Reactive API Workshop

Daniel Hinojosa

About Me...

daniel.hinojosa

Independent
Consultant, Programmer.
Trainer, Author

(Beard Optional)

@dhinojosa
dhinojosa@evolutionnext.com

Code and Slides are available on Github

https://github.com/dhinojosa/nfjs-reactive-api-workshop

FLO
PROGRESSIVE
PROGRESSIVE
AUTOMOTIVE
X
PRIZE
The race to develop
100 MPGe cars
I ♥ INSURANCE
FLO
PROGRESSIVE

Flow

Flow API

- JEP 266 (http://openjdk.java.net/jeps/266)
- Minimal set of interfaces that capture the heart of asynchronous publication and subscription
- Interfaces contain that of what is in reactive-streams.org (Though they are not the same)

Components

- Publisher
- Subscriber
- Processor
- Subscription

Publisher

- The publisher publishes the stream of data items to the registered subscribers.
- It publishes items to the subscriber asynchronously, normally using an Executor.
- Publishers ensure that Subscriber method invocations for each subscription are strictly ordered.

https://community.oracle.com/docs/DOC-1006738

Subscriber

- The Subscriber subscribes to the Publisher for the callbacks.
- Data items are not pushed to the Subscriber unless requested, but multiple items may be requested.
- Subscriber method invocations for a given Subscription are strictly ordered.
- The application can react to the following callbacks, which are available on the subscriber

https://community.oracle.com/docs/DOC-1006738

Subscription

- Links a `Flow.Publisher` and `Flow.Subscriber`.
- Subscribers receive items only when requested, and may cancel at any time, via the `Subscription`

https://community.oracle.com/docs/DOC-1006738

Java 9

Flow Interface - Push Model

https://community.oracle.com/docs/DOC-1006738

Processor

- A component that acts as both a Subscriber and Publisher.
- The processor sits between the Publisher and Subscriber, and transforms one stream to another.
- There could be one or more processor chained together, and the result of the final processor in the chain, is processed by the Subscriber.
- The JDK does not provide any concrete Processors so it is left up to the individual to write whatever processor one requires.

https://community.oracle.com/docs/DOC-1006738

Backpressure def.

Back pressure refers to pressure opposed to the desired flow of a fluid in a confined place such as a pipe. It is often caused by obstructions or tight bends in the confinement vessel along which it is moving, such as piping or air vents.

Backpressure def.

Back pressure is when an asynchronous source is emitting items more rapidly than an operator or subscriber can consume them. This presents the problem of what to do with such a growing backlog of unconsumed items.

Java 9

Flow Interface with request and backpressure

https://community.oracle.com/docs/DOC-1006738

Flow Backpressure

The Flow API does not provide any APIs to signal or deal with back pressure as such, but there could be various strategies one could implement by oneself to deal with back pressure

Demo:
Showing the Interfaces
Java 9 Flow API

RXJava

ReactiveX

An API for asynchronous programming
with observable streams

Choose your platform

Reactive Extensions for Async Programming

RX
for Async Programming

RX Java

Definition

A library for composing asynchronous and event-based programs by using observable sequences.

Reactive Programming

Reacting to changes in data or events using functional abstractions

Reactive Programming

Responding to:

Mouse Clicks, Keyboard Events, Temperature Sensors, Stock Trades, Website Clicks, GPS Signals, HTTP Responses, MQ Messages, Kafka Messages, etc.

Unbiased

The source for any `Observable/Flowable` can from anywhere. Convert your arrays, futures, threads, into `Observables/Flowable`

formerly of....

Microsoft

"Mouse is a database"

Reactive Programming

NETFLIX

RXJava

- Push Model (Reactive)
- Pull Model (Interactive, Subscription Based)

Lazy

Synchronous/ Asynchronous

Synchronous by Default

Observer/
Iterator

Where can you use it?

Anywhere. It is a processing asynchronous glue in applications.

Find all your favorite operators at:

http://reactivex.io/documentation/operators

Demo: RXJava

Project Reactor

Project Reactor

- Non blocking reactive
- Integrates with Java 8/Java 9 Functional APIs
- Integrates with `CompletableFuture`
- Included with Spring 5

Demo:
Project Reactor

Akka Streams

Akka Streams

- Backed by Actor Model
- Stream processors much like their competitors RxJava, Project Reactor
- Builds on the idea of flows and flow graphs that define how a stream is processed
- Streams are built with reusable pieces either
  - Prepackaged components
  - Custom made composites

Actor-A

Actor-A

Actor-A
Actor-B

Future Answered
Actor-A
Actor-B

Restart!
Supervisor
Child-A
Child-B
IllegalStateException

Source[Int] .map(...) filter(...) to(Sink)

Demo:
Akka Streams

Akka Alpakka

Enterprise Extensions

https://developer.lightbend.com/docs/alpakka/current/

Thanks