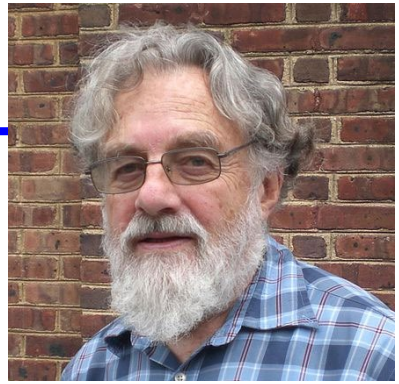
Reactive programming with Spring Webflux

Eonics Hack Night #21

João Esperancinha
joao.esperancinha@eonics.nl

Origins 1960



Jack Dennis:

- Data Flows concepts during the 1960's at MIT
- Data stream concept and iteractions
- Data Flow years 1974 – 1975
- Roots in:
  - Asynchronous Digital Logic
  - Control Structures for Parallel Programming
  - Abstract Models for Concurrent Systems
  - Theory of Program Schemes
  - Structured Programming
  - Functional Programming

Ref: http://csg.csail.mit.edu/Dataflow/talks/DennisTalk.pdf

Origins 2010

Eric Meijer:

- Coined the term Reactive Programming in 2010
- Microsoft included C#, Visual Basic, LINQ, Volta
- Reactive programming framework
- Reactive Extensions for .NET.
- Dataflow programming on steroids

Ref: https://channel9.msdn.com/Blogs/Charles/Erik-Meijer-Rx-in-15-Minutes

Reactive Manifesto

Principles (applies to system and applications):

- Responsive
  - It needs to respond quickly. The time of the request itself is independent of this.
- Resilient
  - It must respond well and support Back-Pressure
  - Messages in control
  - Avoid catastrophic failure
- Elastic
  - Automatic Generation of resources. More threads in our case.
- Message Driven
  - Publisher/Subscriber

Ref: https://www.reactivemanifesto.org/

Spring WebFlux Basics

Observer Pattern

Is a behavioral design pattern that lets you define a subscription mechanism to notify multiple objects about any events that happen to the object they're observing.

In other words, we are going to do declarative programming instead of imperative programming.

Ref: https://en.wikipedia.org/wiki/Observer_pattern

Spring WebFlux Publishers

- Flux
  - A publisher for a stream of objects
  - Used to create lists of objects]
  - Processes one stream end to end
  - Handles stream events
- Mono
  - A publisher for a single object
  - Handles object events

Flux.just, Flux.from, Flux.fromIterable, Flux.fromArray, Flux.fromStream, Flux.zip

Mono.just, Mono.from, Mono.fromCallable, Mono.zip, Mono.fromFuture, Mono.fromDirect, Mono.fromRunnable

Ref:https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html

eonics

Spring WebFlux Parallelism

- Flux and ParallelFlux
  - .parallel(parallelism).runOn(Schedulers.parallel())

- Mono
  - .subscribeOn(Schedulers.parallel())

Ref:https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html

eonics

Blockhound

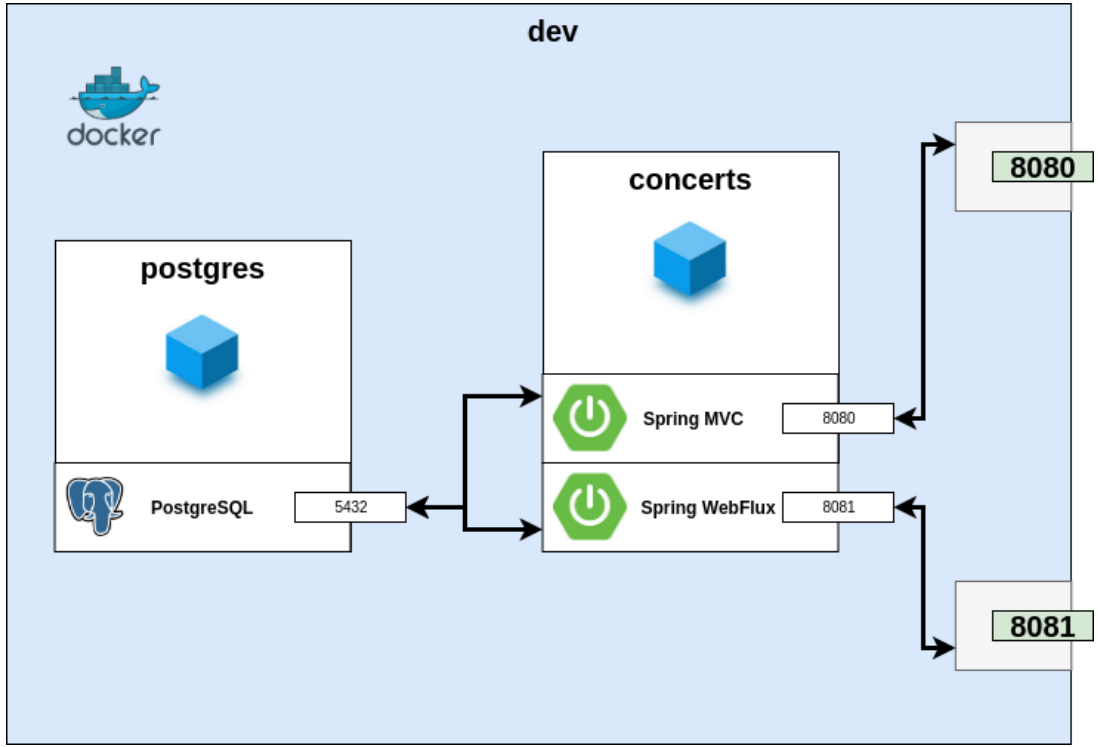
- Test Library
- Needs to be installed:

```
static {
    BlockHound.install();
}
```

- Detects Blocking calls:

```
Mono.delay(Duration.ofMillis(1))
            .doOnNext(it -> {
                try {
                    catController.getCatByIdI(1L);
                } catch (IOException e) {
                    throw new RuntimeException(e);
                }
            })
            .block();
```

Concerts Project



Tests with JMeter

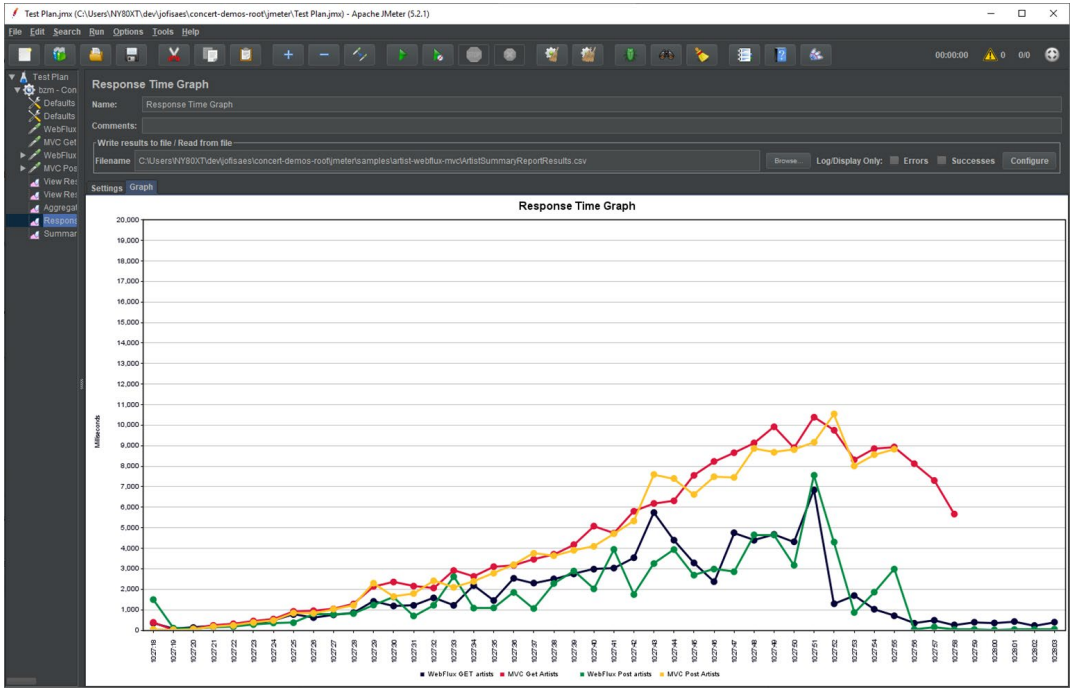
- Load tests
- Making the world a better place by registering thousands of Nicky Minajs 😉

```
{
  "name": "Nicky Minaj",
  "gender": "FEMALE",
  "careerStart": 1000,
  "birthDate": "a date",
  "birthCity": "Port of Spain",
  "country": "Trinidad en Tobago",
  "keywords": "Rap"
}
```

X 1000 / s

Comparing Blocking MVC and Reactive MVC

Results of analysis of Response Times



Comparing Blocking MVC and Reactive MVC

Results of analysis of number of requests:

	Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
	WebFlux GET artists	1818	1714	7	22403	2846.73	0.000%	39.69346	4512.75	5.31	116418.6
	MVC Get Artists	1510	3320	7	16520	3516.68	0.066%	33.09589	5812.04	4.42	179826.8
	WebFlux Post artists	1394	1425	4	14886	2597.78	0.000%	30.97984	10.26	12.49	339.1
	MVC Post Artists	1179	2813	5	20344	3265.32	0.000%	26.18371	8.69	10.56	340.0
	TOTAL	5901	2276	4	22403	3160.81	0.017%	128.35795	10282.47	32.26	82030.3

Let’s Code!

- Cat Care Center
- Bocco and Zuu have been found
- The application is blocking!
- Let’s make it reactive!
- Checkout the repo
- Checkout branch exercise
- Build will fail!
- Make the code reactive!
- Build will run!

git clone
https://jesperancinha@bitbucket.org/jesperancinha/eonics-hacknight-webflux.git
git checkout exercise
mvn clean install

eonics

Q?

References

- https://content.pivotal.io/springone-platform-2018/full-stack-reactive-with-react-and-spring-webflux
- https://bitbucket.org/jesperancinha/eonics-hacknight-webflux
- https://github.com/reactor/BlockHound
- https://github.com/jesperancinha/sea-shell-archiver
- https://github.com/jesperancinha/concert-demos-root
- http://csg.csail.mit.edu/Dataflow/talks/DennisTalk.pdf
- https://medium.com/swlh/comparing-webflux-and-spring-mvc-with-jmeter-79dc134c3c04
- https://medium.com/swlh/reactive-programming-applied-to-legacy-services-a-webflux-example-4d1c2ad40bd4
- https://en.wikipedia.org/wiki/Jack_Dennis
- https://www.reactivemanifesto.org/
- https://refactoring.guru/design-patterns/observer
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.csnip.org%2Fgeneral-faqs&psig=AOvVaw1jPWBfSEBh65Oci1MZyq9E&ust=1582141566798000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCPiZyOPu2-cCFQAAAAAdAAAAABBI
- https://www.instagram.com/pechanko_bocco/?hl=nl

João Esperancinha joao.esperancinha@eonics.nl

eonics