

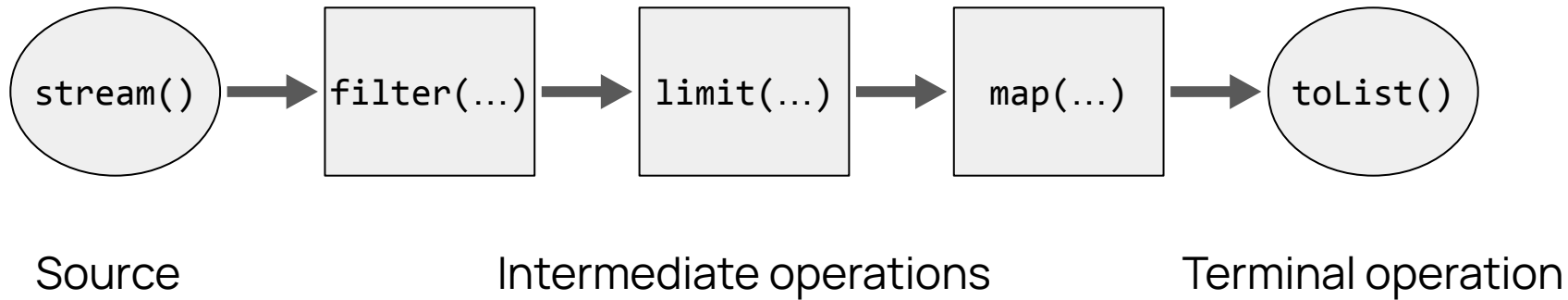


Stream Gatherers

Write Your Own
Stream Operations!

Sven Woltmann

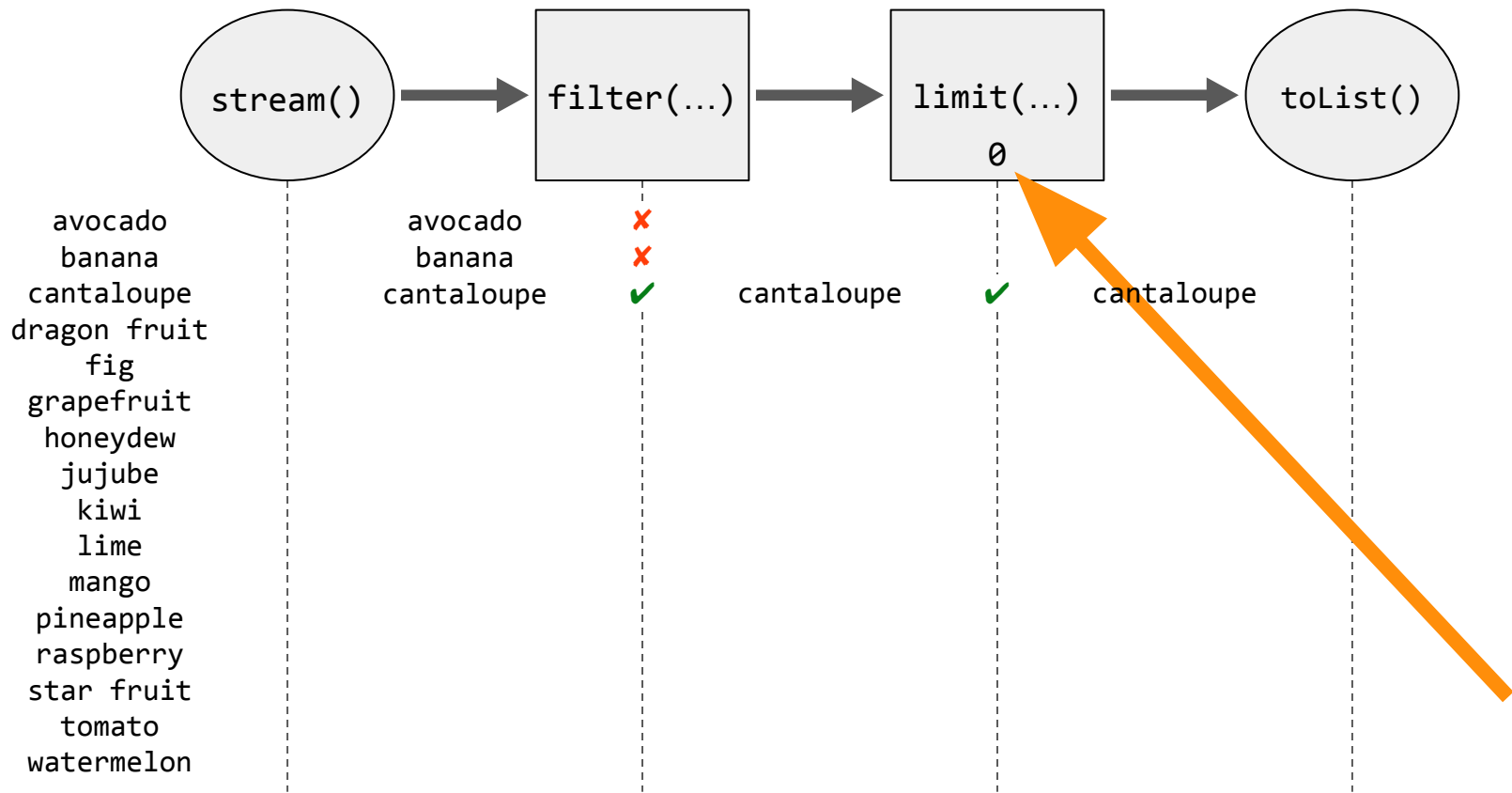
Stream Pipeline

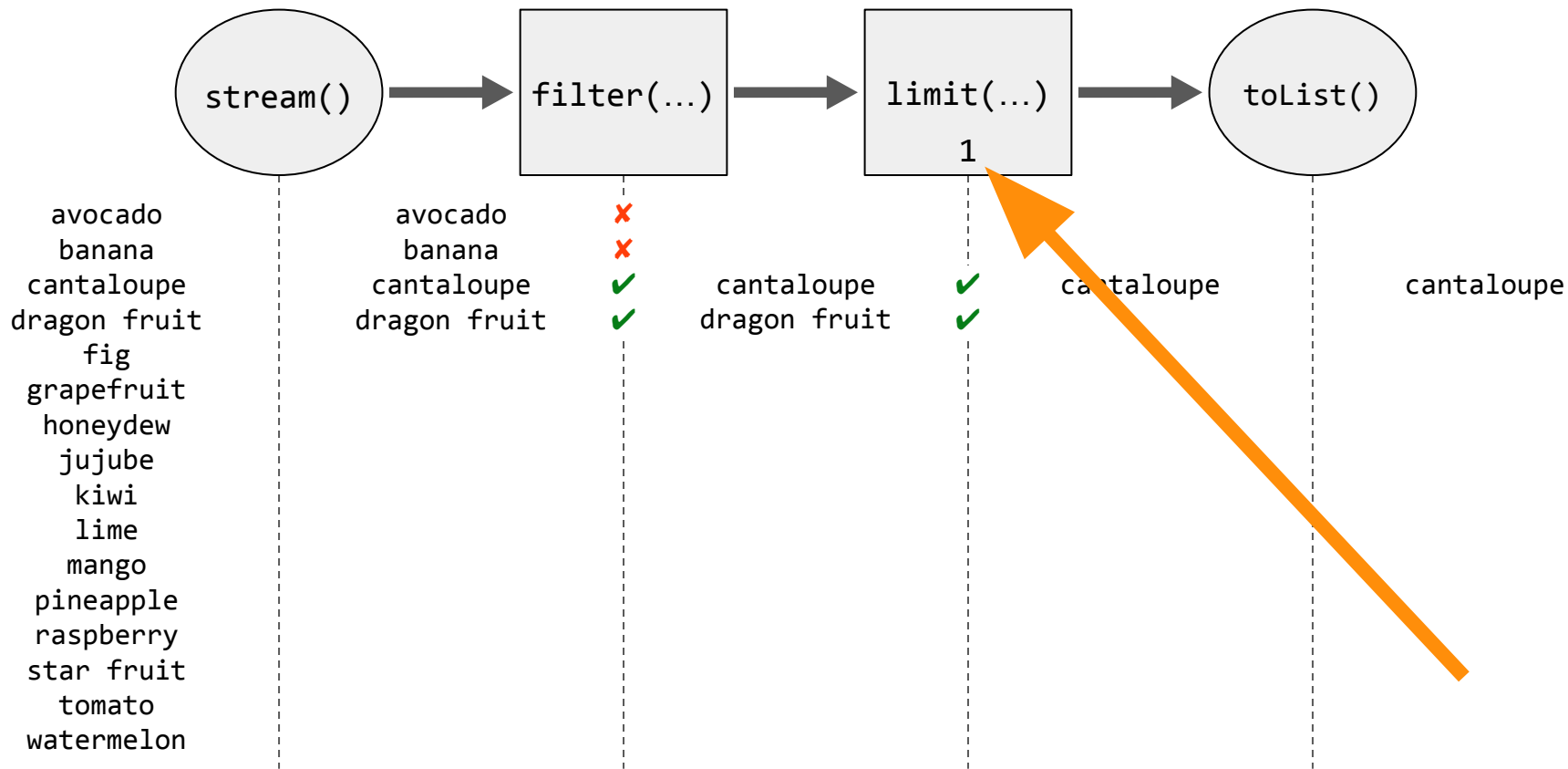


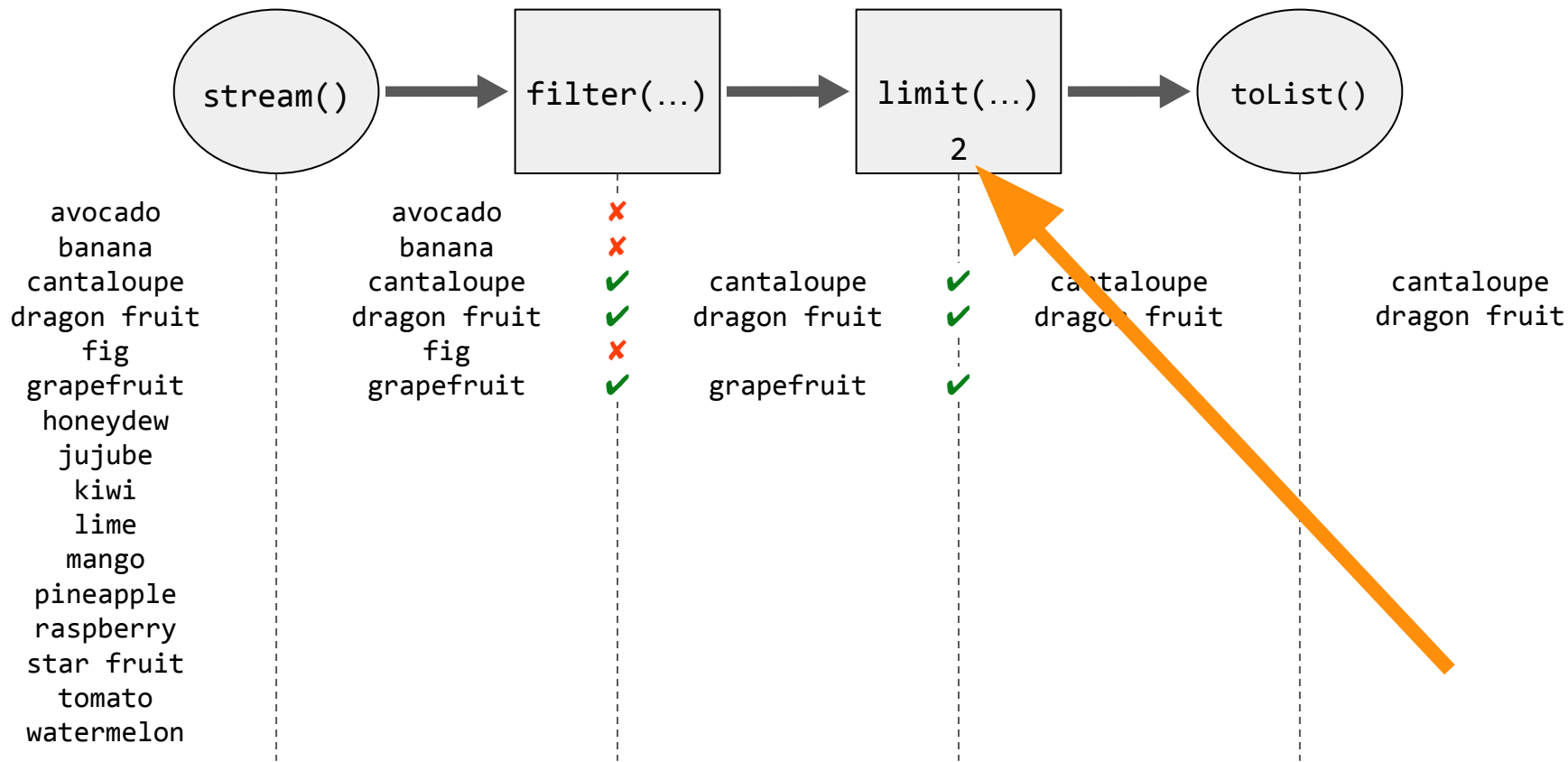
```
List<String> fruits =  
    List.of("avocado", "banana", "cantaloupe",  
            "dragon fruit", "fig", "grapefruit", "honeydew",  
            "jujube", "kiwi", "lime", "mango", "pineapple",  
            "raspberry", "star fruit", "tomato", "watermelon");
```

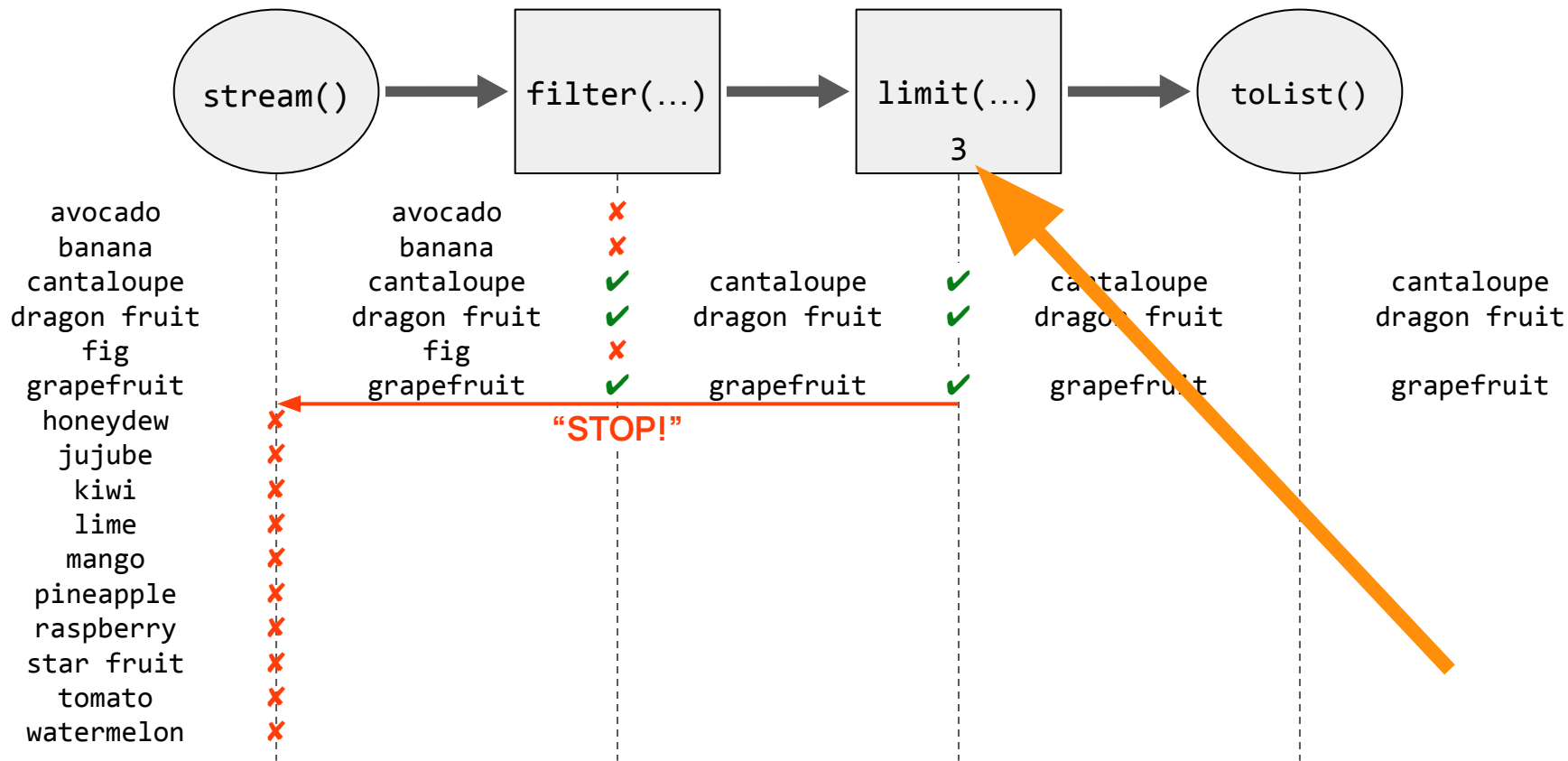
```
List<String> result =  
    fruits.stream()  
        .filter(fruit -> fruit.length() >= 8)  
        .limit(3)  
        .toList();
```



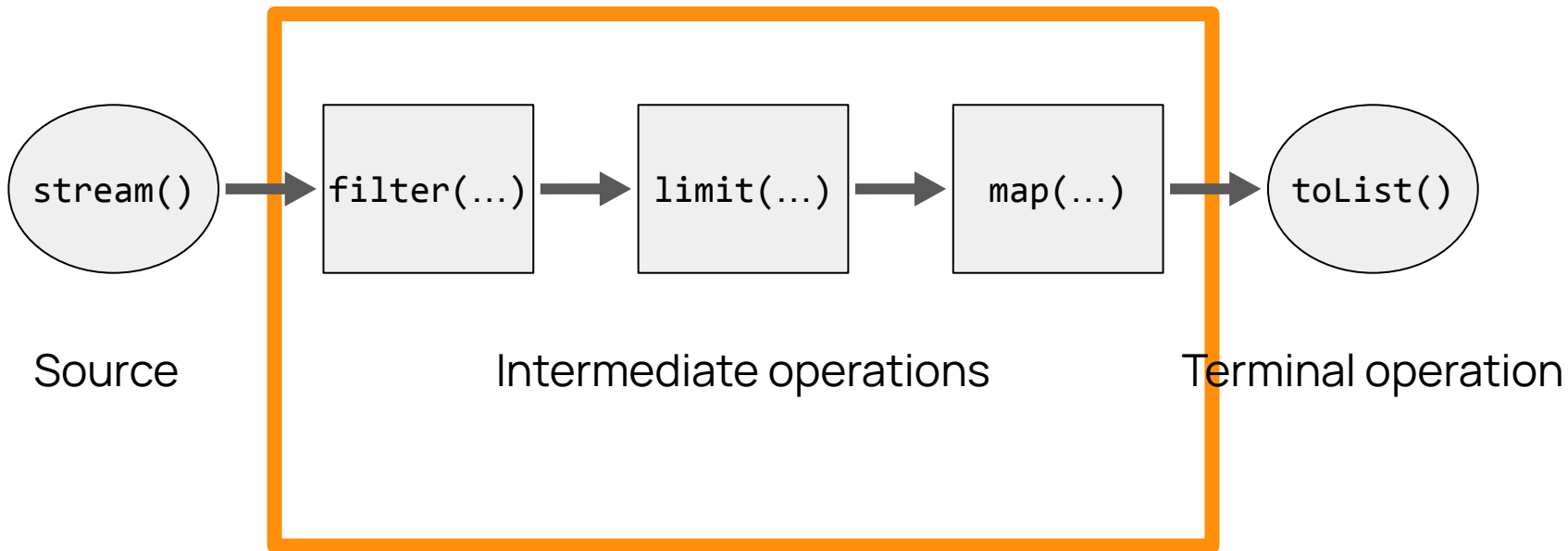








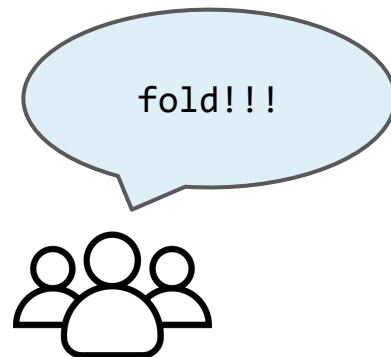
Stream Pipeline



Intermediate Operations in the JDK



`filter(...)`
`limit(...)`
`map(...)`
`flatMap(...)`
`mapMulti(...)`
`distinct()`
`sorted()`
`skip(...)`
`takeWhile(...)`
`dropWhile(...)`
`peek(...)`



Stream Gatherers

JEP 461: Stream Gatherers (Preview)

<i>Owner</i>	Viktor Klang
<i>Type</i>	Feature
<i>Scope</i>	SE
<i>Status</i>	Closed / Delivered
<i>Release</i>	22
<i>Component</i>	core-libs/java.util.stream
<i>Discussion</i>	core dash libs dash dev at openjdk dot org
<i>Relates to</i>	JEP 473: Stream Gatherers (Second Preview)
<i>Reviewed by</i>	Alan Bateman, Alex Buckley, Paul Sandoz
<i>Endorsed by</i>	Paul Sandoz
<i>Created</i>	2023/10/11 13:08
<i>Updated</i>	2024/04/17 14:09
<i>Issue</i>	8317955

Summary

Enhance the [Stream API](#) to support custom intermediate operations. This will allow stream pipelines to transform data in ways that are not easily achievable with the existing built-in intermediate operations. This is a [preview API](#).

Goals

- Make stream pipelines more flexible and expressive.
- Insofar as possible, allow custom intermediate operations to manipulate streams of infinite size.

Non-Goals

- It is not a goal to change the Java programming language to better facilitate stream processing.
- It is not a goal to special-case the compilation of code that uses the Stream API.

Motivation

Java 8 introduced the first API designed specifically for lambda expressions: the Stream API, `java.util.stream`. A stream is a lazily computed, potentially unbounded sequence of values. The API supports the ability to process a stream either sequentially or in parallel.

A stream pipeline consists of three parts: a source of elements, any number of intermediate operations, and a terminal operation. For example:

JEP 473: Stream Gatherers (Second Preview)

<i>Owner</i>	Viktor Klang
<i>Type</i>	Feature
<i>Scope</i>	SE
<i>Status</i>	Closed / Delivered
<i>Release</i>	23
<i>Component</i>	core-libs/java.util.stream
<i>Discussion</i>	core dash libs dash dev at openjdk dot org
<i>Effort</i>	XS
<i>Duration</i>	XS
<i>Relates to</i>	JEP 461: Stream Gatherers (Preview) JEP 485: Stream Gatherers
<i>Reviewed by</i>	Alan Bateman, Paul Sandoz
<i>Endorsed by</i>	Paul Sandoz
<i>Created</i>	2024/03/11 19:39
<i>Updated</i>	2024/09/02 16:05
<i>Issue</i>	8327844

Summary

Enhance the [Stream API](#) to support custom intermediate operations. This will allow stream pipelines to transform data in ways that are not easily achievable with the existing built-in intermediate operations. This is a [preview API](#).

History

We proposed Stream Gatherers as a preview feature in JEP 461 and delivered it in JDK 22. We here propose to re-preview the API in JDK 23, without change, in order to gain additional experience and feedback.

Goals

- Make stream pipelines more flexible and expressive.
- Insofar as possible, allow custom intermediate operations to manipulate streams of infinite size.

Non-Goals

- It is not a goal to change the Java programming language to better facilitate stream processing.
- It is not a goal to special-case the compilation of code that uses the Stream API.

JEP 485: Stream Gatherers

<i>Owner</i>	Viktor Klang
<i>Type</i>	Feature
<i>Scope</i>	SE
<i>Status</i>	Candidate
<i>Component</i>	core-libs/java.util.stream
<i>Discussion</i>	core dash libs dash dev at openjdk dot org
<i>Effort</i>	XS
<i>Duration</i>	XS
<i>Relates to</i>	JEP 473: Stream Gatherers (Second Preview)
<i>Reviewed by</i>	Alan Bateman, Paul Sandoz
<i>Created</i>	2024/07/08 15:42
<i>Updated</i>	2024/10/07 21:43
<i>Issue</i>	8335899

Summary

Enhance the [Stream API](#) to support custom intermediate operations. This will allow stream pipelines to transform data in ways that are not easily achievable with the existing built-in intermediate operations.

History

Stream Gatherers were proposed as a preview feature by JEP 461 in JDK 22 and re-
viewed by JEP 473 in JDK 23. We here propose to finalize the API in JDK 24,
without change.

Goals

- Make stream pipelines more flexible and expressive.
- Insofar as possible, allow custom intermediate operations to manipulate streams of infinite size.

Non-Goals

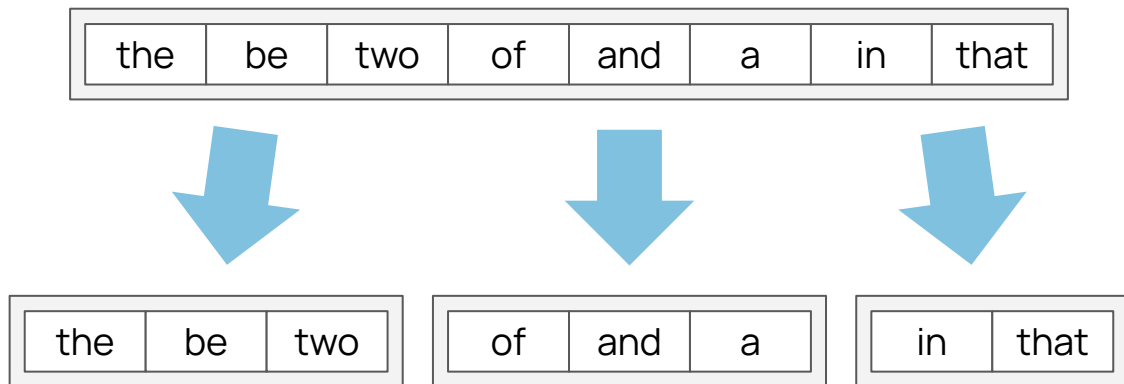
- It is not a goal to change the Java programming language to better facilitate stream processing.
- It is not a goal to special-case the compilation of code that uses the Stream API.

Motivation

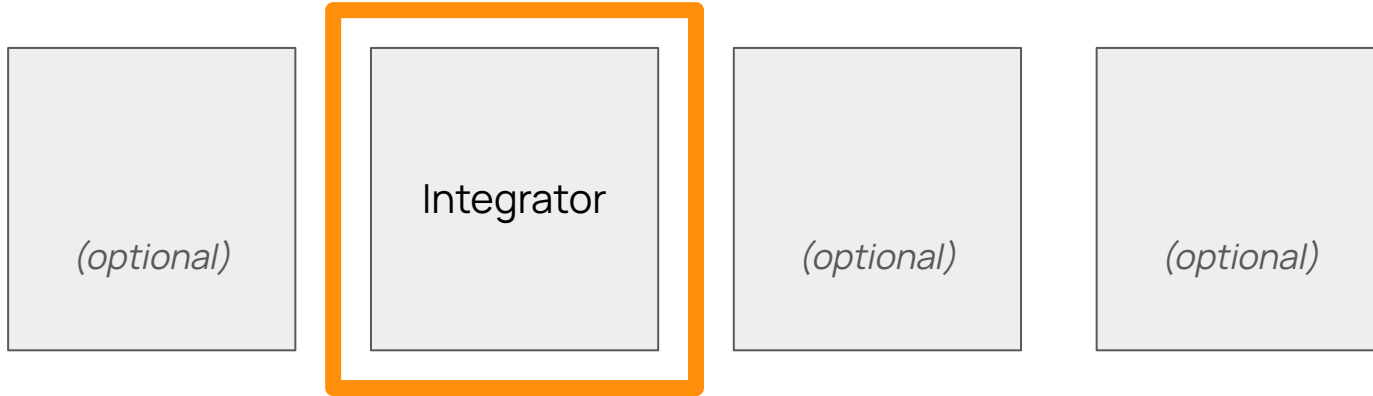
Java 8 introduced the first API designed specifically for lambda expressions: the

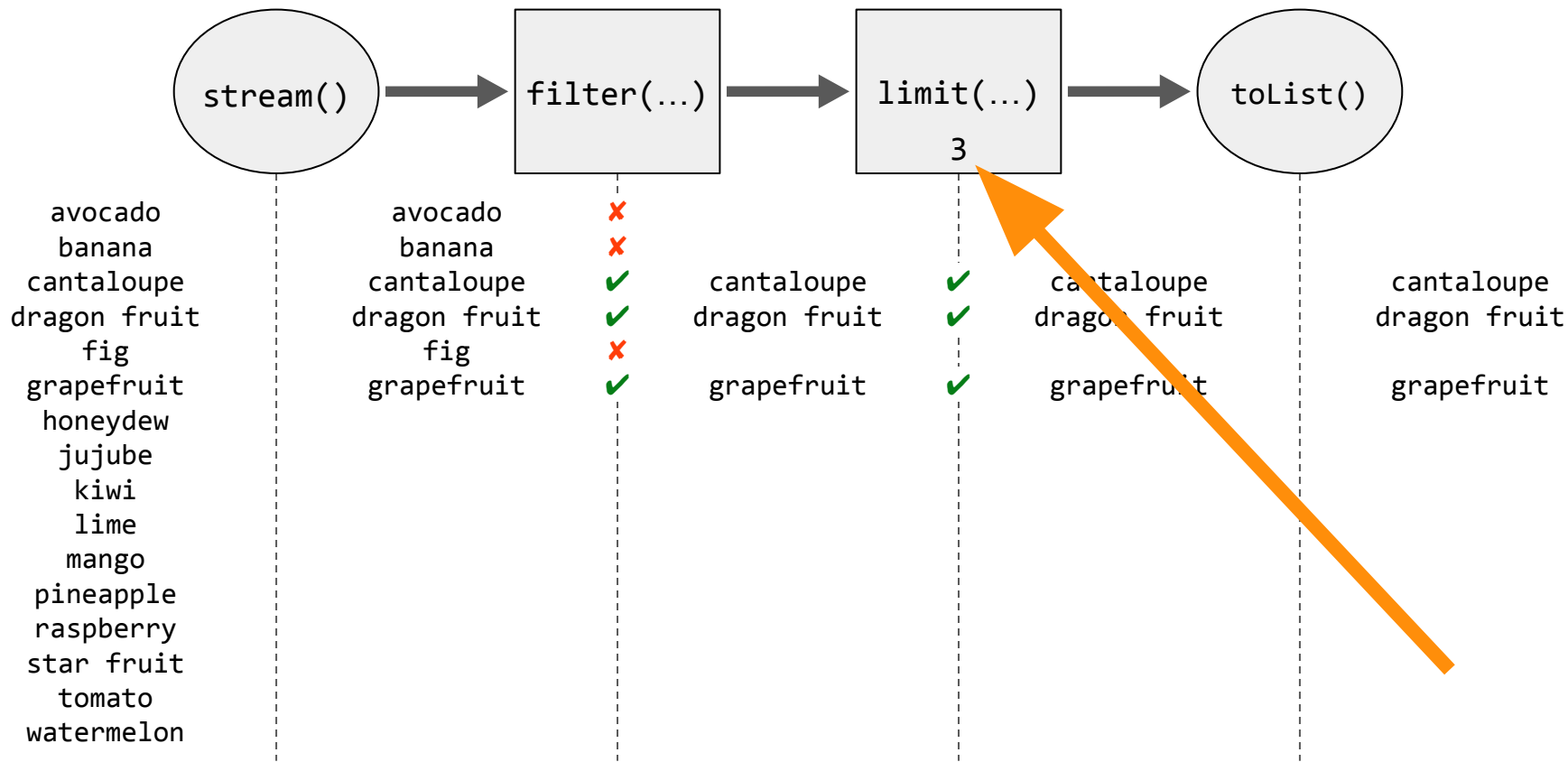


Demo

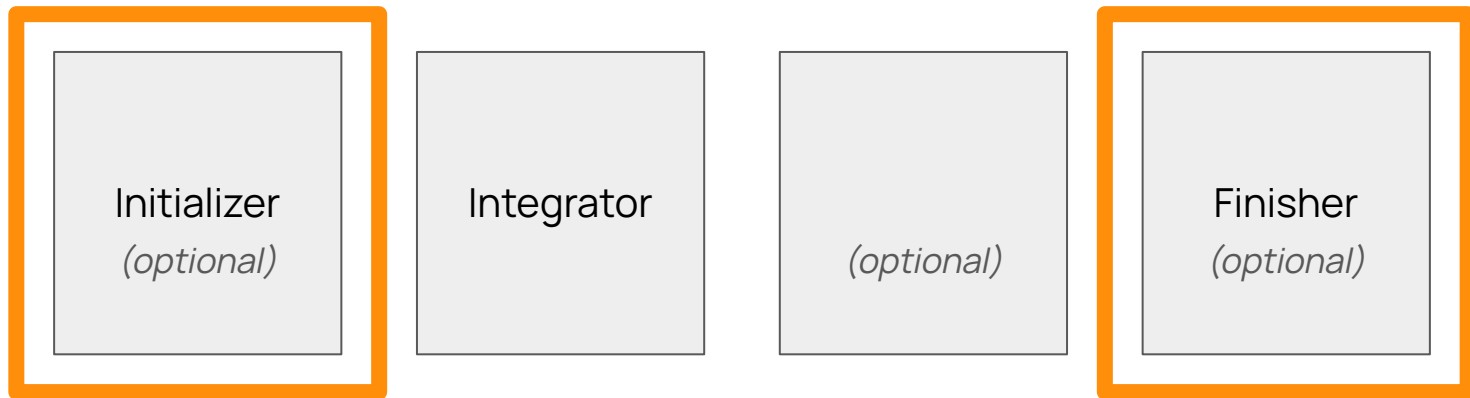


Stream Gatherer Components

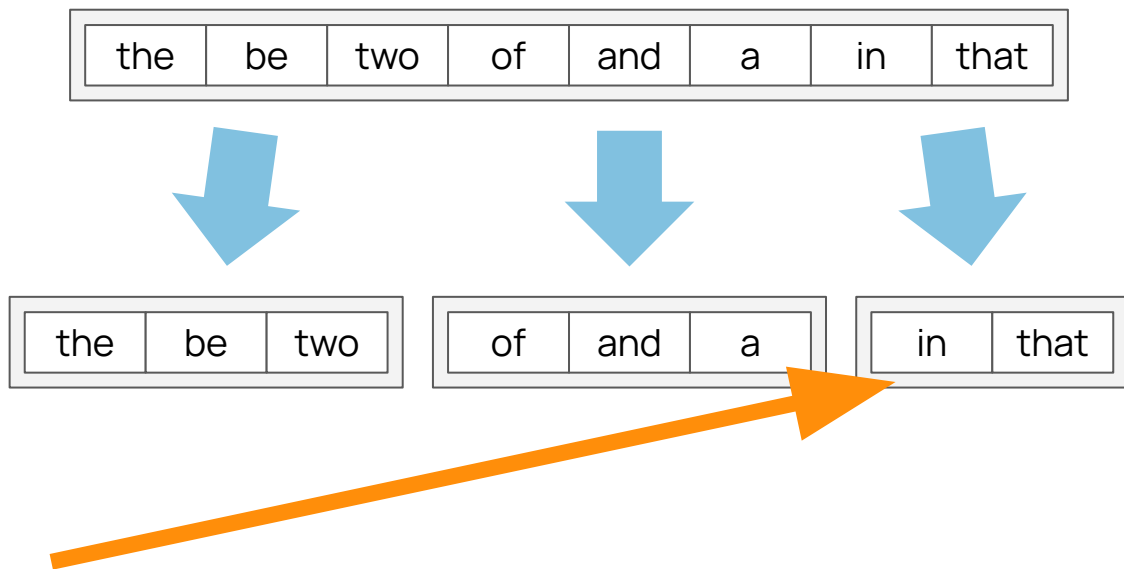




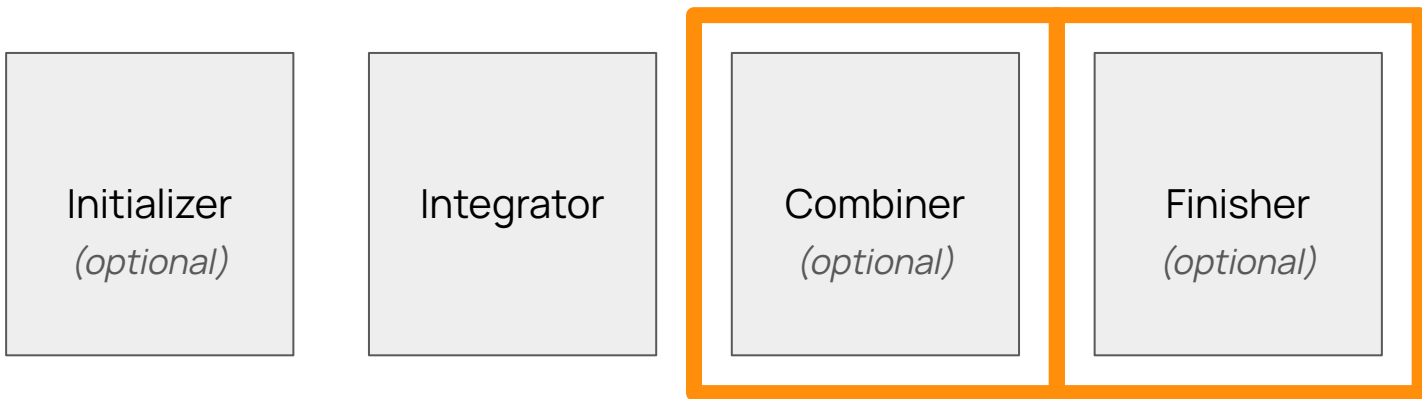
Stream Gatherer Components



Fixed Window



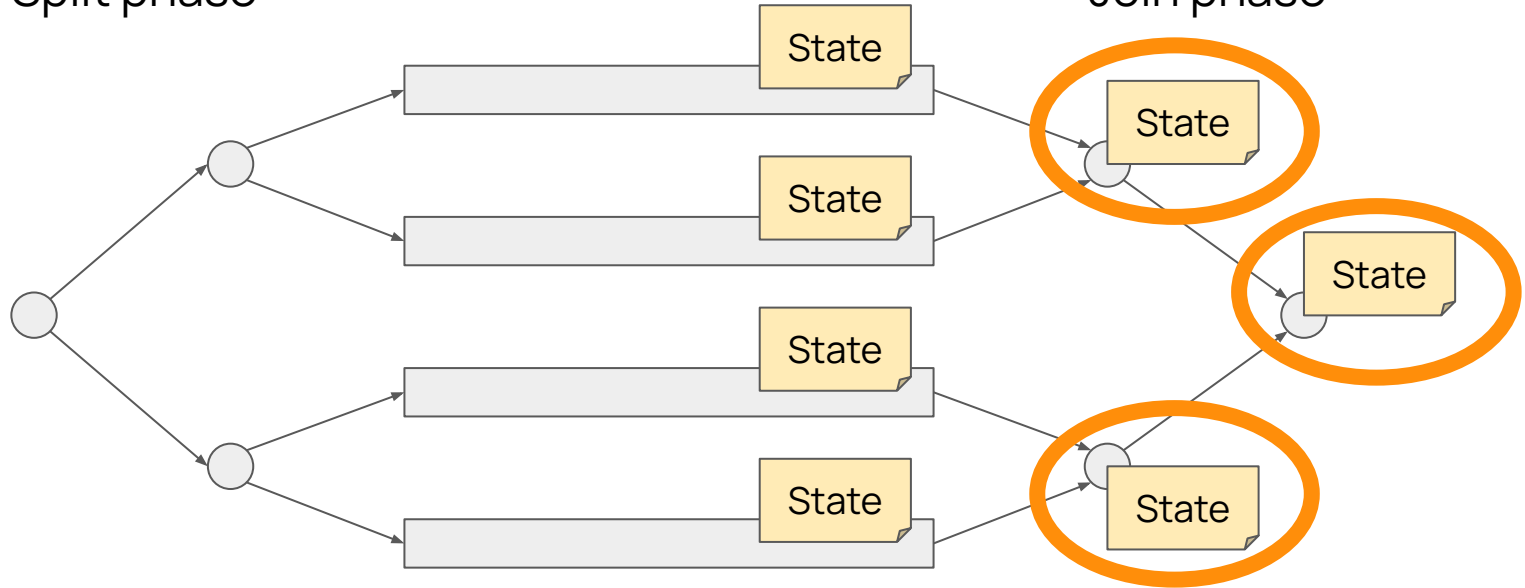
Stream Gatherer Components



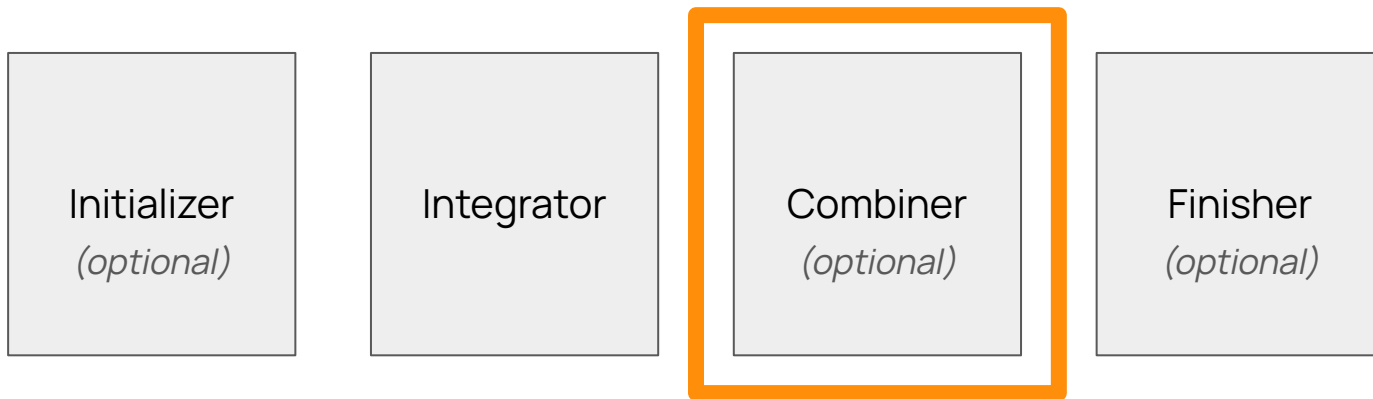
Parallel Stream

Split phase

Join phase



Stream Gatherer Components



Predefined Stream Gatherers

```
windowFixed(int windowSize)
```

```
windowSliding(int windowSize)
```

```
fold(Supplier<R> initial,  
      BiFunction<? super R, ? super T, ? extends R> folder)
```

```
scan(Supplier<R> initial,  
      BiFunction<? super R, ? super T, ? extends R> scanner)
```

```
mapConcurrent(int maxConcurrency,  
              Function<? super T, ? extends R> mapper)
```



3rd Party Stream Gatherers

more-gatherersPublic

SponsorUnwatch2Fork1Starred27

main6 Branches0 Tags

Go to file

Add fileCode

dependabot[bot]

Bump org.pitest:pitest-maven from 1.17.3 to 1.17.4 (#79)

9890b86 · yesterday

77 Commits

github

Add GitHub Actions job for JavaDoc verification (#52)

3 months ago

src

Rename byIndex() to filteringByIndex() (#71)

last month

gitignore

Add initial project skeleton (#1)

3 months ago

CODE_OF_CONDUCT.md

Add initial project skeleton (#1)

3 months ago

CONTRIBUTING.md

Add initial project skeleton (#1)

3 months ago

LICENSE.md

Add initial project skeleton (#1)

3 months ago

README.md

Rename byIndex() to filteringByIndex() (#71)

last month

pom.xml

Bump org.pitest:pitest-maven from 1.17.3 to 1.17.4 (#79)

yesterday

README

Code of conduct

Apache-2.0 license

more-gatherers

Missing Stream API functionality you always longed for - provided via `gatherers`

build

passing

pitest

passing

maven:central

not found



Releases

No releases published

Sponsor this project

pivovarit

Grzegorz Piwowarek

Sponsor

Learn more about GitHub Sponsors

Contributors 2

pivovarit

Grzegorz Piwowarek

dependabot[bot]

Deployments 65

github-pages

yesterday

< 64 deployments

gatherers4jPublic

Unwatch4Fork2Starred36

main2 Branches8 Tags

Go to file

Add fileCode

tginsberg

README fixes (#73)

af8b3a3 · last week

40 Commits

github/workflows

Release v0.6.0 (#54)

2 months ago

gradle/wrapper

Release v0.6.0 (#54)

2 months ago

src

Release v0.7.0 (#72)

last week

gitignore

Initial commit

8 months ago

CHANGELOG.md

Release v0.7.0 (#72)

last week

LICENSE

Initial commit

8 months ago

README.md

README fixes (#73)

last week

VERSION.txt

Release v0.7.0 (#72)

last week

build.gradle.kts

Release v0.7.0 (#72)

last week

gradlew

Upgrade dependencies (#21)

5 months ago

gradlew.bat

Upgrade dependencies (#21)

5 months ago

settings.gradle.kts

Initial commit

8 months ago

README

Apache-2.0 license

Gatherers4j

A library of useful [Stream Gatherers](#) (custom intermediate operations) for Java 23+.

Installing

To use this library, add it as a dependency to your build. This library has one transitive dependency - the [JSpecify](#) set of annotations for static analysis tools.

Maven

About

A library of useful Stream Gatherers (custom intermediate operations) for Java.

java

gatherer

java-23

gatherers

java23

README

Apache-2.0 license

Activity

36 stars

4 watching

2 forks

Report repository

Releases 8

Release v0.7.0

(Latest)

last week

+ 7 releases

Languages

Java

100.0%



Performance + Limits

- No gatherers for the primitive streams
`IntStream`, `LongStream`, and `DoubleStream`



```
int[] array = IntStream.iterate(1, i -> i + 1)
    .filter(i -> i % 7 == 0)
    .limit(3)
    .toArray();
```



Performance + Limits

- No gatherers for the primitive streams
IntStream, LongStream, and DoubleStream
- No access to the characteristics of the stream,
no indication of which characteristics are retained.



```
List<String> words = . . .
```

Faster!



```
long count = words.stream()  
    .map(String::toUpperCase)  
    .count();
```

3,7 μ s



```
long count = words.stream()  
    .gather(mapping(String::toUpperCase))  
    .count();
```

861.529,6 μ s



Performance + Limits

- No gatherers for the primitive streams
`IntStream`, `LongStream`, and `DoubleStream`
- No access to the characteristics of the stream,
no indication of which characteristics are retained.



Links,
Slides



LinkedIn



sven@happycoders.eu



<https://www.happycoders.eu/>



<https://linkedin.com/in/sven-woltmann/>



<https://youtube.com/c/happycoders>

