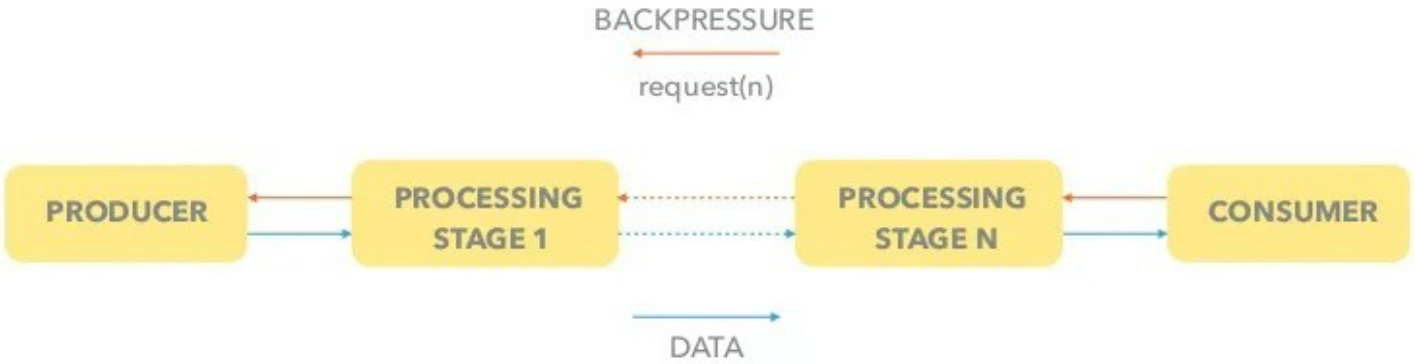
JACEK KUNICKI
@rucek

SOFTWAREMILL

OUT-OF-THE-BOX REACTIVE STREAMS WITH JAVA 9

https://github.com/rucek/reactive-streams-java9

STREAM PROCESSING



Publisher Processor Processor Subscriber

java.util.concurrent.Flow

`j.u.c.Flow.Publisher<T>`

- produces items of type T that subscribers are going to consume
- multiple subscribers receive items in the same order
- subscribers register via `subscribe(Subscriber<? super T>)`

`j.u.c.Flow.Subscriber<T>`

- subscribes to a producer in order to receive:
  - subscription confirmation via `onSubscribe(Subscription)`
  - items via `onNext(T)`
  - errors via `onError(Throwable)`
  - completion signal via `onComplete()`

`j.u.c.Flow.Subscription`

- connects a single producer to a single subscriber, allows to:
  - backpressure with `request(long)`
  - signal (eventual) termination with `cancel()`

`j.u.c.Flow.Processor<T, R>`

- a combination of a `Subscriber<T>` and a `Publisher<R>`

j.u.c.SubmissionPublisher<T>

- the only concrete implementation available so far
- asynchronously issues submitted (non-null) items to current subscribers
- can be used as a base for your own components

```
Publisher.subscribe(Subscriber)

onSubscribe
onNext*
(onComplete | onError)?
```


Talk is cheap
show me the
CODE
SOFTWAREMILL

DEMO 1 - PURE JAVA 9

- number publisher based on an `IntStream`
- filtering and mapping processors based on the `SubmissionPublisher`
- a subscriber that prints to `System.out`

DEMO 2 - INTEGRATION

- Project Reactor's `Flux` as a publisher
- Akka Streams `Flow` as a processor
- a pure Java 9 processor
- a pure Java 9 subscriber

SUMMARY

- not a full Reactive Streams implementation
- allows for interoperability between other implementations
- incubating Reactive Streams support in the new HTTP client

THANK YOU!

https://github.com/rucek/reactive-streams-java9

@rucek

SOFTWAREMILL