3. WebFlux Reactive Service Bean

Introduction

WebFlux ImageService . 2 3 4. Reactive Spring Data mongoDB .

- 1. Reactive Spring WebFlux? Spring MVC ?
  - (1)
  - (2)
- 2. Domain
- 3. ImageService
- Before Next Step..

Prerequisite

, Lombok 2 .

---

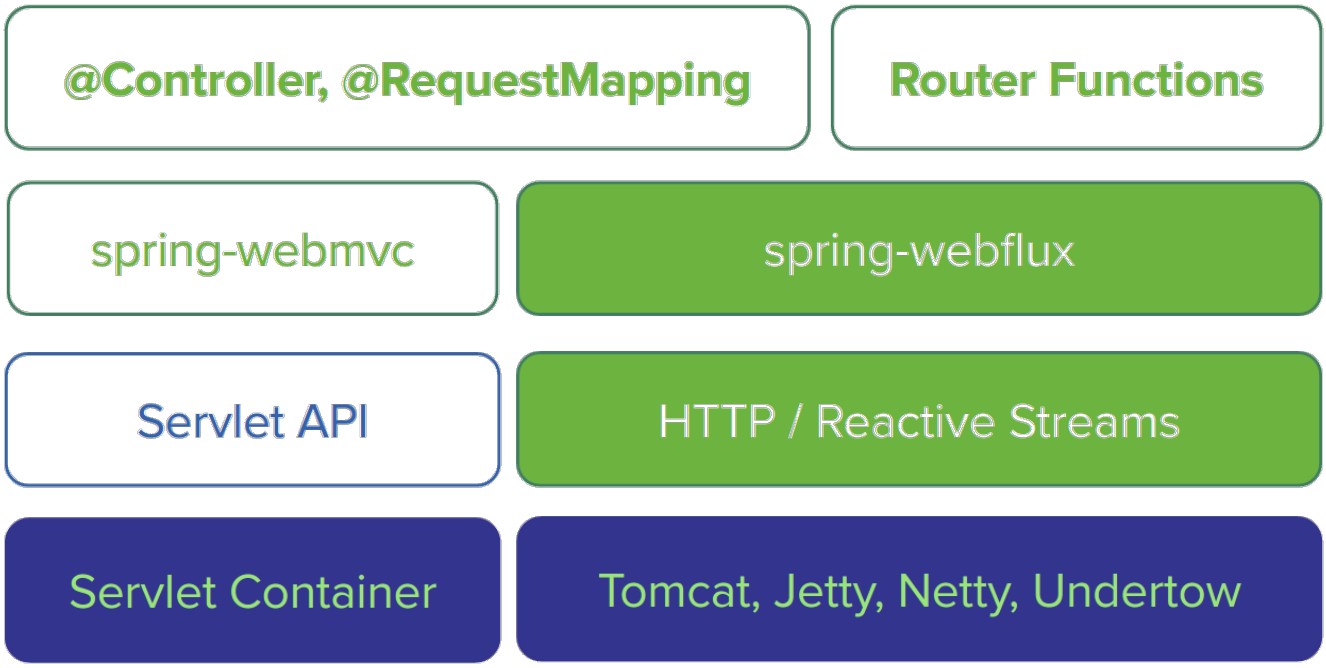
1. Reactive Spring WebFlux? Spring MVC ?

Spring MVC . Spring MVC . WebFlux .

Spring MVC Java EE Servlet spec , . 3 , pool , stack reactive . 5 WebFlux, reactive .

(1)

Spring MVC Spring WebFlux .



MVC , WebFlux , Netty, Undertow , HTTP Reactive Stream .

WebFlux HTTP abstractions, Reactive Stream apdater, Reactive codes non-blocking servlet api core web api . server-side WebFlux .

- Annotated Controller : Spring MVC spring-web , Spring MVC .
- Functional Endpoints : Java 8 lambda style routing handling . routing request , callback annotated controller .

Annotated Controller . Functional Endpoints , Spring Docs - Functional Endpoints .

<SpringBootApplication>

```
@SpringBootApplication
@EnableWebFlux
public class ExampleApplication {

    @Bean
    HelloHandler helloHandler() {
        return new HelloHandler();
    }

    @Bean
    RouterFunction<ServerResponse> helloRouterFunction(HelloHandler helloHandler) {
        return RouterFunctions.route(RequestPredicates.path("/"), helloHandler::handleRequest);
    }

    public static void main(String[] args) throws Exception {
        SpringApplication.run(ExampleApplication.class);
    }
}
```

<Handler >

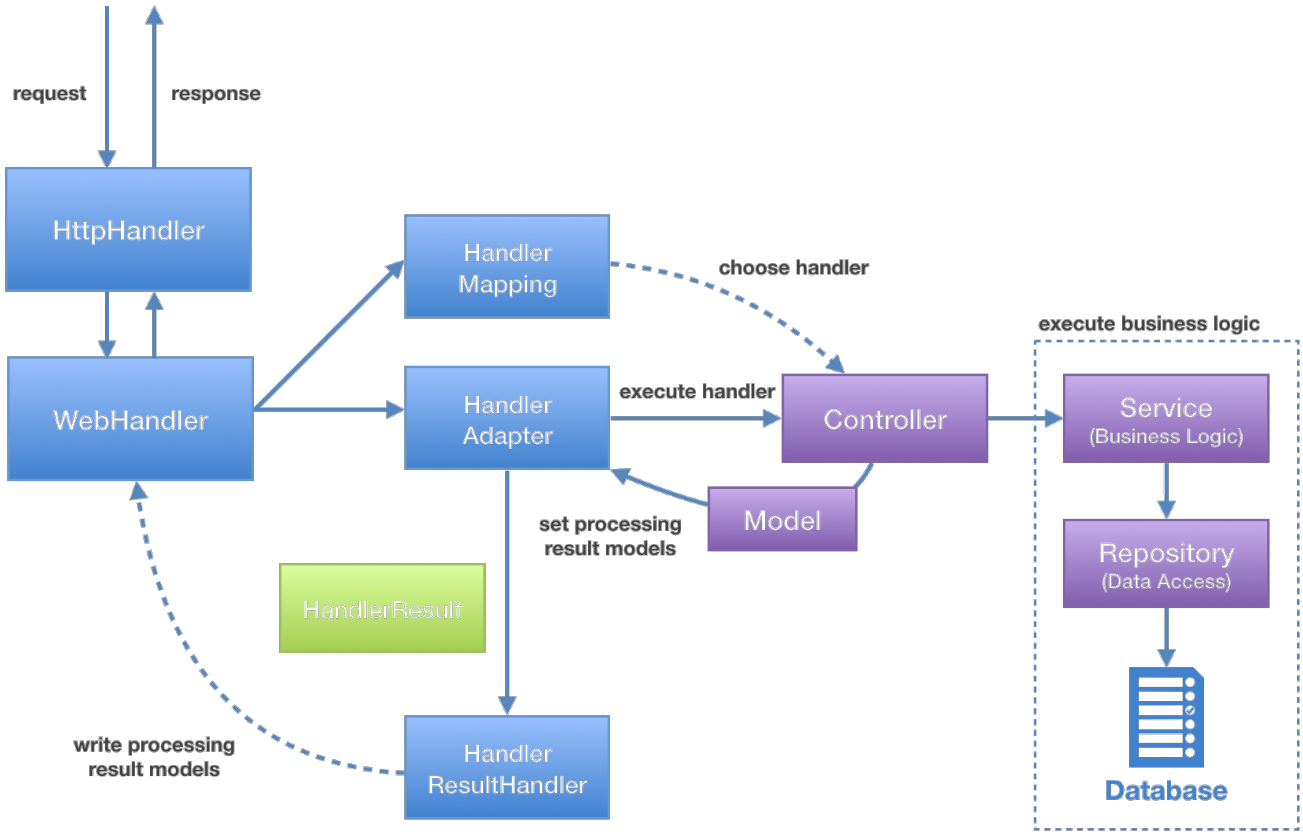
```
public class HelloHandler {
    public  Mono<ServerResponse> handleRequest(ServerRequest serverRequest) {
        return ServerResponse.ok().body(Mono.just("Hello World!"), String.class);
    }
}
```

(2)

Spring MVC

, HandlerMapping, HandlerAdapter , ViewResolver .

Spring WebFlux .



(Servlet Container Netty, Undertow) HttpHandler HttpHandler WebHandler . WebHandler HandlerMapping, HandlerAdapter, HandlerResultHandler 3 . HttpHandler . Spring MVC HandlerMapping, HandlerAdapter , , .

MVC:

- org.springframework.web.servlet.HandlerMapping
- org.springframework.web.servlet.HandlerAdapter

WebFlux:

- org.springframework.web.reactive.HandlerMapping
- org.springframework.web.reactive.HandlerAdapter

---

2. Domain

Reactive Codes .

```
package com.wiki.reactivewithspringboot.reactivewithspringboot;
import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.mapping.Document;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import lombok.ToString;

@Data
@Document
@NoArgsConstructor
@AllArgsConstructor
public class Image {
    @Id private String id;
    private String name;

    public Image(String id) {
        this.id = id;
    }
}
```

id name , lombok .

lombok , bean setter/getter/toString/hashCode/equals , . lombok , lombok .

- @Data : private @Getter, @Setter, @ToString, @EqualsAndHashCode, @RequiredArgsConstructor .
  - @Getter, @Setter : getter/setter
  - @ToString : toString
  - @EqualsAndHashCode : equals hashcode
  - @RequiredArgsConstructor : final , @NotNull , @NotNull null .
- @NoArgsConstructor : .
- @AllArgsConstructor : .

@Document @Id lombok . Spring Data 4 .

---

3. ImageService

web apps . ImageService .

ImageService .

```
@Service
public class ImageService {
    public static String UPLOAD_ROOT = "upload-dir";

    private final ResourceLoader resourceLoader;
    public ImageService(ResourceLoader resourceLoader) {
        this.resourceLoader = resourceLoader;
    }
    ...
}
```

- @Service : Service Spring bean . Spring boot , .
- UPLOAD_ROOT : .
- ResourceLoader : Spring utility class. Spring Boot service .

Spring Reactive Data , .

```
@Bean
CommandLineRunner setUp() throws IOException {
        return (args) -> {
                FileSystemUtils.deleteRecursively(new File(UPLOAD_ROOT));
        Files.createDirectory(Paths.get(UPLOAD_ROOT));
        FileCopyUtils.copy("Test file1", new FileWriter(UPLOAD_ROOT + "/test1.jpg"));
        FileCopyUtils.copy("Test file2", new FileWriter(UPLOAD_ROOT + "/test2.jpg"));
        FileCopyUtils.copy("Test file3", new FileWriter(UPLOAD_ROOT + "/test3.png"));
        };
}
```

- @Bean : ImageService Spring bean .
- bean CommandLineRunner, Spring Application . Spring Boot realize CommandLineRunner run.
- Java 8 lambda , Java 8 SAM(Single Abstract Method) CommandLineRunner . (Functional Interface Lambda Functional Interface Lambda Expression)

CRUD .

, findAllImages .

```
public Flux<Image> allImages() {
        try {
                return Flux.fromIterable(
                        Files.newDirectoryStream(Paths.get(UPLOAD_ROOT)))
            .map(path ->
                    new Image(Integer.toString(path.hashCode()),
                            path.getFileName().toString()));
        } catch (IOException e) {
                return Flux.empty();
        }
}
```

- Flux<Image> , image consumer subscribe .
- Files nio reactive . Files.newDirectoryStream UPLOAD_ROOT lazy DirectoryStream . DirectoryStream next() (recursive) , Reactor Flux .
- Flux.fromIterable lazy iterable , reactive streams client item .
- Image Exception Flux .

(Flux, Mono Part II - Reactor .)

, .

```
public Mono<Resource> getOneImage(String filename) {
        return Mono.fromSupplier(() ->
                resourceLoader.getResource(
                        "file:" + UPLOAD_ROOT + "/" + filename));
}
```

- Mono<Resource> . Resource .
- subscribe , Mono.fromSupplier , getResource lambda .

.

```
public Mono<Void> uploadImage(Flux<FilePart> files) {
        return files.flatMap(file -> file.transferTo(
                Paths.get(UPLOAD_ROOT, file.filename()).toFile())).then();
}
```

- , Mono<Void> return value .
- FilePart Flux flatMap .
- .
- FilePart content UPLOAD_ROOT .
- then() Flux Mono<Void> .

.

```
public Mono<Void> deleteImage(String filename) {
        return Mono.fromRunnable(() -> {
                try {
                        Files.deleteIfExists(Paths.get(UPLOAD_ROOT, filename));
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
        });
}
```

- return value Mono<Void> .
- subscribe , Mono.fromRunnable , lambda expression Runnable .
- Java.NIO Files.deleteIfExists .

Before Next Step..

ImageService . Spring Data 4 .

, . . callback hell reactive .

References

https://www.jerriepelser.com/blog/aspnet-core-aws-lambda-serverless-application/

Related Information

https://docs.spring.io/spring/docs/5.0.0.BUILD-SNAPSHOT/spring-framework-reference/html/web-reactive.html

https://projectlombok.org/

Content by label

There is no content with the specified labels