Stream Gatherers

Sander Mak

Software developer & architect

@sander_mak

Streams in Java

```
Stream.of("one", "two", "three", "four")
      .map(String::toUpperCase)
      .filter(s -> s.length() > 3)
      .toList()

> [THREE, FOUR]
```

Intermediate

Terminal

Streams in Java: Collectors

```
Stream.of("one", "two", "three", "four")
      .map(String::toUpperCase)
      .filter(s -> s.length() > 3)
      .collect(Collectors.toList()); // or Collectors.toSet()
```

```
> [THREE, FOUR]
```

Streams in Java: Collectors

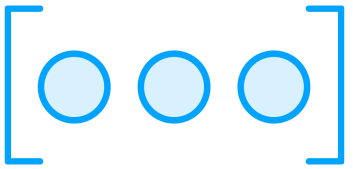
```
Stream.of("one", "two", "three", "four")
      .map(String::toUpperCase)
      .filter(s -> s.length() > 3)
      .collect(<any collector>);
```

```
Collectors.joining("; ")
> "THREE; FOUR"

Collectors.groupingBy(e -> e.substring(0, 1))
> {T=[THREE], F=[FOUR]}

Collectors.averagingInt(String::length)
> 4.5
```

Streams in Java

```
map
filter
limit
anyMatch
...
```

```
pair?
accumulate?
```



Streams in Java: Gatherers

```
Stream.of("one", "two", "three", "four")
      .gather(<any intermediate operation>)
      .toList()
```

Lazy
Parallel
Short-circuiting

Streams in Java: Gatherers

```
Gatherer<String, Void, String> gatherer = ...

Stream.of("one", "two", "three", "four")
      .gather(gatherer)
      .toList()
```

Built-in Gatherers

```
java.util.stream.Gatherers

      fold(initial, foldFunction)

      scan(initial, scanFunction)

      mapConcurrent(maxConcurrency, mapFunction)

      windowFixed(windowSize)

      windowSliding(windowSize)
```

Creating Gatherers

```
interface Gatherer<T, A, R> {

   Integrator<A,T,R> integrator()

   // several methods with default
   // implementation
}
```

T	Input element type
A	Internal state type
R	Output element type

Creating Gatherers

```
interface Integrator<A,T,R> {

   boolean integrate(A state, T element,
     Downstream<? super R> downstream)

}
```

Creating Gatherers

```
interface Integrator<A,T,R> {

   boolean integrate(A state, T element,
     Downstream<? super R> downstream)

}
```

Creating Gatherers

```
interface Integrator<A,T,R> {

   boolean integrate(A state, T element,
     Downstream<? super R> downstream)

}
```

Creating Gatherers

```
interface Integrator<A,T,R> {

   boolean integrate(A state, T element,
     Downstream<? super R> downstream)

}
```

Creating Gatherers: Next Steps

Managing mutable state

Chaining gatherers

Combining state for parallel gatherers

Stream Gatherers tutorial

https://dev.java/learn/api/streams/gatherers/