Building a REST API with Annotated Controllers

Esteban Herrera
Author | Developer | Consultant

@eh3rrera   eherrera.net

WIRED BRAIN
COFFEE

Product Catalog

- Get all products
- Get a specific product
- Register a new product
- Update a product
- Delete a product
- Delete all products
- Events

Overview

Spring WebFlux annotated controllers

Setting up the project with Spring Boot

Reactive Spring Data with MongoDB

Initializing the embedded database

Building the controller

Server-side events

Annotated Controllers

Controllers

@Controller

@RestController

Request Mapping

@GetMapping

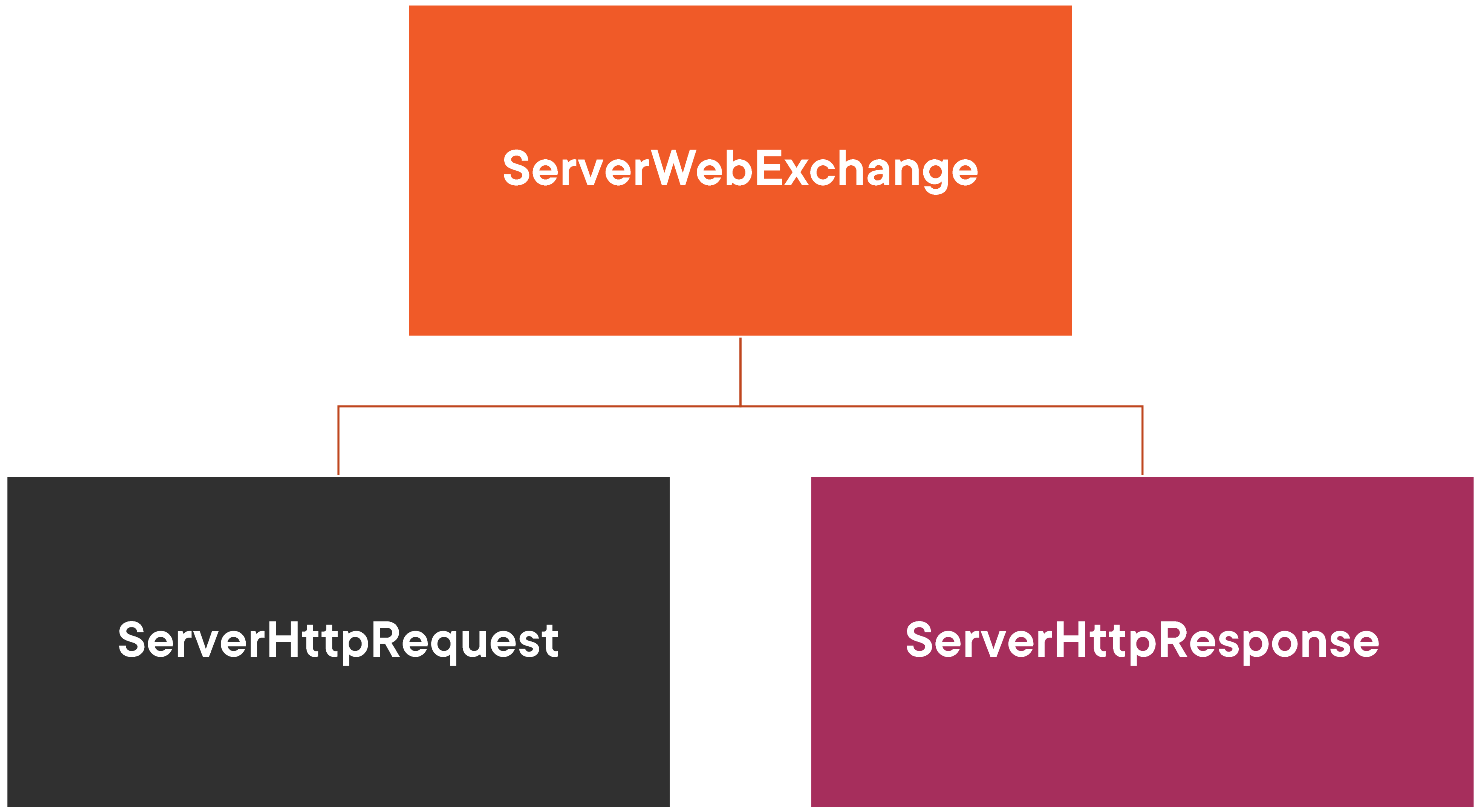
@PostMapping

@PutMapping

@DeleteMapping

@PatchMapping

Reactive Request and Response



Method Arguments with Reactive Support

WebSession

java.security.Principal

@RequestBody

HttpEntity

@RequestPart

Reactive types are supported for all return values

Return Values

ResponseEntity

Object

ServerSentEvent

String

- **@RestController**
  - **Mono<String>**
- **@Controller**
  - **View name**

Return Values

void, Mono<Void>, or returning null

- Response handled
  - ServerHttpResponse
  - ServerWebExchange
  - @ResponseStatus
- @RestController
  - No response body
- @Controller
  - Default view name

Execute Code after Spring Boot App Starts Up

CommandLineRunner

ApplicationRunner

Things to Remember

REST API with annotations

- Create
- Read
- Update
- Delete
- Server-side events

Similar to Spring MVC

Non-blocking/asynchronous model

- Easier to scale

Spring Boot

Reactive Spring Data