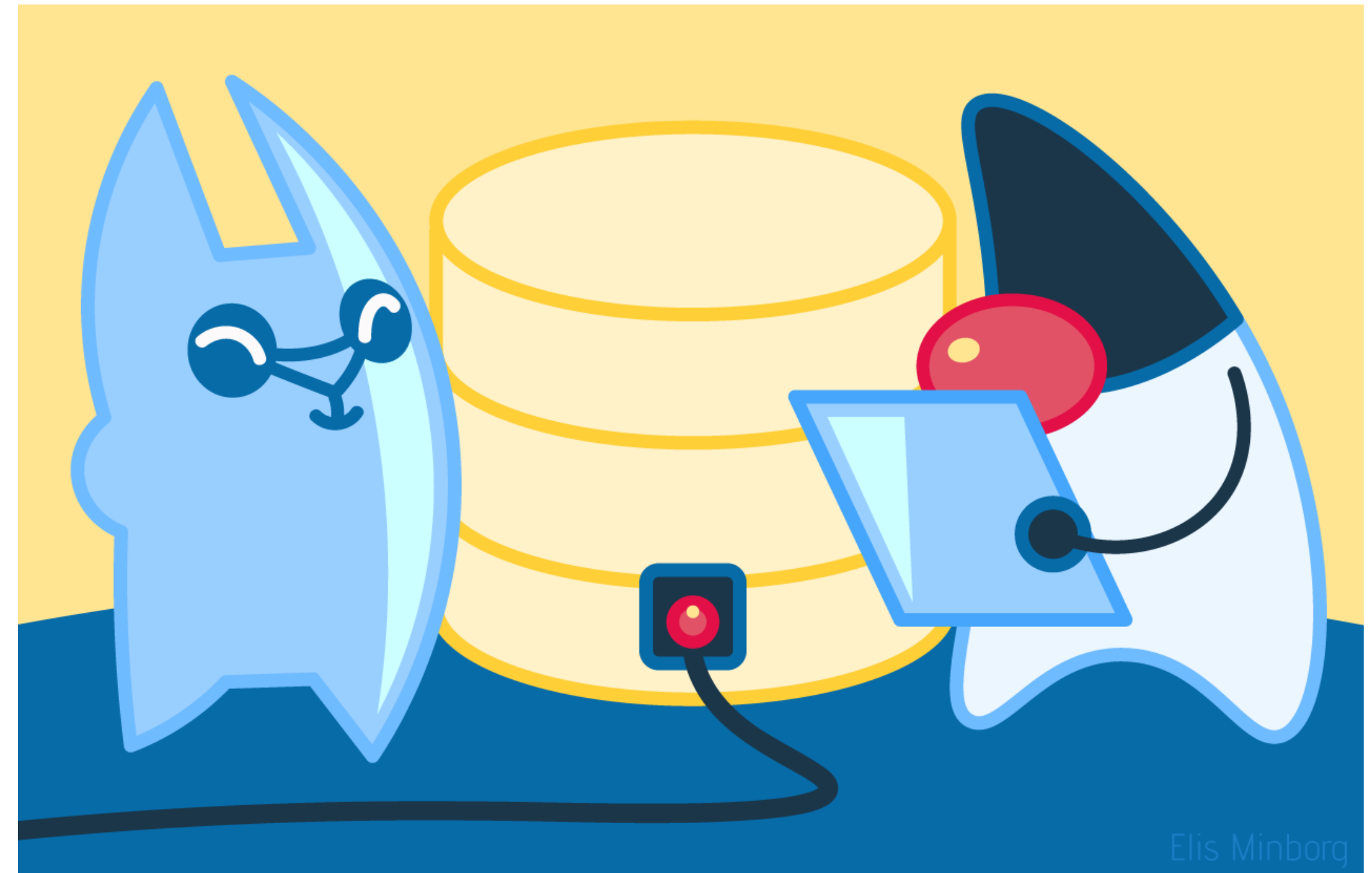
Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
 - Split packages
 - Concealed package conflicts
 - Duplicate package names



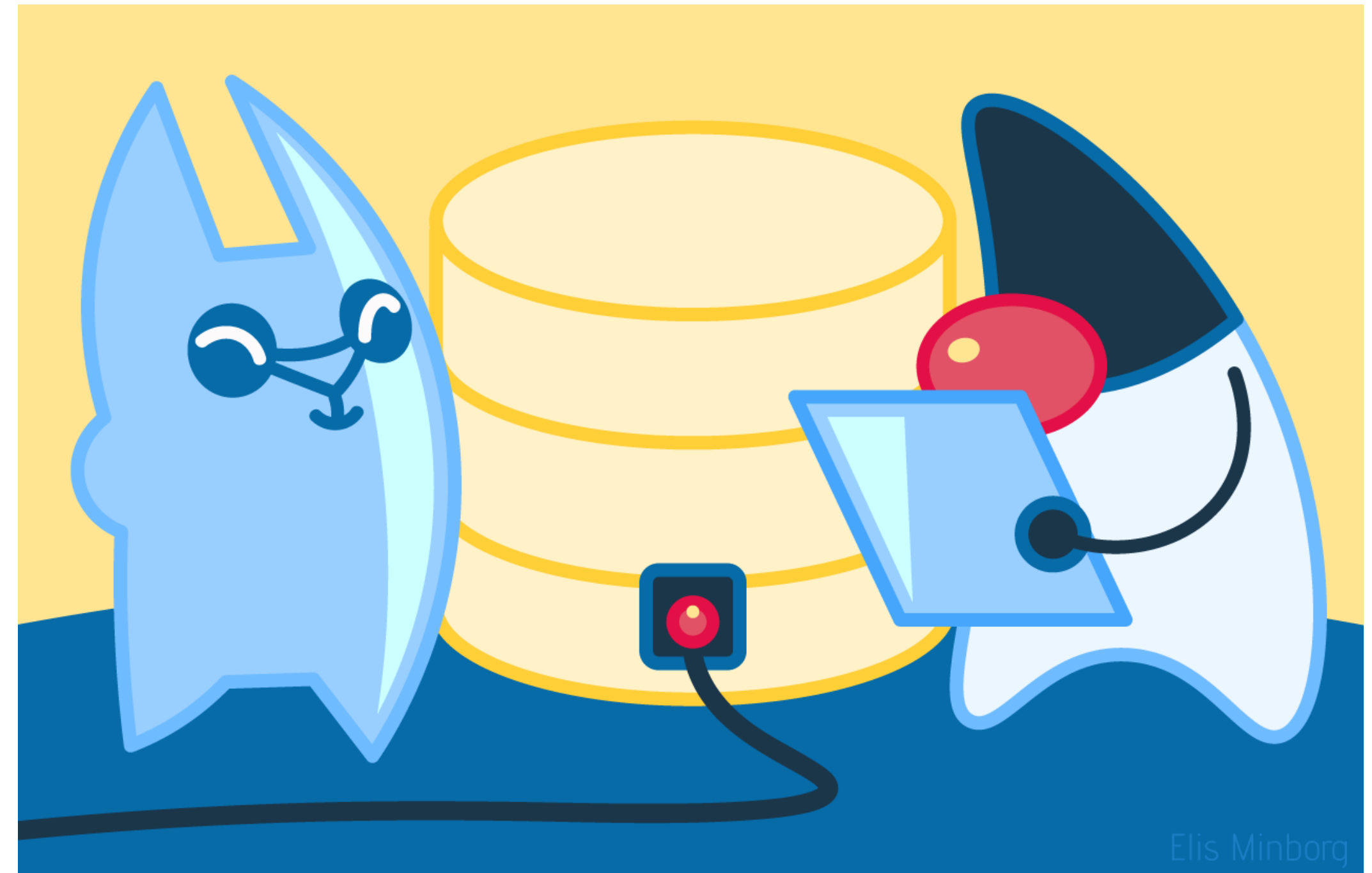
Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
 - Split packages
 - Concealed package conflicts
 - Duplicate package names
- Spring Integration



Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
 - Split packages
 - Concealed package conflicts
 - Duplicate package names
- Spring Integration
- Unit test frameworks



Conclusions

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,
- rework references to non-open parts of the JDK and

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,
- rework references to non-open parts of the JDK and
- revisit design decisions regarding Unsafe.

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,
- rework references to non-open parts of the JDK and
- revisit design decisions regarding Unsafe.

Postpone Decisions (S.E.P.):

- Working around third party dependencies

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,
- rework references to non-open parts of the JDK and
- revisit design decisions regarding Unsafe.

Postpone Decisions (S.E.P.):

- Working around third party dependencies
- Spring Integration

Conclusions

Lean Principle – "Defer Commitment" or "Decide as late as possible"

Time well spent:

- refine application modularity,
- rework references to non-open parts of the JDK and
- revisit design decisions regarding Unsafe.

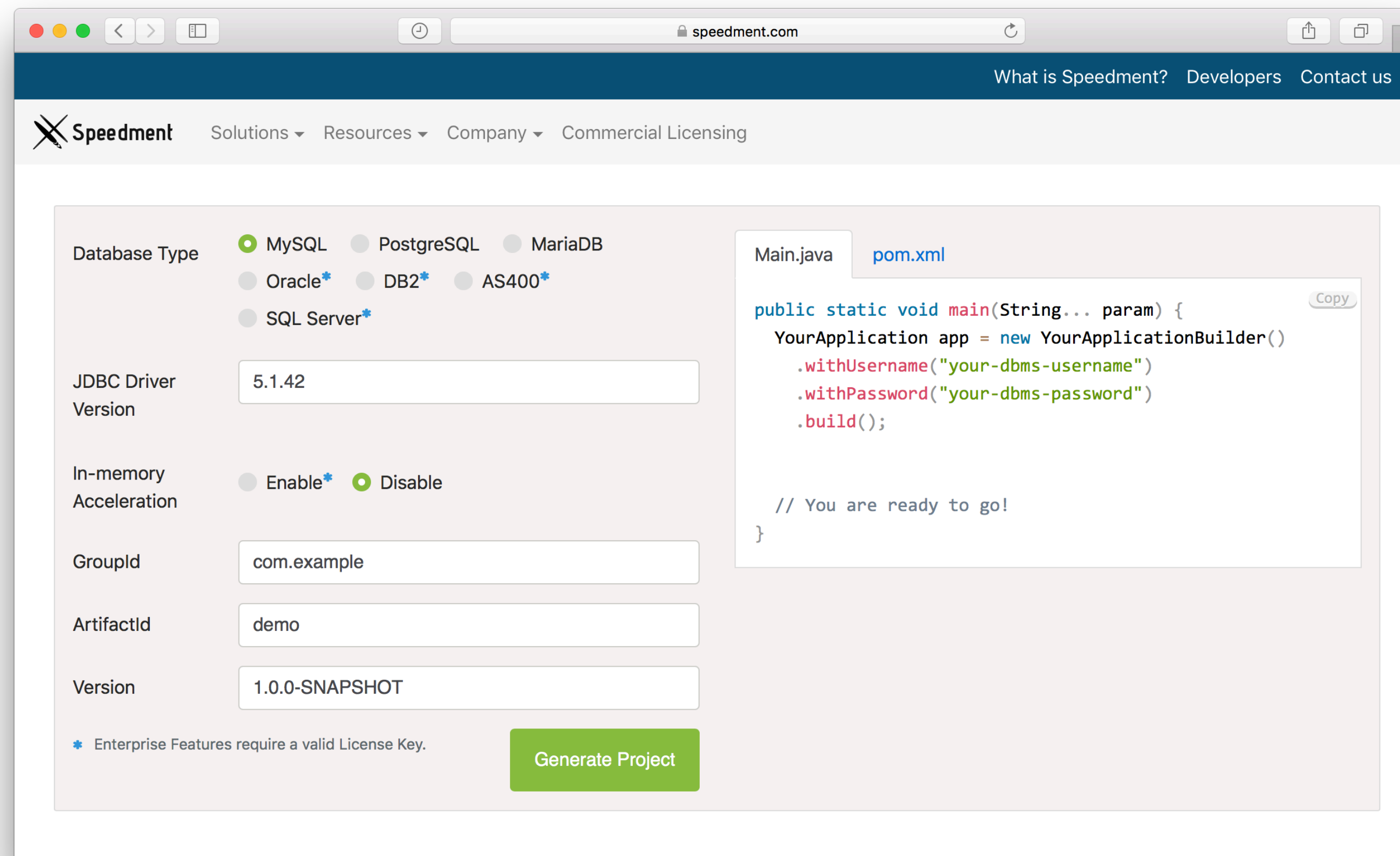
Postpone Decisions (S.E.P.):

- Working around third party dependencies
- Spring Integration
- OSGi bundling

Try Speedment

github.com/speedment/speedment

www.speedment.com/initializer/



The screenshot shows a web browser window displaying the Speedment initializer interface. The browser's address bar shows "speedment.com". The page has a dark blue header with navigation links: "What is Speedment?", "Developers", and "Contact us". Below the header is a navigation bar with the Speedment logo and links for "Solutions", "Resources", "Company", and "Commercial Licensing".

The main content area is a form for configuring a project. It includes the following fields and options:

- Database Type:** Radio buttons for MySQL (selected), PostgreSQL, MariaDB, Oracle*, DB2*, AS400*, and SQL Server*.
- JDBC Driver Version:** Text input field containing "5.1.42".
- In-memory Acceleration:** Radio buttons for Enable* and Disable (selected).
- GroupId:** Text input field containing "com.example".
- ArtifactId:** Text input field containing "demo".
- Version:** Text input field containing "1.0.0-SNAPSHOT".

At the bottom left of the form, there is a note: "* Enterprise Features require a valid License Key." A green "Generate Project" button is located at the bottom right of the form.

On the right side of the form, there is a code editor showing the "Main.java" file. The code is as follows:

```
public static void main(String... param) {
    YourApplication app = new YourApplicationBuilder()
        .withUsername("your-dbms-username")
        .withPassword("your-dbms-password")
        .build();

    // You are ready to go!
}
```


Q&A

@speedment

@dan_lawesson

Extra Slides for Q&A

Many-to-Many

```
Map<Human, List<Hare>> humanFriends = friends.stream()
    .collect(
        groupingBy(humans.finderBy(Friend.HUMAN), // Classifier
            mapping(
                hares.finderBy(Friend.HARE), //Friend to Hare finder
                toList() // Downstream aggregation w. List collector
            )
        )
    );
```

Many-to-Many

```
Map<Human, List<Hare>> humanFriends = friends.stream()
    .collect(
        groupingBy(humans.finderBy(Friend.HUMAN), // Classifier
            mapping(
                hares.finderBy(Friend.HARE), //Friend to Hare finder
                toList() // Downstream aggregation w. List collector
            )
        )
    );
```

Update

```
hares.stream()  
  .filter(Hare.ID.equal(42)) // All Hares with ID = 42 (just one)  
  .map(Hare.AGE.setTo(10))  // Applies a setter  
  .forEach(hares.updater()); // Applies the updater function
```

Update

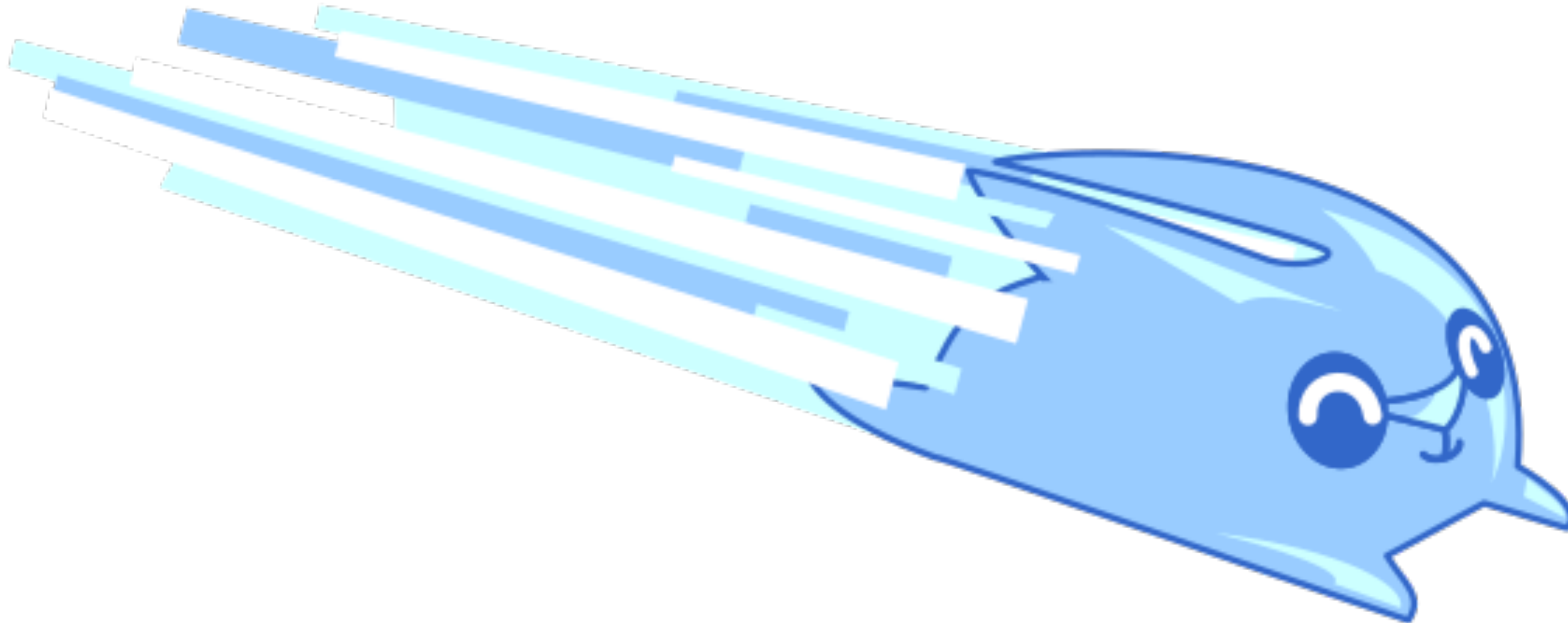
```
hares.stream()  
  .filter(Hare.ID.equal(42)) // All Hares with ID = 42 (just one)  
  .map(Hare.AGE.setTo(10))  // Applies a setter  
  .forEach(hares.updater()); // Applies the updater function
```

Database Connectors



Beyond Open-Source

- Speedment Enterprise
- Run your database queries orders of magnitude faster!
 - 10x, 100x, 1 000x, 10 000x, ...
- Use the same Stream API



How is This Possible?

- No changes to user-written code
- Alternative stream source
- Data is stored in-memory
- Generates optimized serializers for off-heap storage



In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units |
|--------------------------------------|------|-----|-------------------|-------|-------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.join | avgt | 200 | 0.476 ± | 0.005 | s/op |
| SpeedmentBenchmark.joinSql | avgt | 200 | 5.174 ± | 0.010 | s/op |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | 24.661 ± | 0.670 | s/op |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op |
| SpeedmentBenchmark.countSql | avgt | 200 | 5.143 ± | 0.012 | s/op |

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.

In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units |
|--------------------------------------|------|-----|-------------------|---------|-------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.join | avgt | 200 | $0.476 \pm$ | 0.005 | s/op |
| SpeedmentBenchmark.joinSql | avgt | 200 | $5.174 \pm$ | 0.010 | s/op |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | $24.661 \pm$ | 0.670 | s/op |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op |
| SpeedmentBenchmark.countSql | avgt | 200 | $5.143 \pm$ | 0.012 | s/op |

10x

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.

In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units |
|--------------------------------------|------|-----|-------------------|---------|-------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.join | avgt | 200 | $0.476 \pm$ | 0.005 | s/op |
| SpeedmentBenchmark.joinSql | avgt | 200 | $5.174 \pm$ | 0.010 | s/op |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | $24.661 \pm$ | 0.670 | s/op |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op |
| SpeedmentBenchmark.countSql | avgt | 200 | $5.143 \pm$ | 0.012 | s/op |

10x

>10x

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.



In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units | |
|--------------------------------------|------|-----|-------------------|---------|-------|------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op | 10x |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.join | avgt | 200 | $0.476 \pm$ | 0.005 | s/op | >10x |
| SpeedmentBenchmark.joinSql | avgt | 200 | $5.174 \pm$ | 0.010 | s/op | |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op | 100x |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op | |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | $24.661 \pm$ | 0.670 | s/op | |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op | |
| SpeedmentBenchmark.countSql | avgt | 200 | $5.143 \pm$ | 0.012 | s/op | |

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.

In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units | |
|--------------------------------------|------|-----|-------------------|---------|-------|------------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op | 10x |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.join | avgt | 200 | $0.476 \pm$ | 0.005 | s/op | >10x |
| SpeedmentBenchmark.joinSql | avgt | 200 | $5.174 \pm$ | 0.010 | s/op | |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op | 100x |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op | 2,000,000x |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | $24.661 \pm$ | 0.670 | s/op | |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op | |
| SpeedmentBenchmark.countSql | avgt | 200 | $5.143 \pm$ | 0.012 | s/op | |

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.

In-JVM-Memory, JMH Benchmarks(*)

| Benchmark | Mode | Cnt | Score | Error | Units | |
|--------------------------------------|------|-----|-------------------|---------|-------|--------------|
| SpeedmentBenchmark.findStartsWith | avgt | 200 | $\approx 10^{-5}$ | | s/op | 10x |
| SpeedmentBenchmark.findStartsWithSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.join | avgt | 200 | $0.476 \pm$ | 0.005 | s/op | >10x |
| SpeedmentBenchmark.joinSql | avgt | 200 | $5.174 \pm$ | 0.010 | s/op | |
| SpeedmentBenchmark.sort | avgt | 200 | $\approx 10^{-6}$ | | s/op | 100x |
| SpeedmentBenchmark.sortSql | avgt | 200 | $\approx 10^{-4}$ | | s/op | |
| SpeedmentBenchmark.scrollSorted | avgt | 200 | $\approx 10^{-5}$ | | s/op | 2,000,000x |
| SpeedmentBenchmark.scrollSortedSql | avgt | 200 | $24.661 \pm$ | 0.670 | s/op | |
| SpeedmentBenchmark.count | avgt | 180 | $\approx 10^{-8}$ | | s/op | 500,000,000x |
| SpeedmentBenchmark.countSql | avgt | 200 | $5.143 \pm$ | 0.012 | s/op | |

(*) Preliminary results, Mac Book Pro, 2.2 GHz i7, 16GB, MySQL 5.7.16 standard with indexes on all relevant columns.

Gitkraken view of Jigsawing Script Commits

The screenshot displays the Gitkraken interface. On the left, a vertical list of commit messages is shown, with the commit 'enterprise.license.tool: Require javafx.graphics to get access to ja...' selected. On the right, the commit details view is open, showing the commit message, author 'Dan Lawesson', and the commit hash '40ecc5'. Below this, a file tree view shows the path 'license-tool/src/com.speedment.enterprise.license.tool/main/java/module-info.java'.

enterprise.license.runtime: Export com.speedment.enterprise.lice...
enterprise.license.runtime: Export com.speedment.enterprise.lice...
enterprise.license.tool: Require com.speedment.common.json to ...
enterprise.license.tool: Require javafx.graphics to get access to ja...
enterprise.license.tool: Require javafx.controls to get access to jav...
enterprise.license.tool: Require javafx.fxml to get access to javafx...
enterprise.license.tool: Require javafx.base to get access to javafx...
enterprise.license.tool: Require com.speedment.enterprise.licens...
enterprise.license.tool: Require com.speedment.enterprise.datas...
enterprise.license.tool: Require com.speedment.common.rest to ...
enterprise.license.tool: Require com.speedment.common.logger t...
enterprise.license.tool: Require com.speedment.common.injector...
datastore-tool: Workaround: using jar packaging since bundle pac...
enterprise.datastore.tool: Require javafx.graphics to get access to...
enterprise.datastore.generator: Export com.speedment.enterpris...
enterprise.datastore.tool: Require javafx.base to get access to jav...
enterprise.datastore.tool: Require com.speedment.tool.propertye...
enterprise.datastore.tool: Require com.speedment.tool.config to

enterprise.license.tool: Require javafx.graphics to get access to javafx.stage

Dan Lawesson
authored 2017-6-20 @ 10:26
commit: 40ecc5
parent: c2a745

1 modified

Name ▲ Collapse All Tree View ☰

- license-tool
 - src
 - com.speedment.enterprise.license.tool
 - main
 - java
 - module-info.java

Gitkraken view of Jigsawing Script Commits

The screenshot displays the Gitkraken interface. The top-left pane shows a diff for the file `license-tool/src/com.speedment.enterprise.license.tool/main/java/module-info.java`. The diff highlights the addition of `requires javafx.graphics;` on line 2. The top-right pane shows the commit message: `enterprise.license.tool: Require javafx.graphics to get access to javafx.stage`. Below the message, the author is identified as Dan Lawesson, committed on 2017-6-20 at 10:26. The bottom-right pane shows a file tree with the following structure:

- license-tool
 - src
 - com.speedment.enterprise.license.tool
 - main
 - java
 - module-info.java