QCon NEW YORK

Migrating Speedment to Java 9

Dan Lawesson, @dan_lawesson
CSO, Speedment, Inc

Speedment

About Us

Speedment

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

Speedment

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

Speedment

Outline

Speedment

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

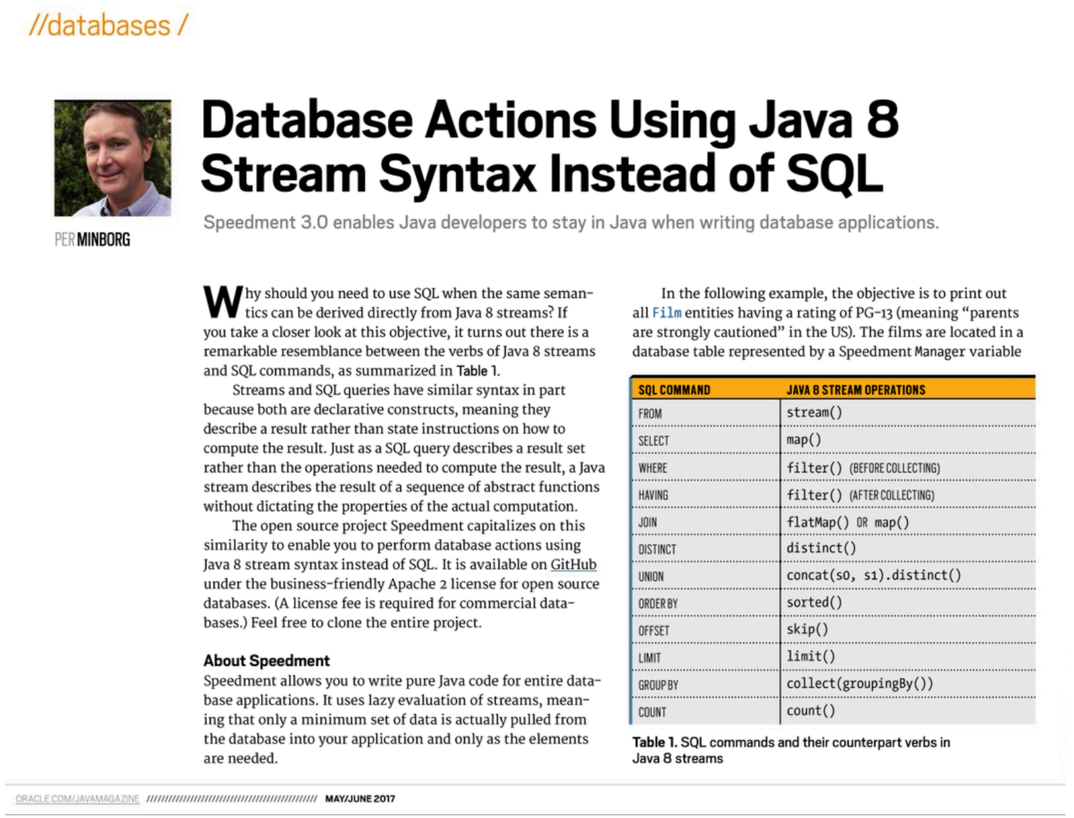
Oracle Java Magazine May-June pages 34-40

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.

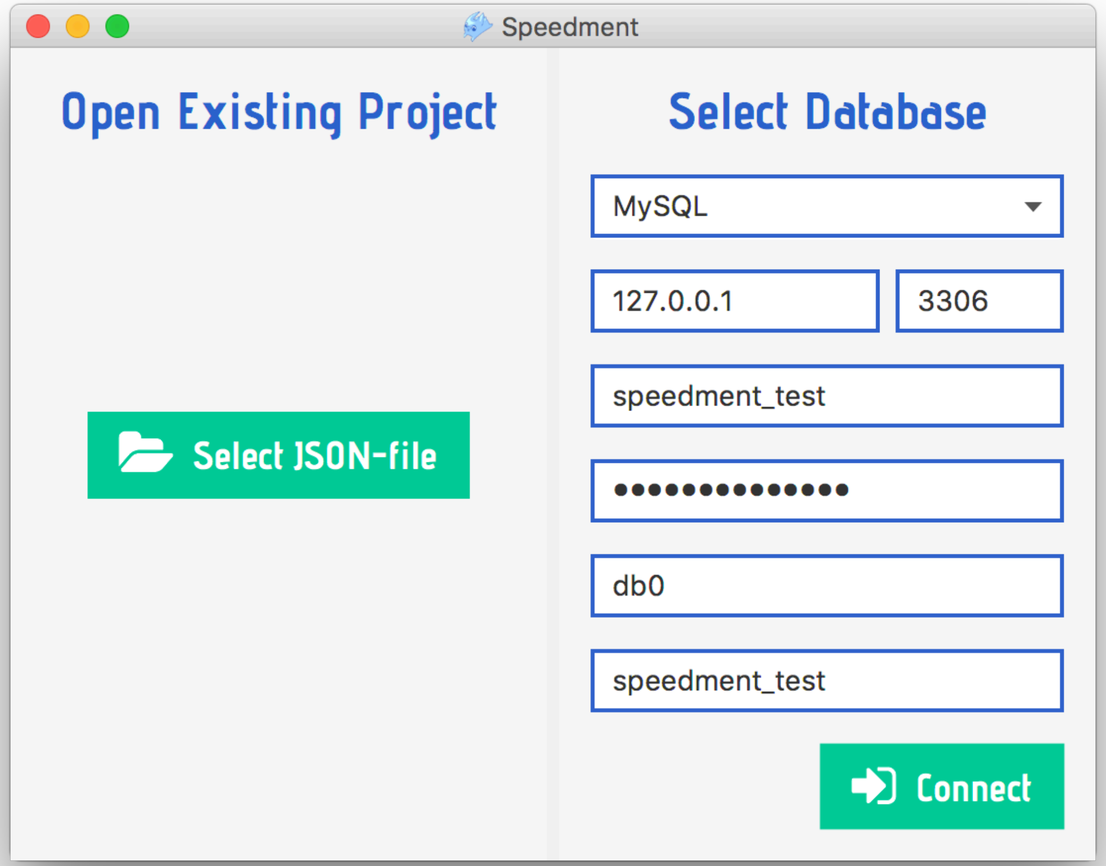
In the following example, the objective is to print out all `Film` entities having a rating of PG-13 (meaning "parents are strongly cautioned" in the US). The films are located in a database table represented by a Speedment `Manager` variable

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

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

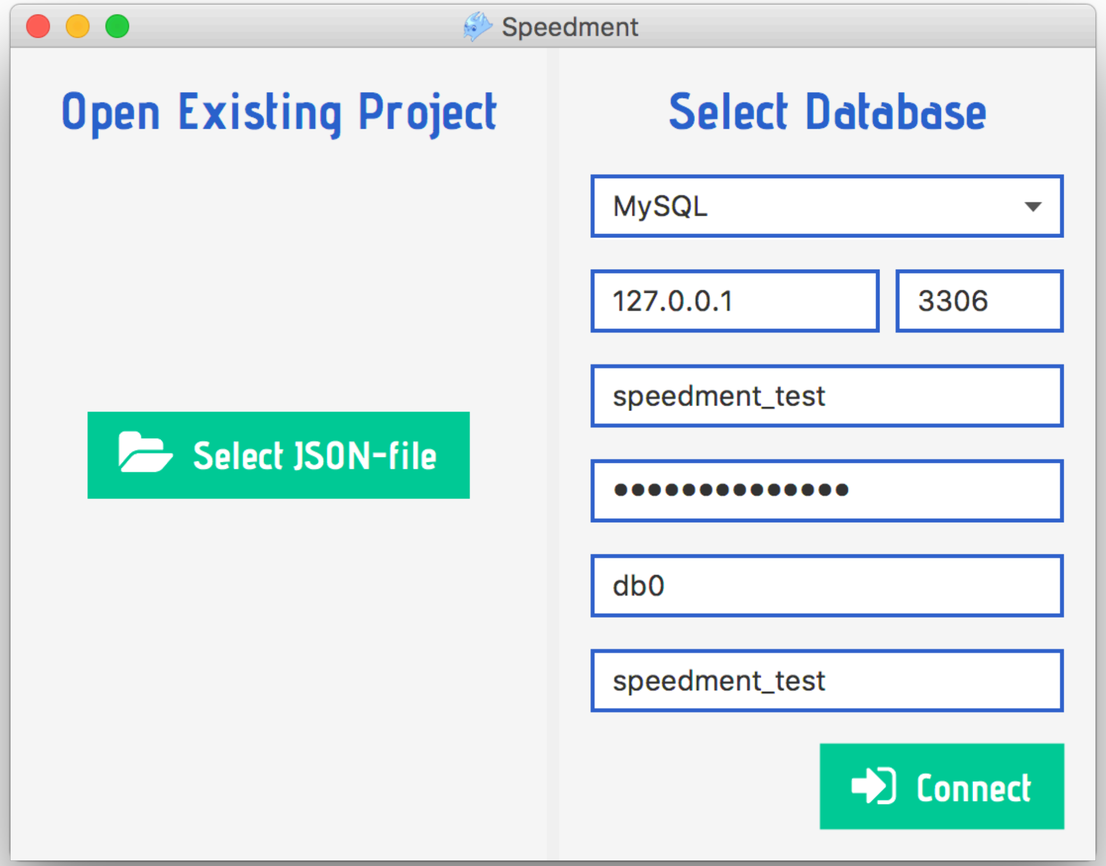
Speedment

Compile-time Code Generation Tool

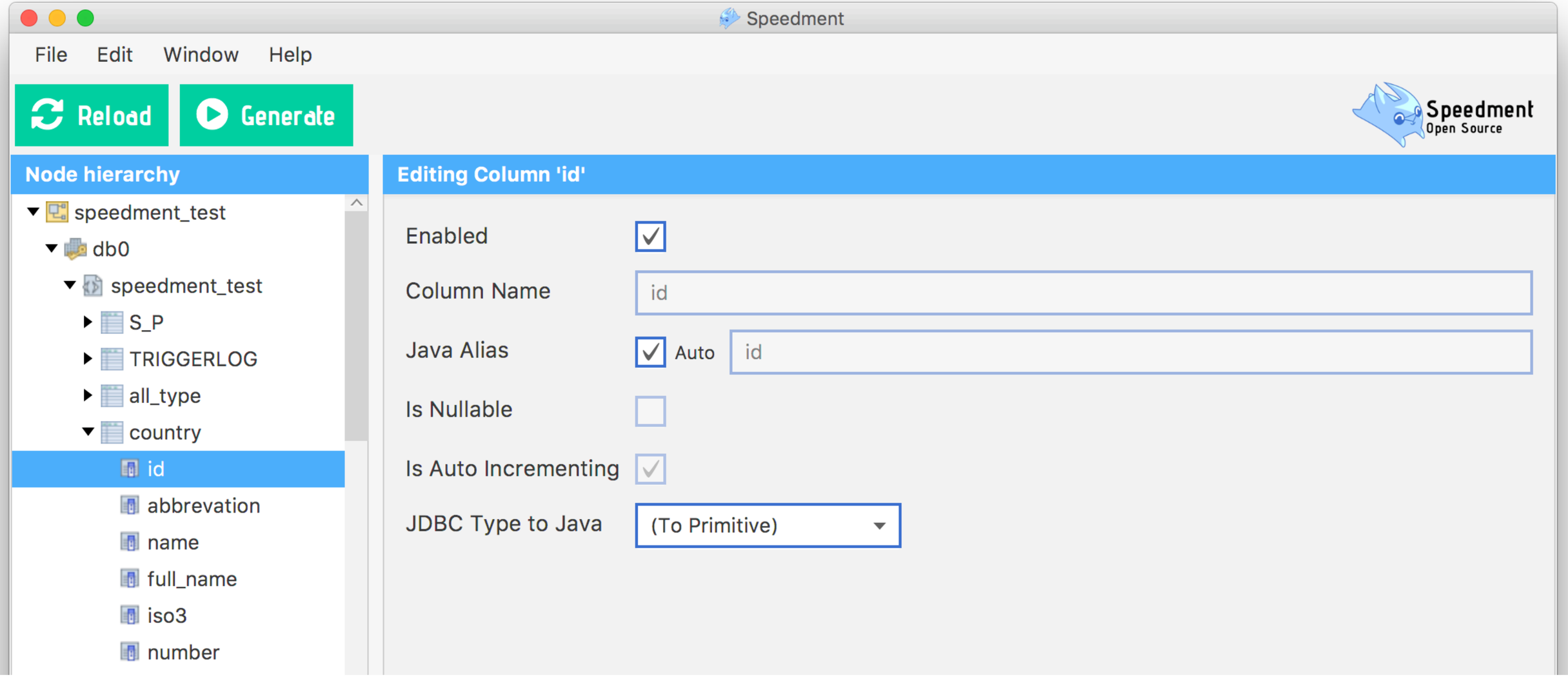


Speedment

Compile-time Code Generation Tool



Speedment


Speedment
File
Edit
Window
Help
Reload
Generate
Speedment
Open Source
Node hierarchy
speedment_test
db0
speedment_test
S_P
TRIGGERLOG
all_type
country
id
abbrevation
name
full_name
iso3
number
Editing Column 'id'
Enabled
Column Name
id
Java Alias
Auto
id
Is Nullable
Is Auto Incrementing
JDBC Type to Java
(To Primitive)

Speedment

Querying the Database using Streams

Speedment

Querying the Database using Streams

- Queries are expressed using the standard Java 8 Stream API

Speedment

Querying the Database using Streams

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

Speedment

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();
```

Speedment

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();
```

Speedment

Expressing Queries as Streams

Standard Stream API

Full Type-Safety

Generated Enum Constants

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

Speedment

Expressing Queries as Streams

Standard Stream API

Full Type-Safety

Generated Enum Constants

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

Only 1 value is loaded from DB

Speedment

Expressing Queries as Streams

Standard Stream API

Generated Enum Constants

Full Type-Safety

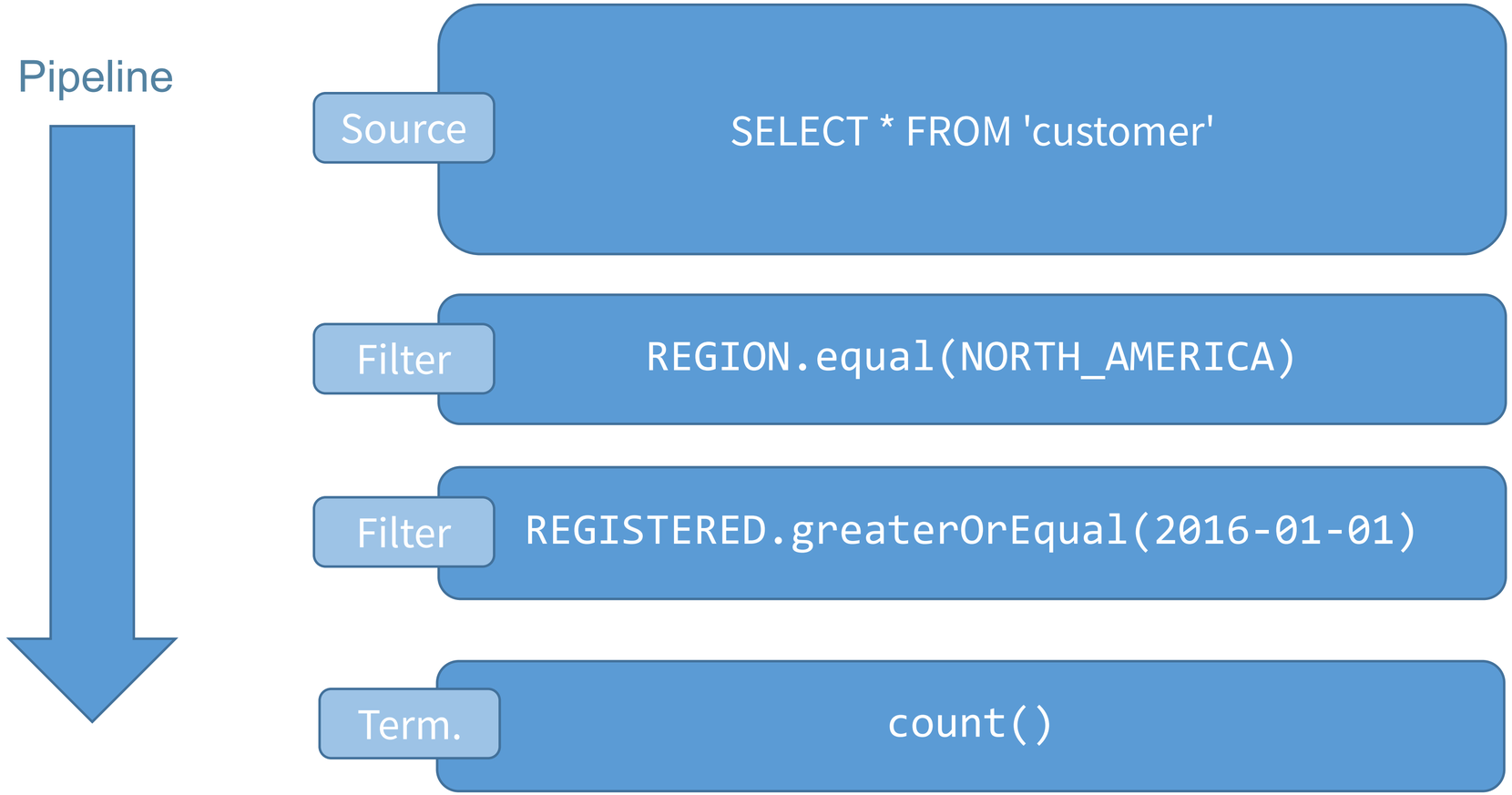
```
customers.stream()
    .filter(Customer.REGION.equal(Region.NORTH_AMERICA))
    .filter(Customer.REGISTERED.greaterOrEqual(startOfYear))
    .count();
```

Only 1 value is loaded from DB

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

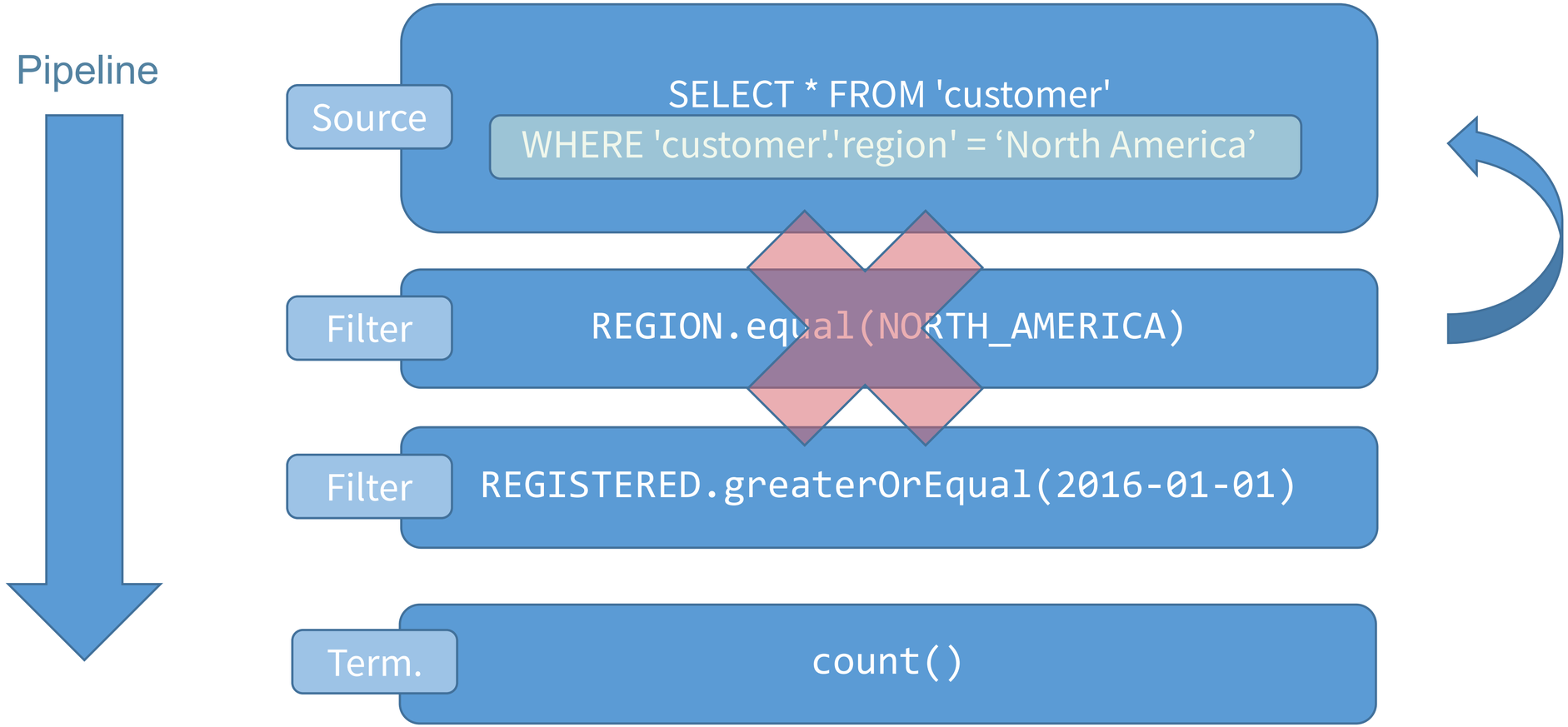
Speedment

Querying the Database using Streams



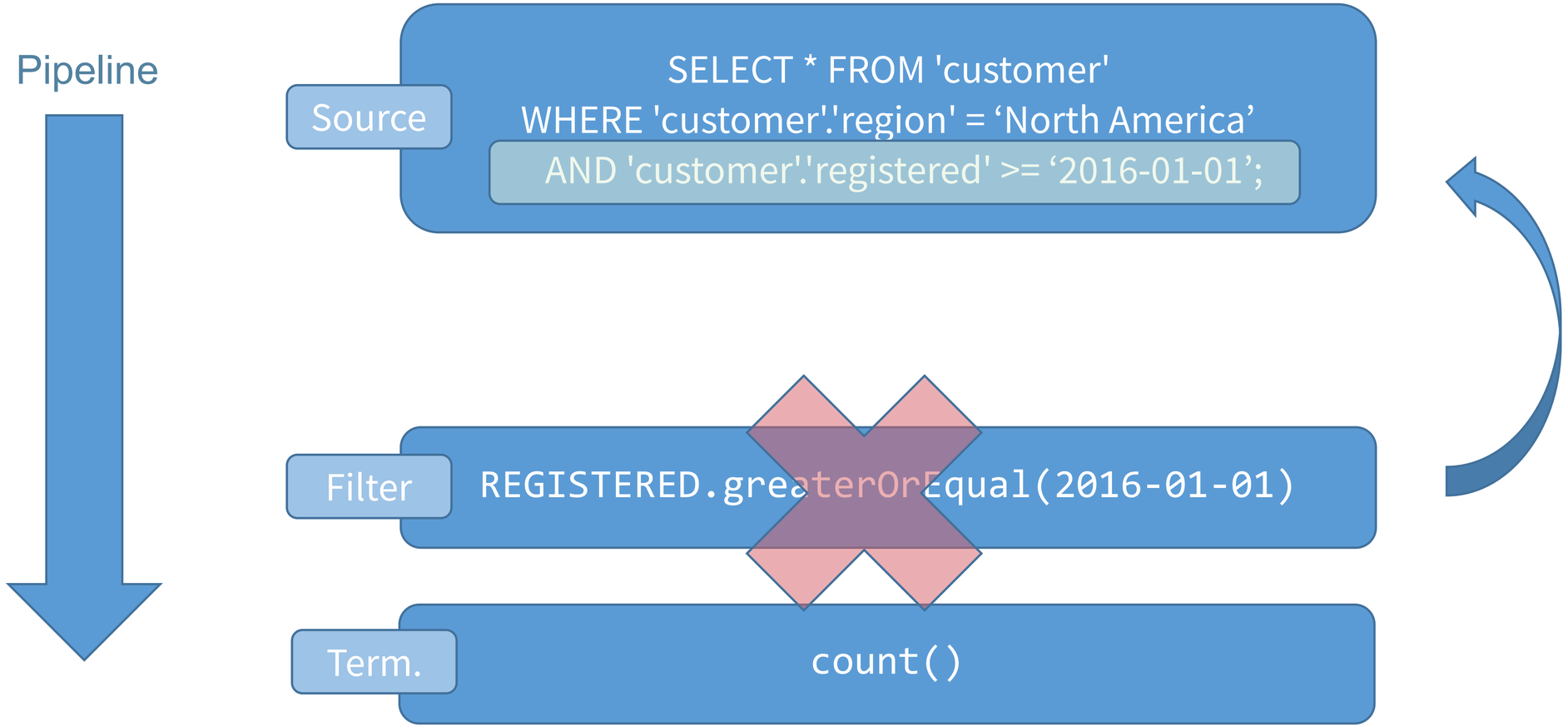
Speedment

Querying the Database using Streams



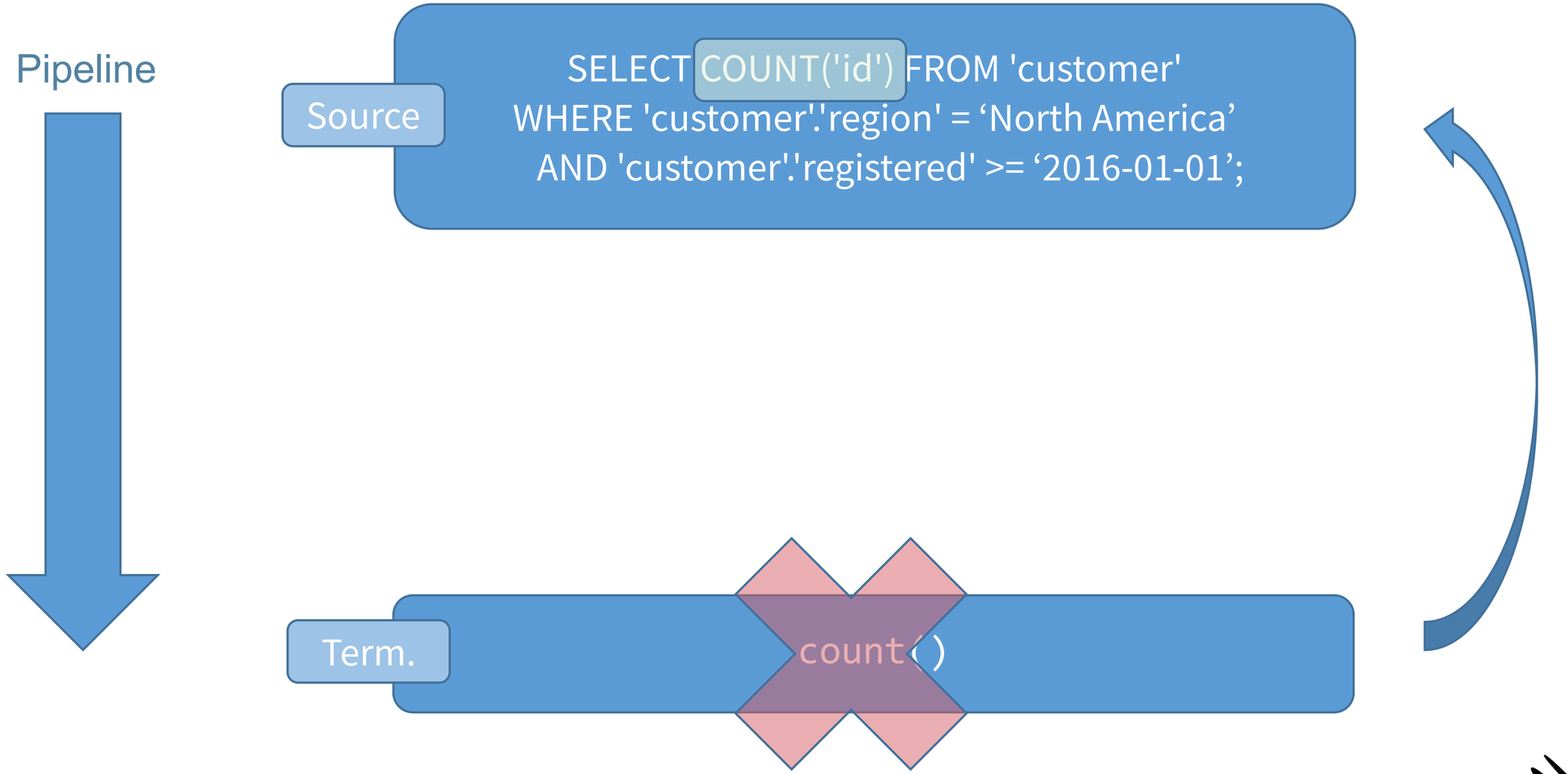
Speedment

Querying the Database using Streams



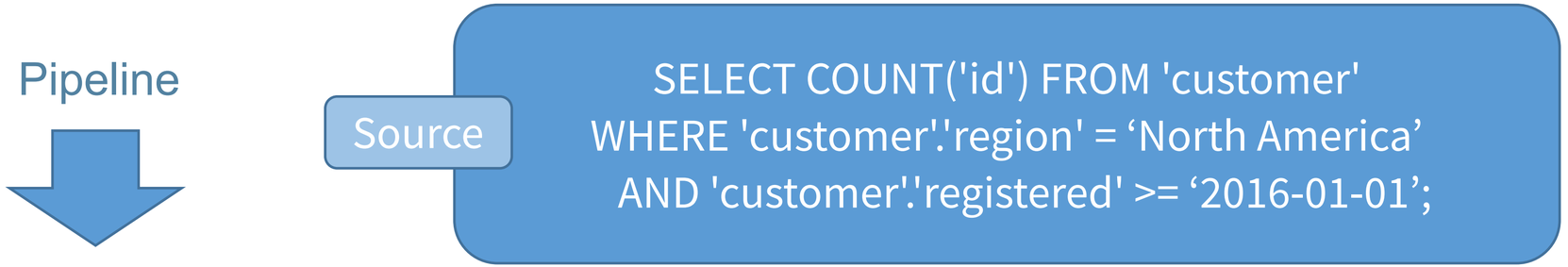
Speedment

Querying the Database using Streams



Speedment

Querying the Database using Streams



Speedment

Expressing Queries as Streams

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

Speedment

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”: …},
  {…},
  …
]
```

Speedment

Parallelism as Expected for a Stream

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

Speedment

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(expensiveOperatation());
```

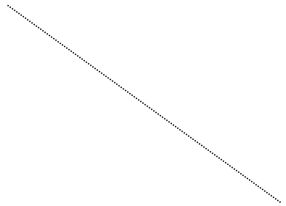
Speedment

Querying

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

Querying

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

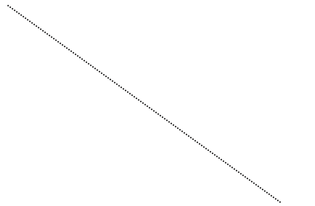


Predicate Builder

Speedment

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;
```

Speedment

Entities are Linked

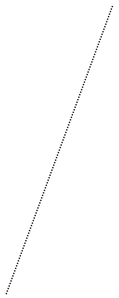
```
// Find the owner of the orange carrot
Optional<Hare> hare = carrots.stream()
 .filter(Carrot.NAME.equal("Orange"))
 .map(hares.finderBy(Carrot.OWNER))
 .findAny();

// 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))
    );
```

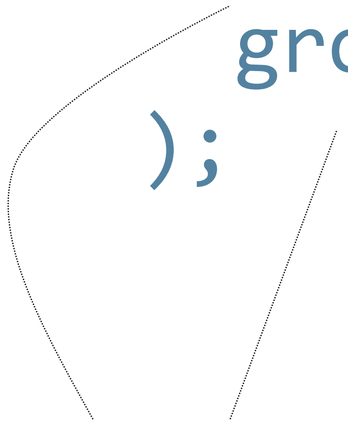


Standard Java 8

Speedment

Joins

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



Standard Java 8

Speedment

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`

Speedment

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

Speedment - Breaking the Language Barrier

Speedment

Speedment - Breaking the Language Barrier

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

Speedment

Speedment - Breaking the Language Barrier

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

the program is *declarative*,

Speedment

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

Speedment - Fully Declarative DB Applications

Speedment

Speedment - Fully Declarative DB Applications

Fully declarative ->

Speedment

Speedment - Fully Declarative DB Applications

Fully declarative ->

Speedment Enterprise: add in-memory-acceleration

Speedment

Speedment - Fully Declarative DB Applications

Fully declarative ->

Speedment Enterprise: add in-memory-acceleration

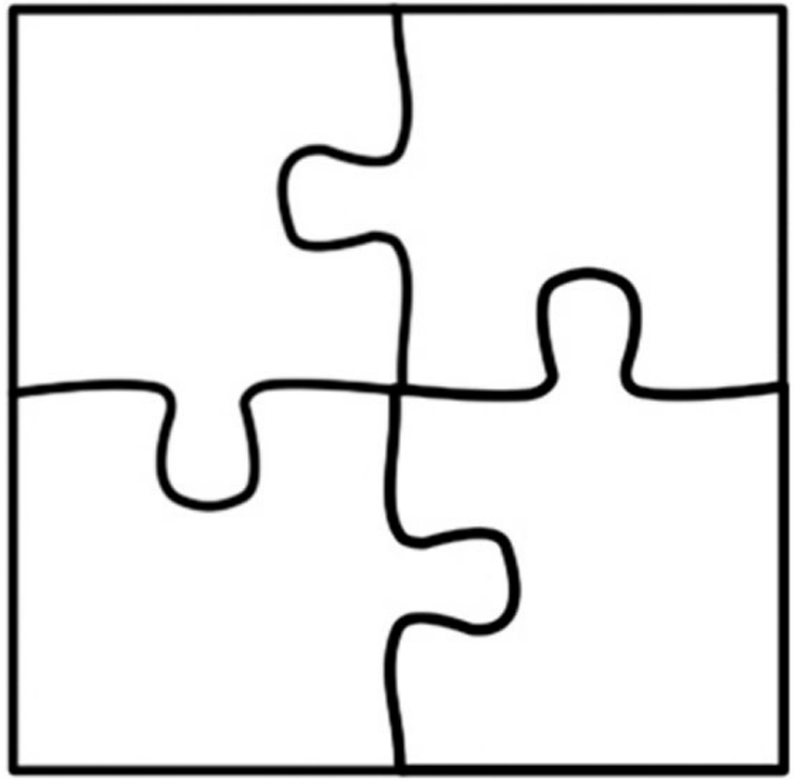
without changing anything in the stream application.

Speedment

Jigsaw

Speedment

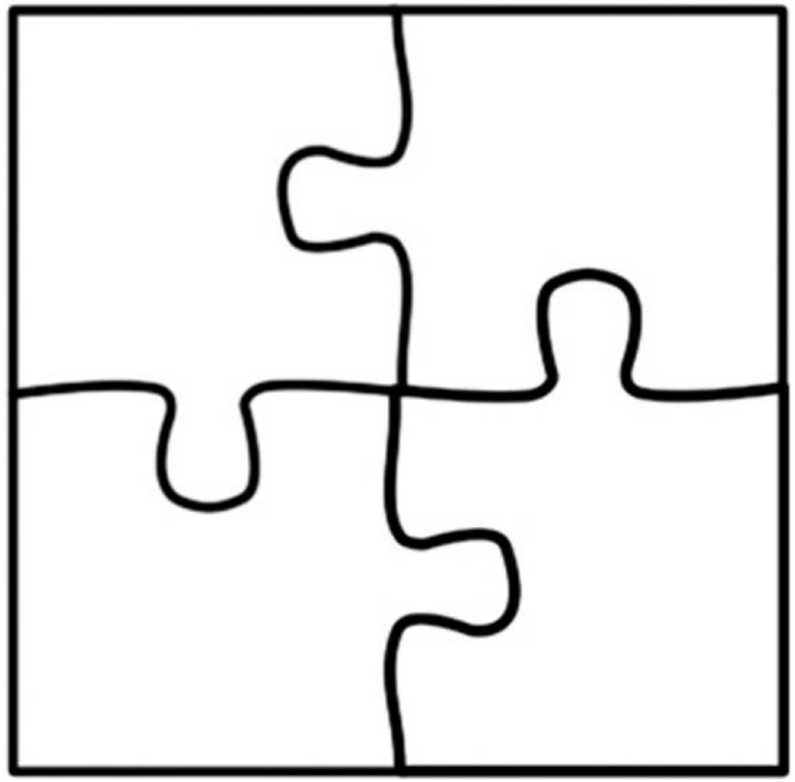
Jigsaw



Speedment

Jigsaw

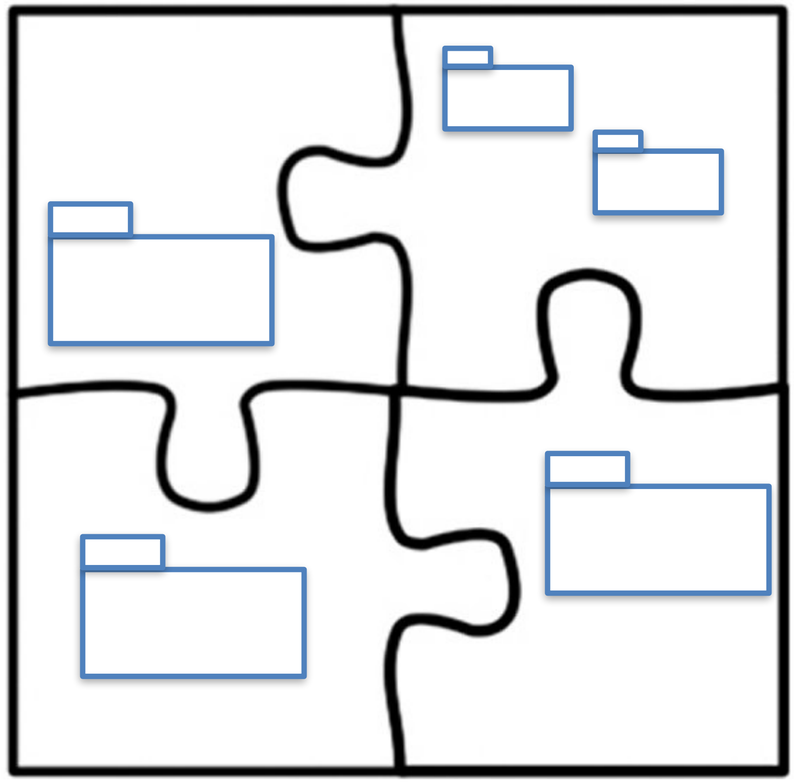
- Modules - an abstraction on top of packages
  - Strong Encapsulation
  - Reliable Configuration



Speedment

Jigsaw

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



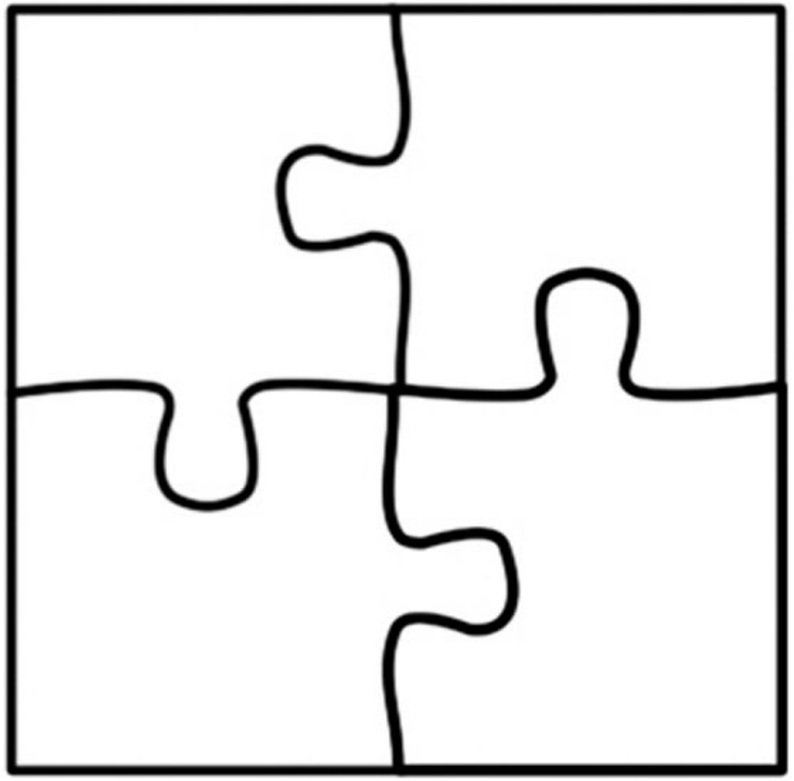
Speedment

Split Packages

Reliable configuration - a package may only belong to one module

Split Packages

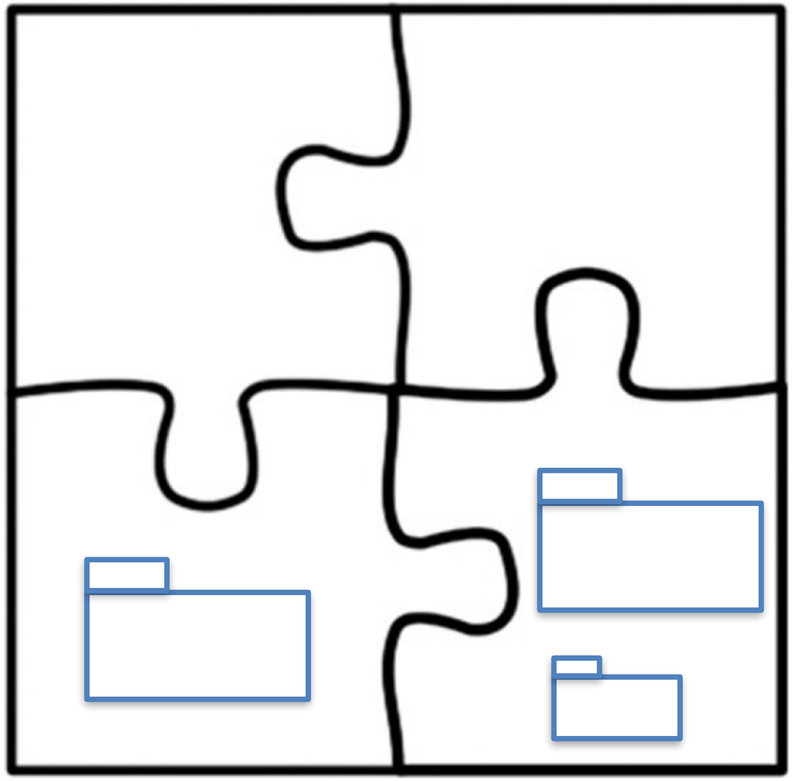
Reliable configuration - a package may only belong to one module



Speedment

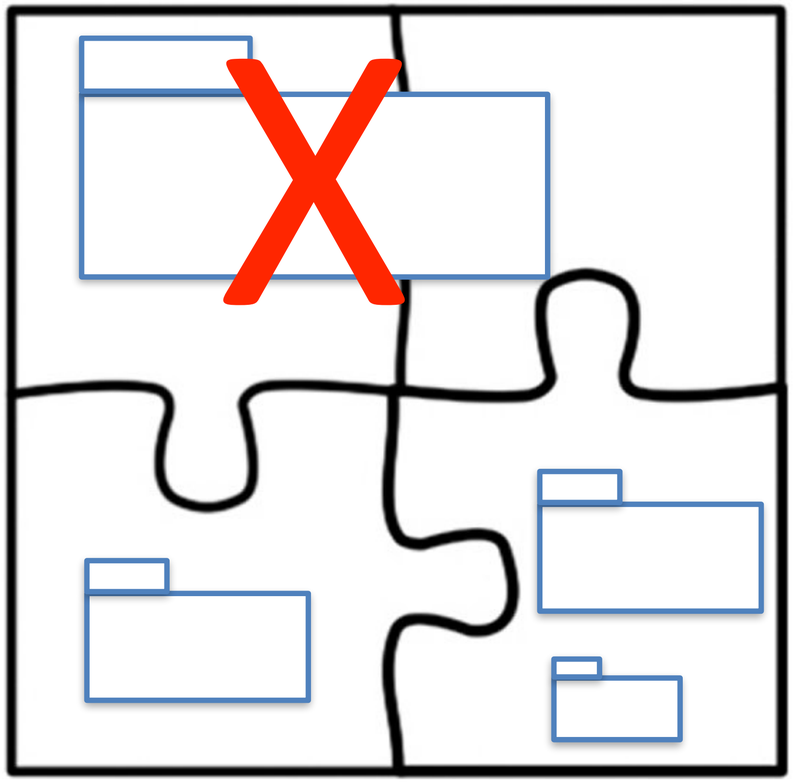
Split Packages

Reliable configuration - a package may only belong to one module



Split Packages

Reliable configuration - a package may only belong to one module



Speedment

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**

$\perp$

Some Other Jigsaw Controversies

Some Other Jigsaw Controversies

- Works for JDK `!=` works for applications
  - Restricts current application domain use cases

Some Other Jigsaw Controversies

- Works for JDK `!=` works for applications
  - Restricts current application domain use cases

- sun.misc.Unsafe
  - ”Should not be used” **vs** ”A key for Java real-world success”

Speedment

Some Other Jigsaw Controversies

- Works for JDK `!=` 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

Speedment

Jigsawing the Java 8 Application

- Run Java 8 under Java 9 is super easy

Speedment

Jigsawing the Java 8 Application

- Run Java 8 under Java 9 is super easy

```
java -cp <…> -jar app.jar
```

Speedment

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

Speedment

Inter module dependencies

Speedment

Inter module dependencies

Dependencies explicitly given in module-info.java

Speedment

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
}
```

Speedment

A Straight-Forward Modularization Approach

class path

module path

Speedment

A Straight-Forward Modularization Approach

	class path	module path
1. All JARs on class path	`<all jars>`	

Speedment

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`

Speedment

A Straight-Forward Modularization Approach

1. All JARs on class path
2. App as a monolithic module
3. Declare dependencies

	class path	module path
1.	`<all jars>`	
2.	`<all deps>`	`myapp.jar`
3.		

Speedment

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>`	`->`

Speedment

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		

Speedment

The Speedment Java 8 Modules

The Speedment Java 8 Modules

Speedment pom.xml

Speedment

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

Speedment

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

Speedment

Automatic Jigsawing Script Outline

- Create separate directories for the modules

Speedment

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

Speedment

Automatic Jigsawing - add requires

Speedment

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;
 }
```

Speedment

Automatic Jigsawing - add exports

Speedment

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;
 }
```

Speedment

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>
```

Speedment

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>
…
```

Speedment

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

Speedment

No Reflection in Speedment

- O/R mapping -> reflection

Speedment

No Reflection in Speedment

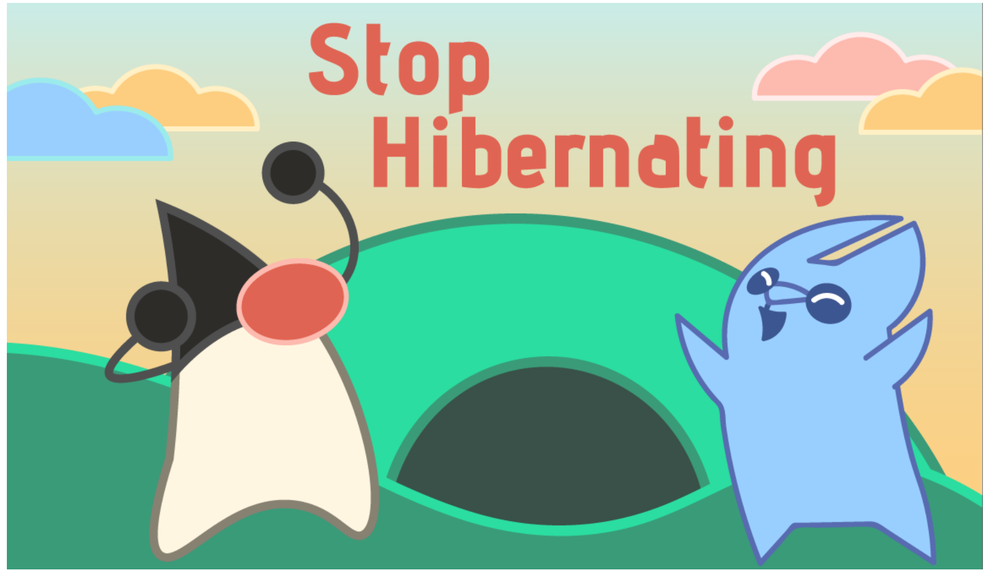
- O/R mapping -> reflection
  - instantiate user code

Speedment

No Reflection in Speedment

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

- Code Generation to the Rescue!



Speedment

Speedment Enterprise - a Jigsaw Puzzle

Speedment

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

Speedment Enterprise - a Jigsaw Puzzle

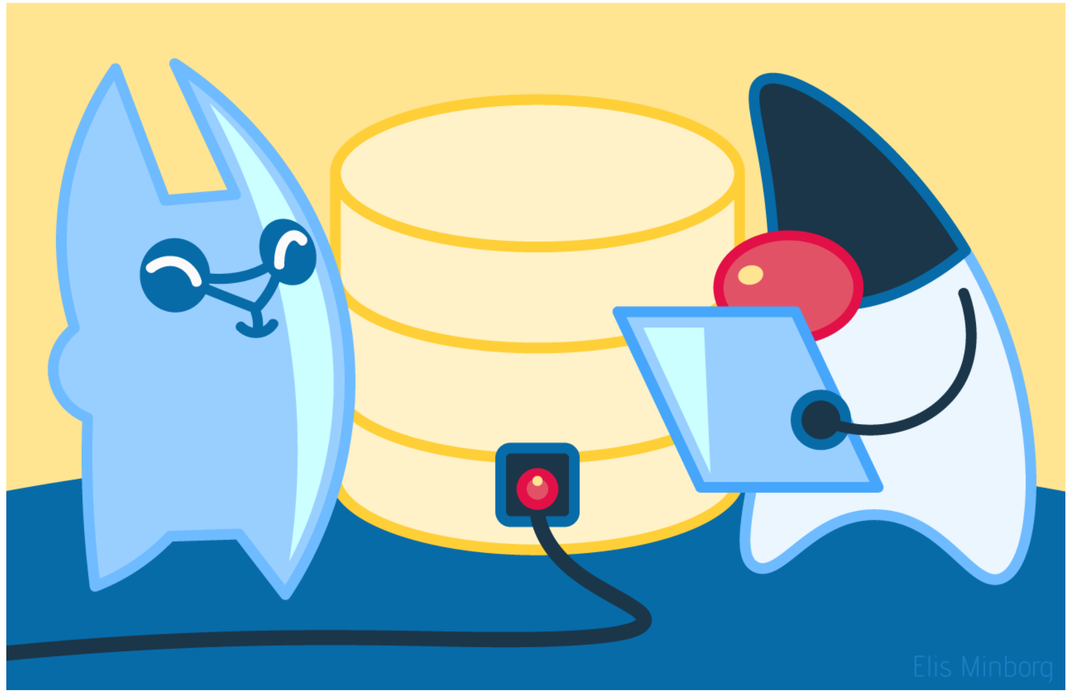
Uses sun.misc.Unsafe

- `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

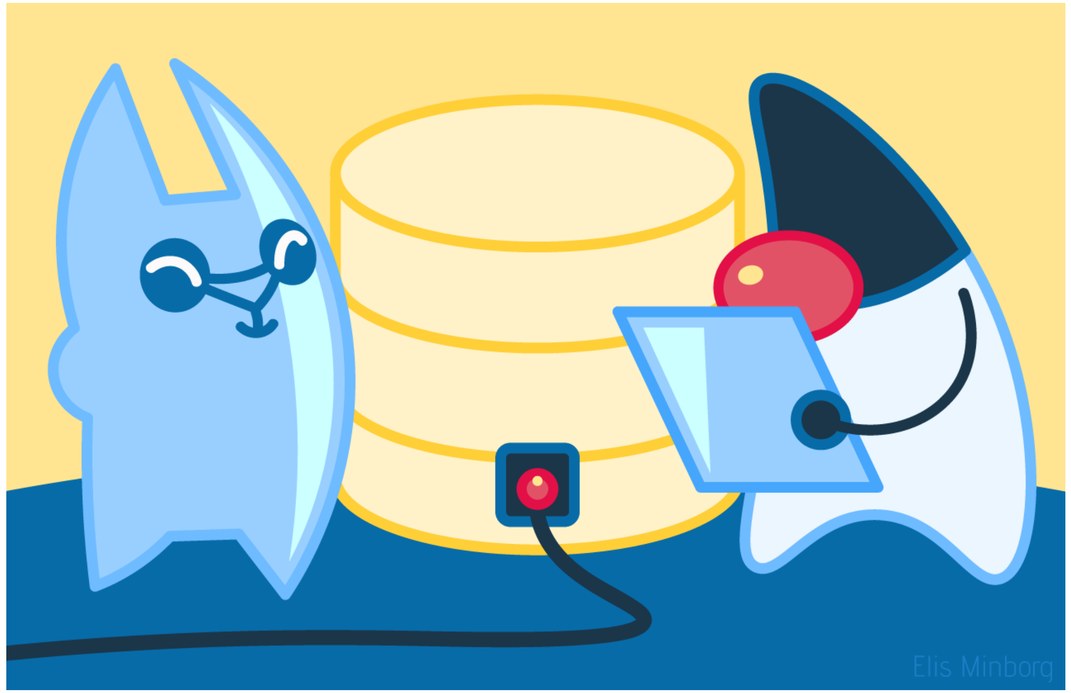
Awaiting Community Progress



Speedment

Awaiting Community Progress

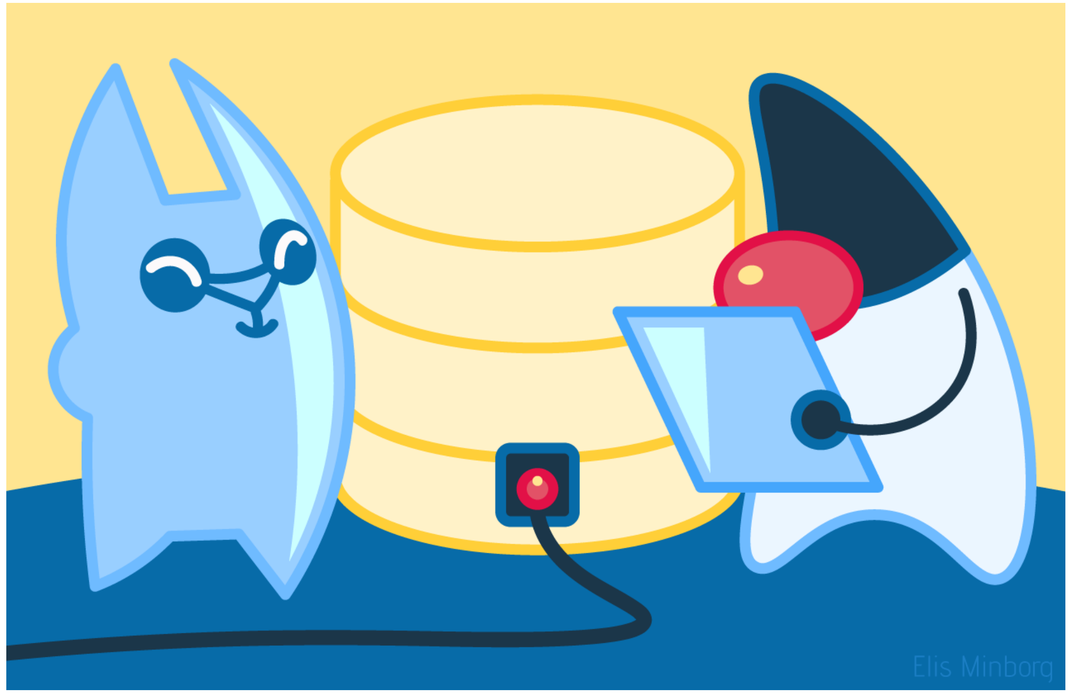
- OSGi bundles



Speedment

Awaiting Community Progress

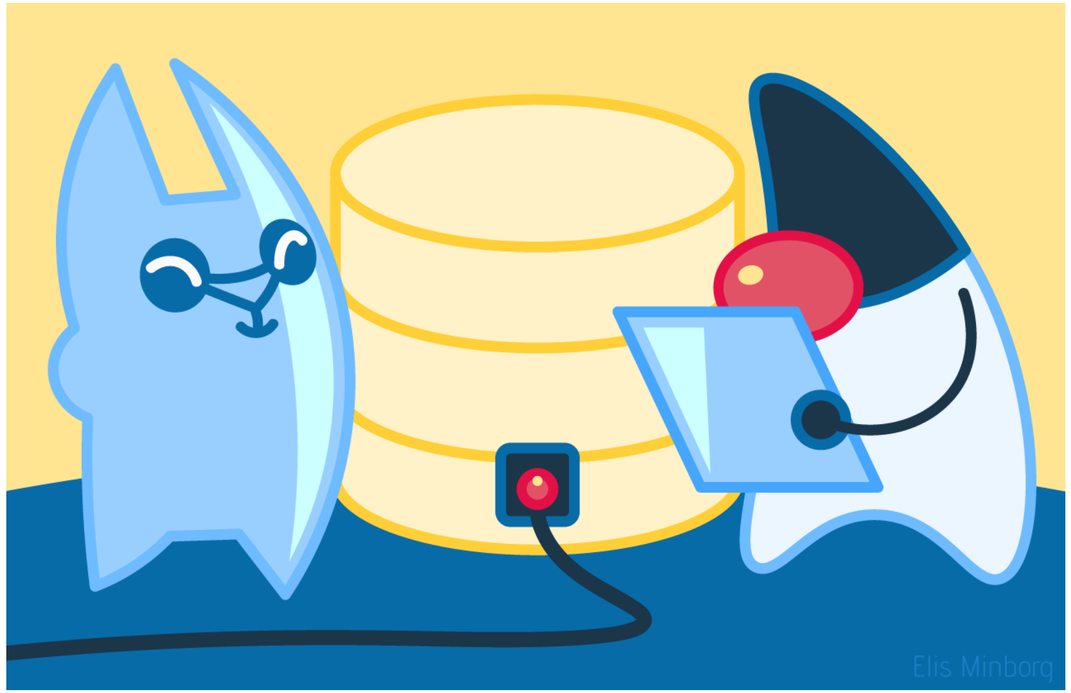
- OSGi bundles
- Third Party Dependencies



Speedment

Awaiting Community Progress

- OSGi bundles
- Third Party Dependencies
  - Split packages

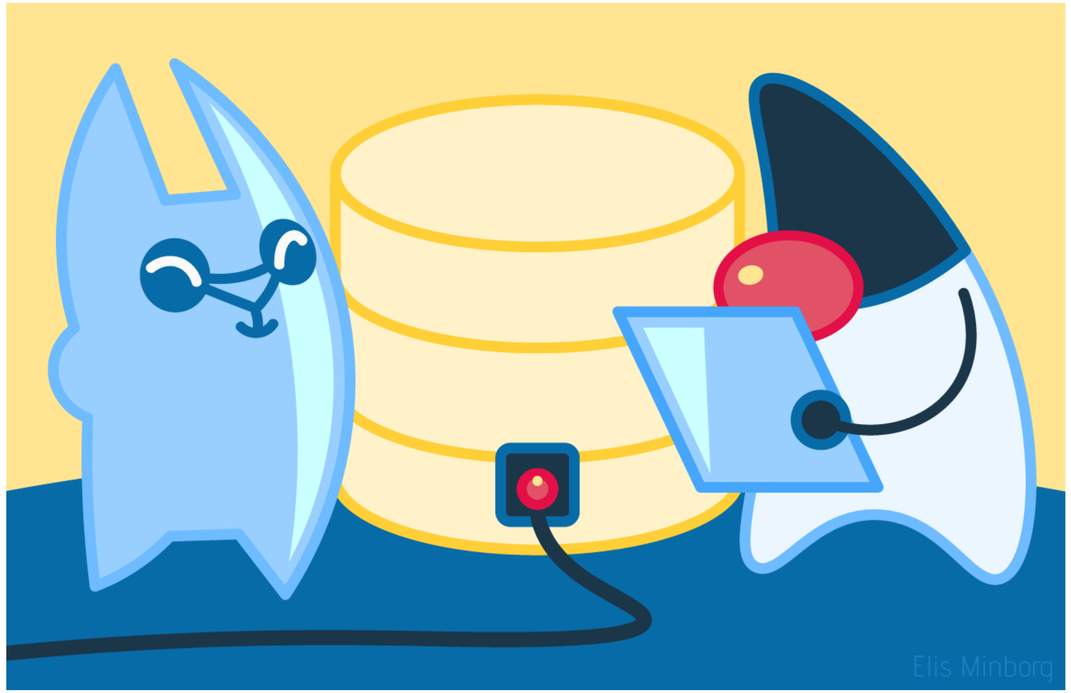


Speedment

Awaiting Community Progress

- OSGi bundles

- Third Party Dependencies
  - Split packages
  - Concealed package conflicts

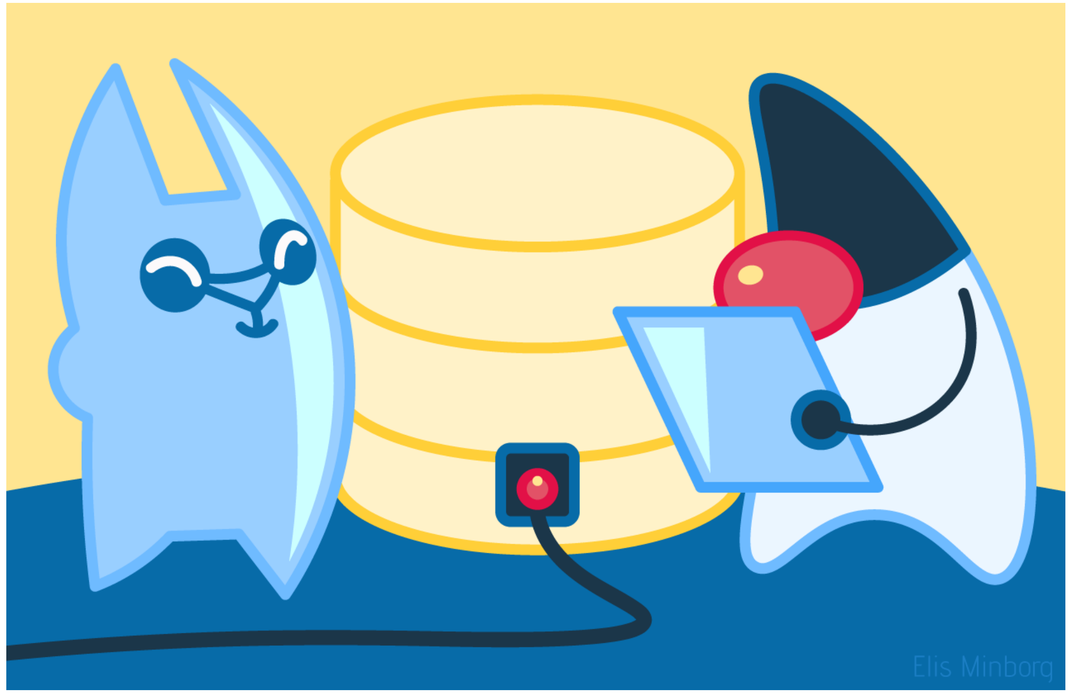


Speedment

Awaiting Community Progress

- OSGi bundles

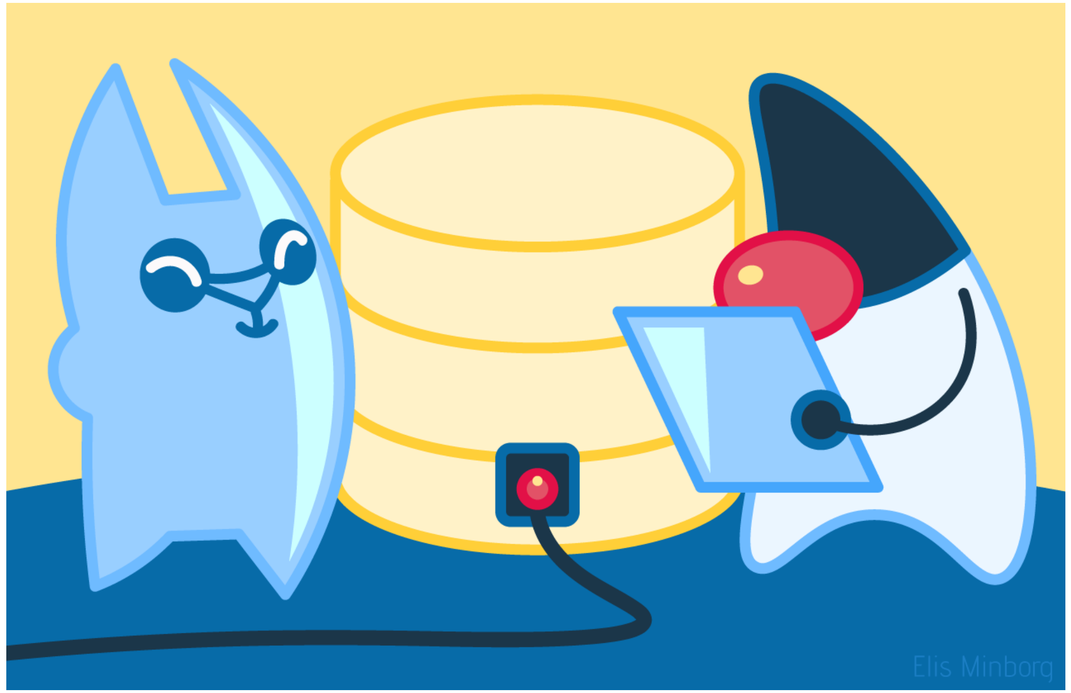
- Third Party Dependencies
  - Split packages
  - Concealed package conflicts
  - Duplicate package names



Speedment

Awaiting Community Progress

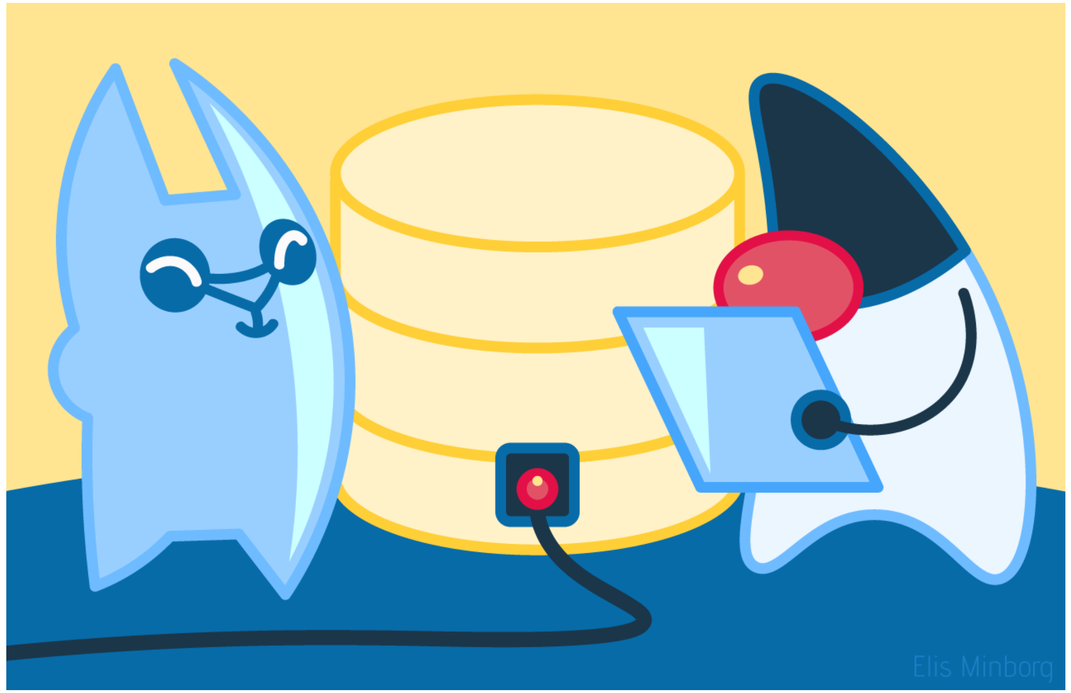
- OSGi bundles
- Third Party Dependencies
  - Split packages
  - Concealed package conflicts
  - Duplicate package names
- Spring Integration



Speedment

Awaiting Community Progress

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



Speedment

Conclusions

Speedment

Conclusions

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

Speedment

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:

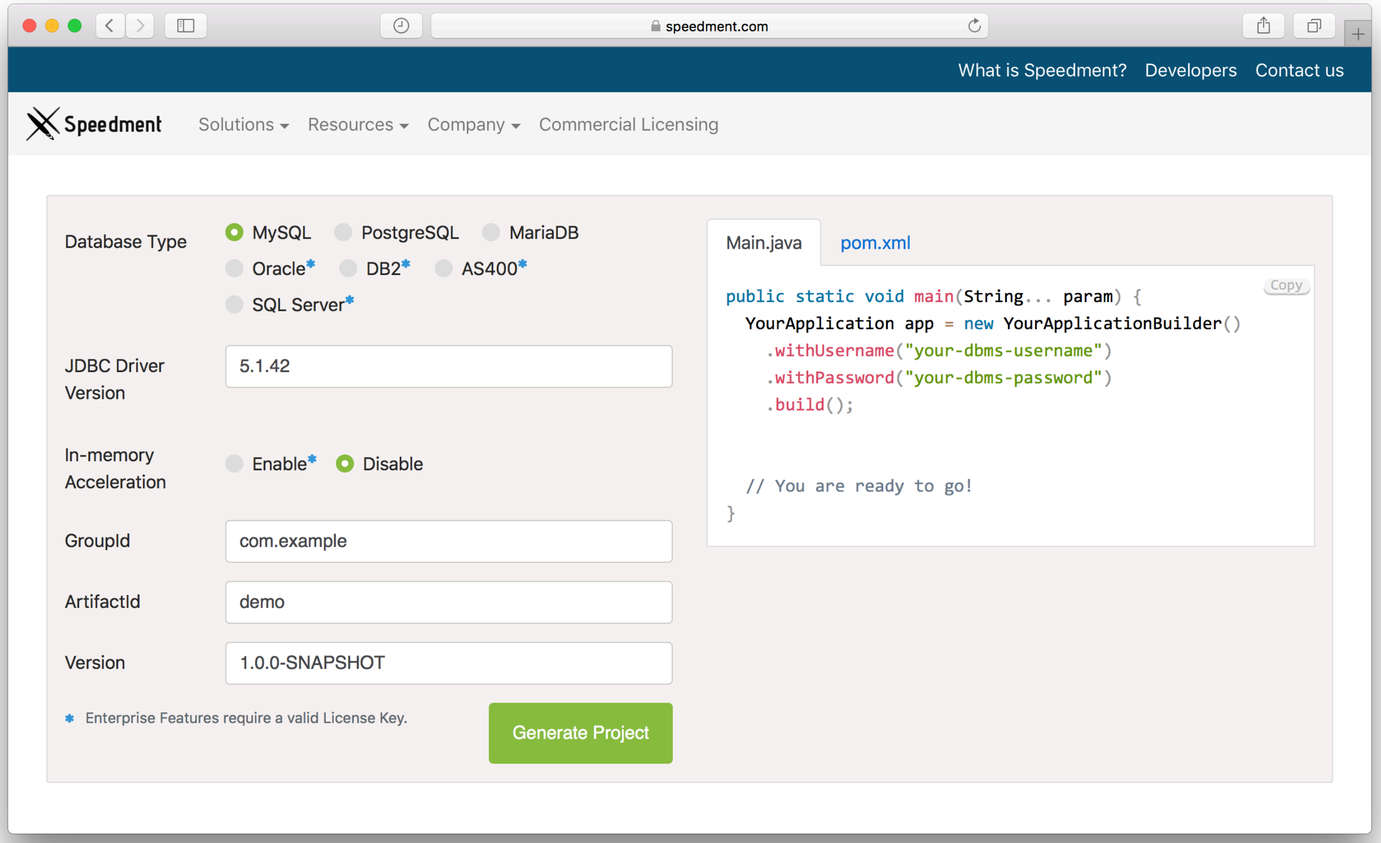
- refine application modularity,
- rework references to non-open parts of the JDK and
- revisit design decisions regarding Unsafe.

Postpone Decisions (S.E.P.):

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

Try Speedment

github.com/speedment/speedment
www.speedment.com/initializer/



Speedment

Q&A

@speedment

@dan_lawesson

Speedment

Extra Slides for Q&A

Speedment

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
    )
  )
);
```

Speedment

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
    )
  )
);
```

Standard Java 8

Speedment

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
```

Speedment

Update

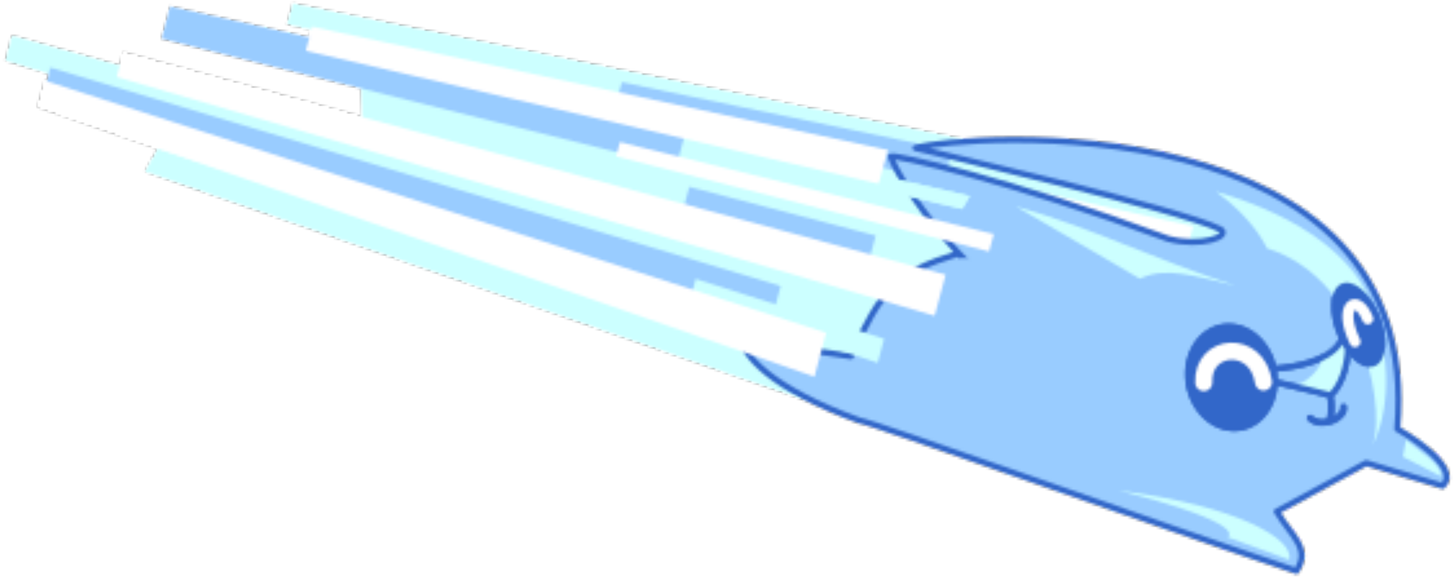
```
hares.stream()
   .filter(Hare.ID.equal(42)) // All Hares with ID = 42 (just one)
   .map(Hare.AGE.setTo(10))   // Applies a setter
   .forEach(hares.updater()); // Applies the updater function
```

Database Connectors

Speedment

Beyond Open-Source

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



Speedment

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

Speedment

In-JVM-Memory, JMH Benchmarks(*)

Benchmark	Mode	Cnt	Score	Error	Units
SpeedmentBenchmark.findStartsWith	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.findStartsWithSql	avgt	200	≈ 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	≈ 10^{-6}		s/op
SpeedmentBenchmark.sortSql	avgt	200	≈ 10^{-4}		s/op
SpeedmentBenchmark.scrollSorted	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.scrollSortedSql	avgt	200	24.661 ±	0.670	s/op
SpeedmentBenchmark.count	avgt	180	≈ 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.

Speedment

In-JVM-Memory, JMH Benchmarks(*)

Benchmark	Mode	Cnt	Score	Error	Units
SpeedmentBenchmark.findStartsWith	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.findStartsWithSql	avgt	200	≈ 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	≈ 10^{-6}		s/op
SpeedmentBenchmark.sortSql	avgt	200	≈ 10^{-4}		s/op
SpeedmentBenchmark.scrollSorted	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.scrollSortedSql	avgt	200	24.661 ±	0.670	s/op
SpeedmentBenchmark.count	avgt	180	≈ 10^{-8}		s/op
SpeedmentBenchmark.countSql	avgt	200	5.143 ±	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.

Speedment

In-JVM-Memory, JMH Benchmarks(*)

Benchmark	Mode	Cnt	Score	Error	Units
SpeedmentBenchmark.findStartsWith	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.findStartsWithSql	avgt	200	≈ 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	≈ 10^{-6}		s/op
SpeedmentBenchmark.sortSql	avgt	200	≈ 10^{-4}		s/op
SpeedmentBenchmark.scrollSorted	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.scrollSortedSql	avgt	200	24.661 ±	0.670	s/op
SpeedmentBenchmark.count	avgt	180	≈ 10^{-8}		s/op
SpeedmentBenchmark.countSql	avgt	200	5.143 ±	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.

Speedment

In-JVM-Memory, JMH Benchmarks(*)

Benchmark	Mode	Cnt	Score	Error	Units
SpeedmentBenchmark.findStartsWith	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.findStartsWithSql	avgt	200	≈ 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	≈ 10^{-6}		s/op
SpeedmentBenchmark.sortSql	avgt	200	≈ 10^{-4}		s/op
SpeedmentBenchmark.scrollSorted	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.scrollSortedSql	avgt	200	24.661 ±	0.670	s/op
SpeedmentBenchmark.count	avgt	180	≈ 10^{-8}		s/op
SpeedmentBenchmark.countSql	avgt	200	5.143 ±	0.012	s/op

10x

>10x

100x

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

Speedment

In-JVM-Memory, JMH Benchmarks(*)

Benchmark	Mode	Cnt	Score	Error	Units
SpeedmentBenchmark.findStartsWith	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.findStartsWithSql	avgt	200	≈ 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	≈ 10^{-6}		s/op
SpeedmentBenchmark.sortSql	avgt	200	≈ 10^{-4}		s/op
SpeedmentBenchmark.scrollSorted	avgt	200	≈ 10^{-5}		s/op
SpeedmentBenchmark.scrollSortedSql	avgt	200	24.661 ±	0.670	s/op
SpeedmentBenchmark.count	avgt	180	≈ 10^{-8}		s/op
SpeedmentBenchmark.countSql	avgt	200	5.143 ±	0.012	s/op

10x

>10x

100x

2,000,000x

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

Speedment

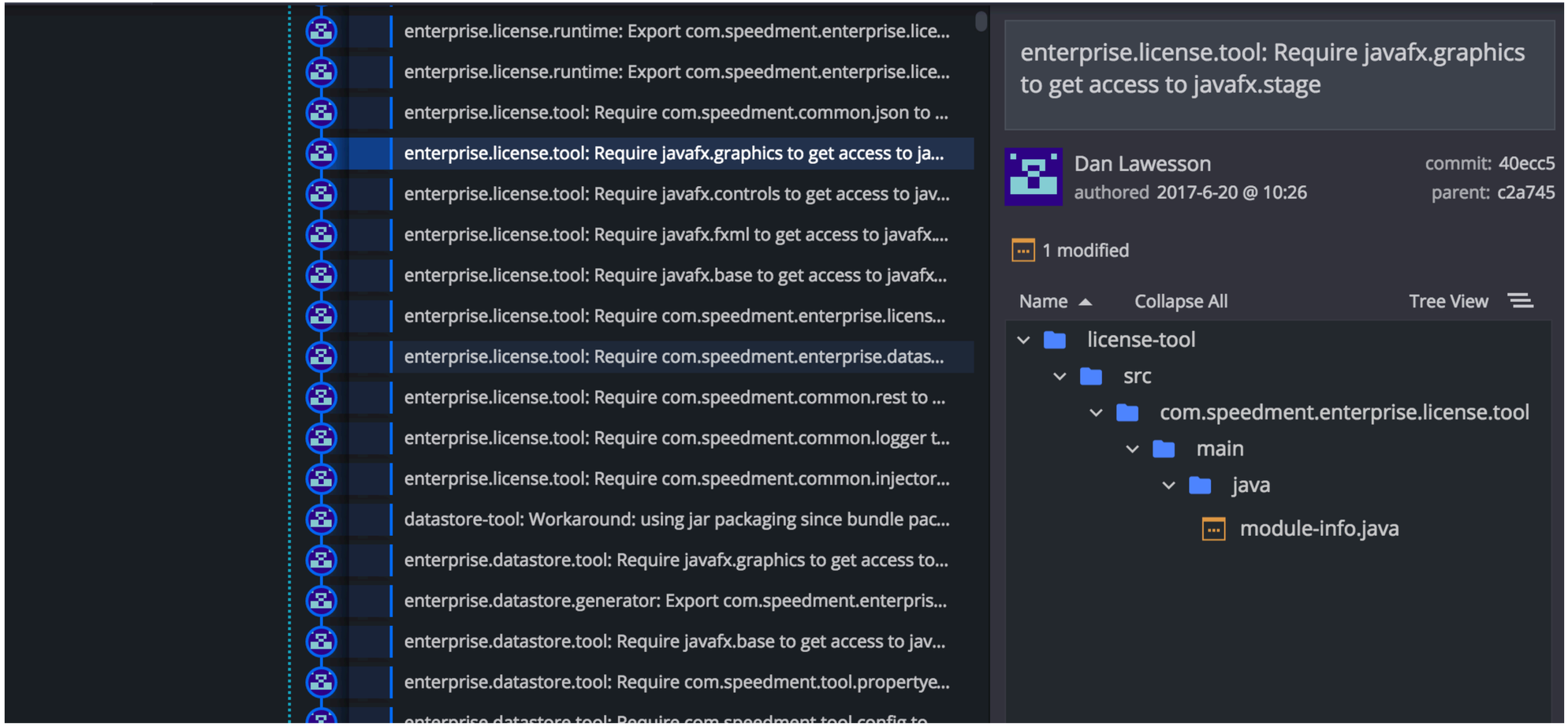
In-JVM-Memory, JMH Benchmarks(*)

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

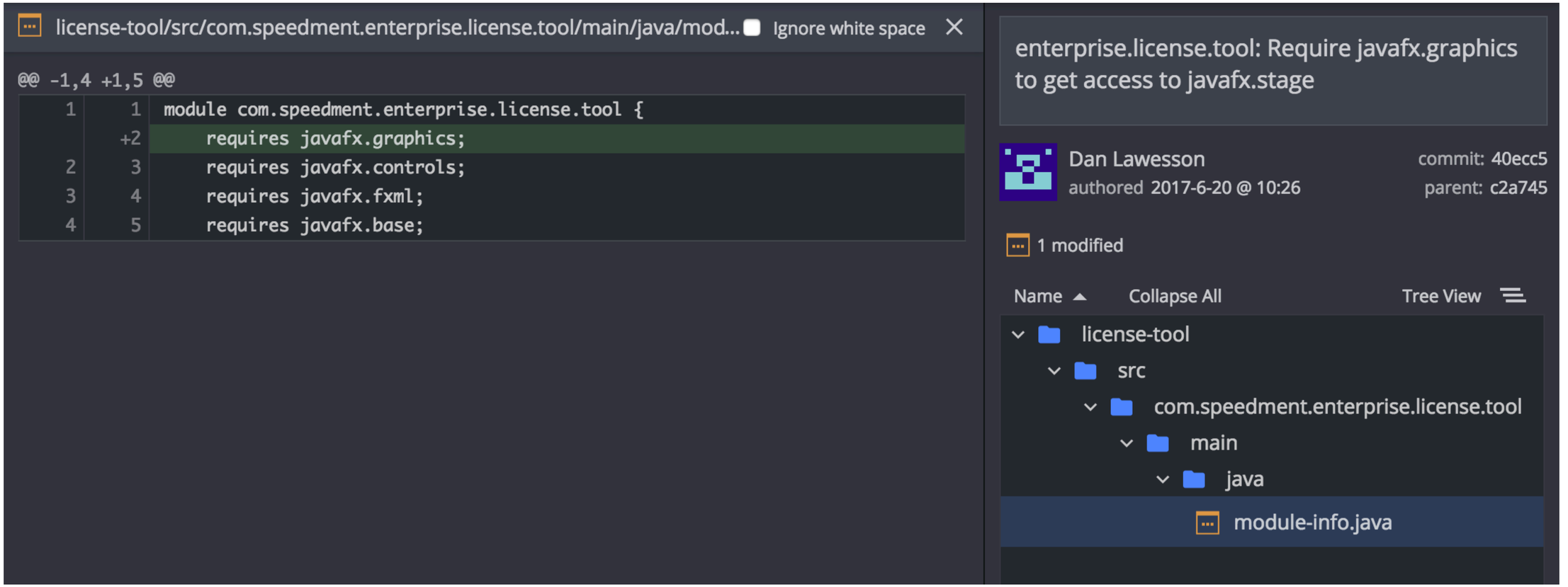
Speedment

Gitkraken view of Jigsawing Script Commits



Speedment

Gitkraken view of Jigsawing Script Commits



Speedment