### Douglas C. Schmidt <u>d.schmidt@vanderbilt.edu</u> www.dre.vanderbilt.edu/~schmidt



**Professor of Computer Science** 

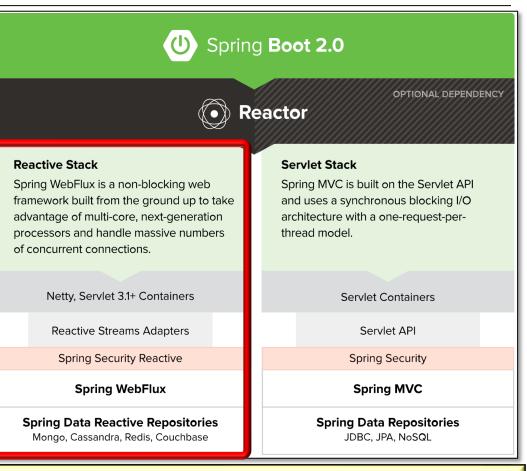
Institute for Software Integrated Systems

Vanderbilt University Nashville, Tennessee, USA



#### Learning Objectives in this Lesson

 Understand the structure & functionality of the Spring WebFlux framework supported by Spring Boot 2.0



See <u>docs.spring.io/spring-framework/docs/current/reference/html/web-reactive.html#webflux</u>

#### • Spring WebFlux

## (U) Spring Boot 2.0



#### OPTIONAL DEPENDENCY

#### **Reactive Stack**

Spring WebFlux is a non-blocking web framework built from the ground up to take advantage of multi-core, next-generation processors and handle massive numbers of concurrent connections.

#### Servlet Stack

Spring MVC is built on the Servlet API and uses a synchronous blocking I/O architecture with a one-request-perthread model.

Netty, Servlet 3.1+ Containers

**Reactive Streams Adapters** 

Spring Security Reactive

Spring WebFlux

Spring Data Reactive Repositories Mongo, Cassandra, Redis, Couchbase Servlet Containers

Servlet API

Spring Security

Spring MVC

Spring Data Repositories JDBC, JPA, NoSQL

See <u>docs.spring.io/spring-framework/docs/current/reference/html/web-reactive.html#webflux</u>

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections

Netty, Servlet 3.1+ Containers

**Reactive Streams Adapters** 

Spring Security Reactive

#### Spring WebFlux

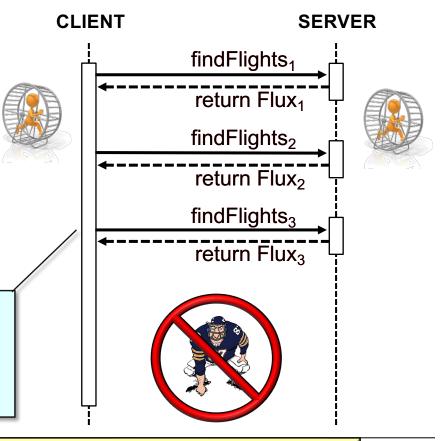
#### **Spring Data Reactive Repositories**

Mongo, Cassandra, Redis, Couchbase

See en.wikipedia.org/wiki/Non-blocking\_algorithm

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner

A request to a list of flights from a database over the network might take a few seconds, but the threads servicing requests & responses don't block



See <a href="mailto:en.wikipedia.org/wiki/Asynchrony\_(computer\_programming)">en.wikipedia.org/wiki/Asynchrony\_(computer\_programming)</a>

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner
    - A small # of threads are thus required



See www.baeldung.com/spring-webflux-concurrency

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner
    - A small # of threads are thus required
      - Typically based on # of processor cores





- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner
    - A small # of threads are thus required
      - Typically based on # of processor cores
      - I/O-bound operations may require adaptively increasing the # of threads

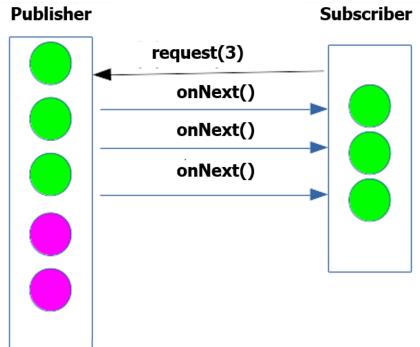




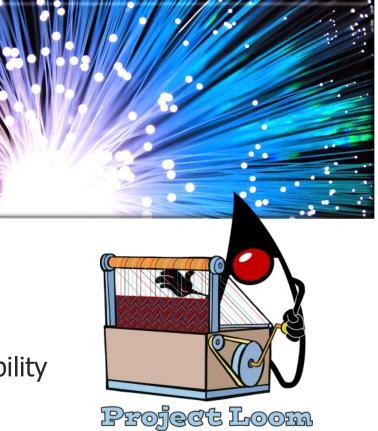
See projectreactor.io/docs/core/release/api/reactor/core/scheduler/Schedulers.html#boundedElastic-

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner
    - A small # of threads are thus required
    - However, there's often a need for nonblocking backpressure
      - i.e., control event rate so a fast publisher does not overwhelm a slower subscriber



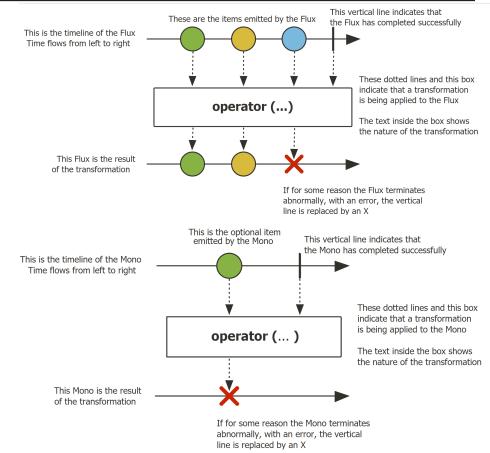


- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
    - Requests are handled in an entirely asynchronous (& "lazy") manner
    - A small # of threads are thus required
    - However, there's often a need for nonblocking backpressure
    - Java 19's "virtual threads" provide scalability without non-blocking clients & servers



See <u>fabiangotzen.net/2023/01/19/java-project-loom</u>

- Spring WebFlux
  - A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
  - Network communication uses Project Reactor reactive types



See <a href="mailto:spring.io/blog/2016/04/19/understanding-reactive-types">spring.io/blog/2016/04/19/understanding-reactive-types</a>

#### • Spring WebFlux

- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses Project Reactor reactive types
  - Work on data sequences of 0..1 (Mono) and 0..N (Flux)

public class FlightController {

@GetMapping(FLIGHT\_DATES)
Flux<LocalDate>

findDepartureDates

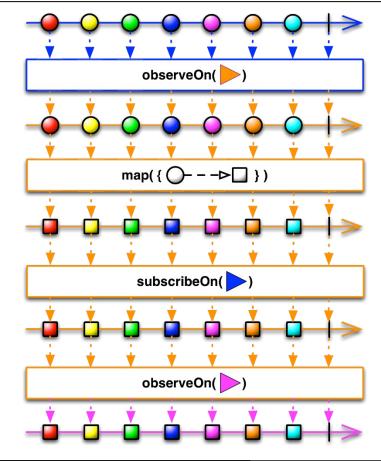
(@RequestParam String departureAirport, @RequestParam String arrivalAirport) {...}

@GetMapping(EXCHANGE)
Mono<ExchangeRate> getRate
 (@RequestParam String from,
 @RequestParam String to)
 {...}

See <a href="flights-reactive-microservices/-/blob/master/src/main/java/server/flight/FlightController.java">flights-reactive-microservices/-/blob/master/src/main/java/server/flight/FlightController.java</a>

#### • Spring WebFlux

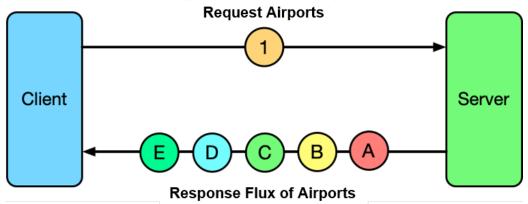
- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses Project Reactor reactive types
  - Work on data sequences of 0..1 (Mono) and 0..N (Flux)
  - Provides a rich set of operators aligned with the ReactiveX vocabulary of operators



#### See <a href="mailto:reactivex.io/documentation/operators.html">reactivex.io/documentation/operators.html</a>

#### • Spring WebFlux

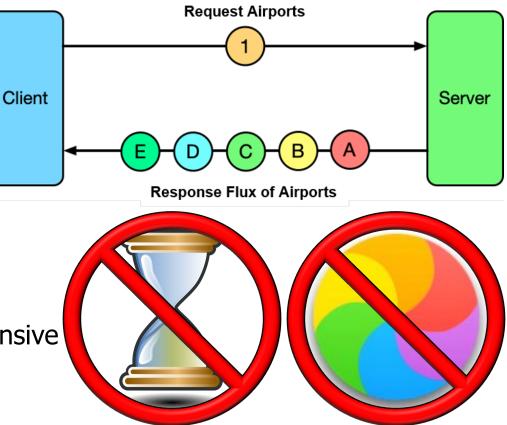
- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses
   Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element



#### See processing-streaming-data-with-spring-webflux-ed0fc68a14de

#### • Spring WebFlux

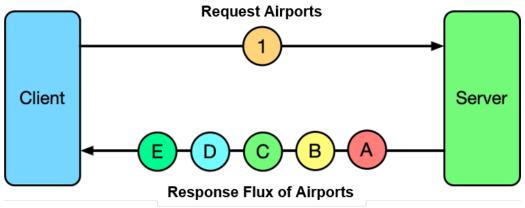
- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses
   Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element
  - Clients can thus be more responsive



#### See <u>en.wikipedia.org/wiki/Responsiveness</u>

#### • Spring WebFlux

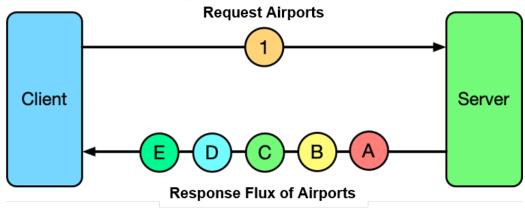
- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses
   Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element
  - Clients can thus be more responsive
    - Android retrofit doesn't support WebFlux reactive clients..





#### • Spring WebFlux

- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses
   Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element
  - Clients can thus be more responsive
    - Android retrofit doesn't support WebFlux reactive clients..
    - Spring 6 HTTP interface also doesn't work on Android





#### • Spring WebFlux

- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses
   Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element
  - Clients can thus be more responsive
  - The stream can be kept "live" via Spring server-sent-events

#### See <a href="https://www.baeldung.com/spring-server-sent-events">www.baeldung.com/spring-server-sent-events</a>

...
@GetMapping(RATES,
 produces = MediaType.
 TEXT\_EVENT\_STREAM\_VALUE) {
Flux<ExchangeRate>
 getRates(@RequestParam
 String toCurrency)
 {...}

public class FlightController {

### 

O Executions

Mean reg/s

🕑 Response Time (

50th percentile

75th percentile

95th percentile 99th percentile

Std Deviation

Min

Max Mean

#### • Spring WebFlux

- A non-blocking web framework that leverages multiple cores & handles large # of concurrent connections
- Network communication uses Project Reactor reactive types
- Reactive Flux types can be streamed element-by-element
  - Clients can thus be more responsive
  - The stream can be kept "live" via Spring server-sent-events

The reactive stack can have better response time & the # of request per second

See springboot-2-performance-servlet-stack-vs-webflux-reactive-stack-528ad5e9dadc

view	or Spr	ing we	DFIUX
		Servlet Stack	

let Sta	ck		► STATISTICS	Reactive Sta	ck	
			C Executions	;		
otal	ОК	КО		Total	ОК	КО
10000	39994	6		40000	40000	0
4.314	784.196	0.118	Mean req	/s 975.61	975.61	
ms)			Sesponse <sup>™</sup>	Time (ms)		
otal	ОК	ко		Total	ОК	КО
202	202	2443	Mi	in 201	201	
2093	2093	3151	50th percentil	le 315	315	
3017	3017	3763	75th percentil	le 422	422	
4547	4547	4104	95th percentil	le 865	865	
5061	5061	4154	99th percenti	le 1392	1392	
5356	5356	4166	Ma	ix 1883	1883	
2102	2101	3216	Mea	in 390	390	
1304	1304	660	Std Deviatio	on 232	232	

## End of Overview of Spring WebFlux