Overview of Spring WebFlux

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
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 [docs.spring.io/spring-framework/docs/current/reference/html/web-reactive.html#webflux]
Overview of Spring WebFlux
Overview of Spring WebFlux

- Spring WebFlux

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.

Netty, Servlet 3.1+ Containers
Reactive Streams Adapters
Spring Security Reactive
Spring WebFlux

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

Servlet Containers
Servlet API
Spring Security
Spring MVC

Spring Data Reactive Repositories
Mongo, Cassandra, Redis, Couchbase

See [docs.spring.io/spring-framework/docs/current/reference/html/web-reactive.html#webflux]
Overview of Spring WebFlux

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

See [en.wikipedia.org/wiki/Non-blocking_algorithm](en.wikipedia.org/wiki/Non-blocking_algorithm)
Overview of Spring WebFlux

- 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 en.wikipedia.org/wiki/Asynchrony_(computer_programming)
Overview of Spring WebFlux

• 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](http://www.baeldung.com/spring-webflux-concurrency)
Overview of Spring WebFlux

- **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
Overview of Spring WebFlux

• 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
Overview of Spring WebFlux

- 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 non-blocking backpressure
      - i.e., control event rate so a fast publisher does not overwhelm a slower subscriber

See [www.baeldung.com/spring-webflux-backpressure](http://www.baeldung.com/spring-webflux-backpressure)
Overview of Spring WebFlux

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

See [spring.io/blog/2016/04/19/understanding-reactive-types](spring.io/blog/2016/04/19/understanding-reactive-types)
Overview of Spring WebFlux

• **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)

```java
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)
        {...}
}
```

Overview of Spring WebFlux

- 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 reactivex.io/documentation/operators.html
Overview of Spring WebFlux

- **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 medium.com/@nithinmallya4/processing-streaming-data-with-spring-webflux-ed0fc68a14de
Overview of Spring WebFlux

- **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 [en.wikipedia.org/wiki/Responsiveness](en.wikipedia.org/wiki/Responsiveness)
Overview of Spring WebFlux

- **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

```java
public class FlightController {
    ...
    @GetMapping(RATES,
        produces = MediaType.
        TEXT_EVENT_STREAM_VALUE)
    Flux<ExchangeRate>
    getRates(@RequestParam
        String toCurrency)
    {
        ...
    }
    ...

See [www.baeldung.com/spring-server-sent-events](http://www.baeldung.com/spring-server-sent-events)
Overview of Spring WebFlux

• 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 medium.com/@the.raj.saxena/springboot-2-performance-servlet-stack-vs-webflux-reactive-stack-528ad5e9dadc
End of Overview of Spring WebFlux