PROGRAMMERS'
WEEK

softvision

Spring WebFlux

Reactive web programing

PROGRAMMERS' WEEK
softvision

Spring WebFlux

- Why it is needed?
  - Non-blocking web stack to handle concurrency with a small number of threads
  - Scale with fewer hardware resources
  - Non-blocking apps getting more popular
    - CompletableFuture
    - ReactiveX
  - Functional programming
    - See Java 8 additions

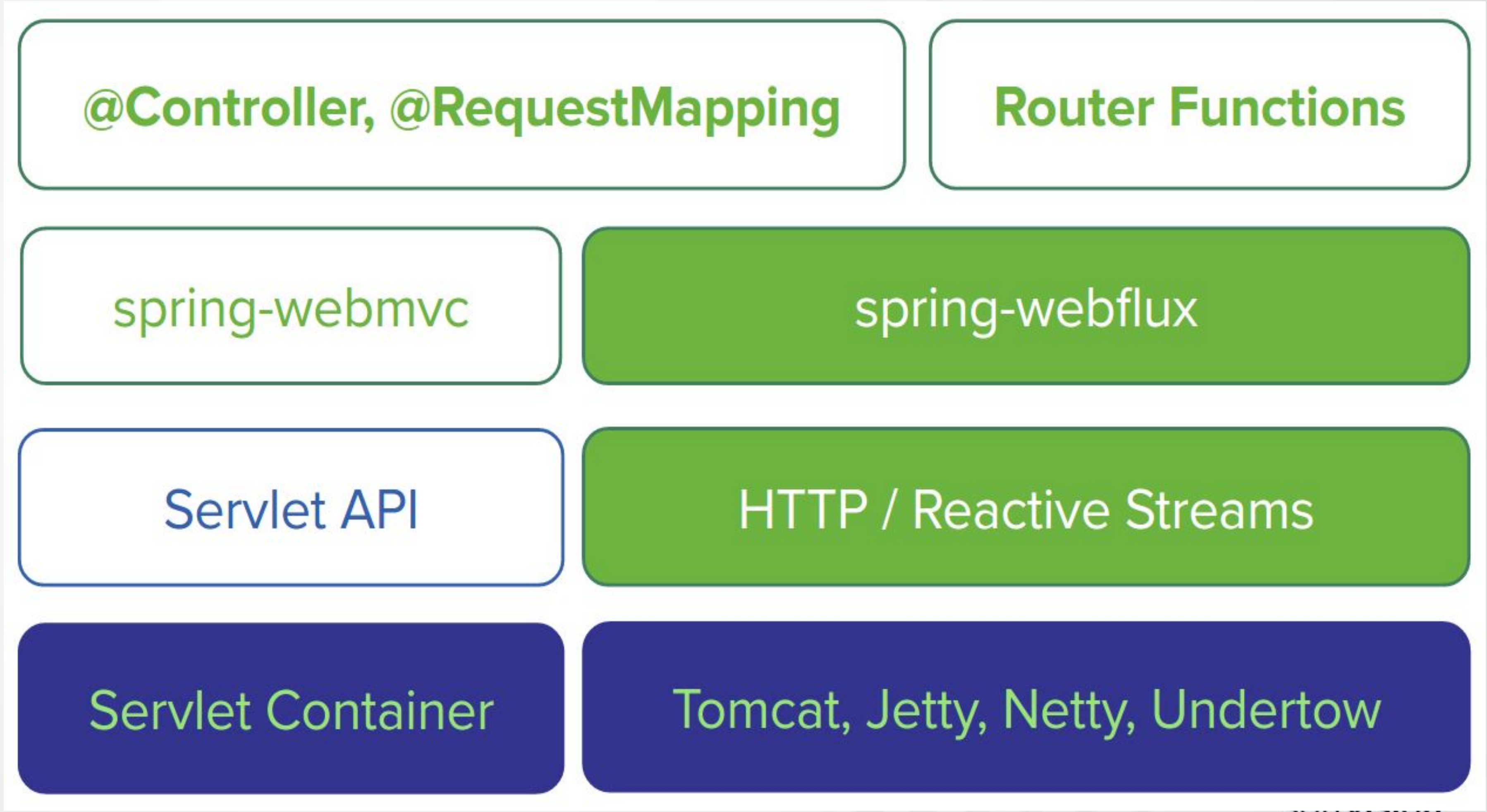
PROGRAMMERS' WEEK
softvision

Building blocks

- Reactive?
  - Non-blocking
  - Supports Reactive Stream back pressure

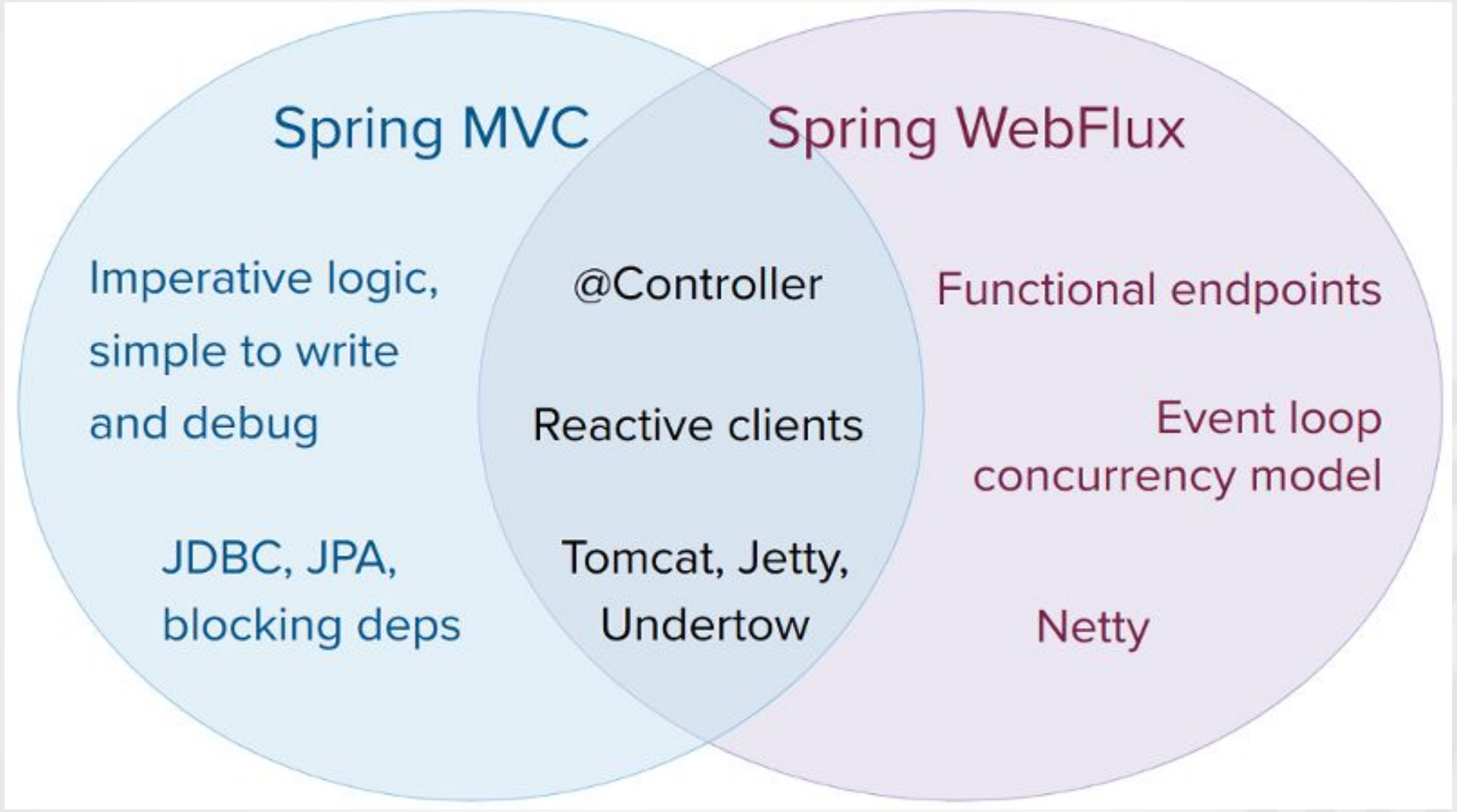
WebFlux Building blocks

- Project Reactor
  - Mono and Flux as Publishers
- Specific back end web server implementation



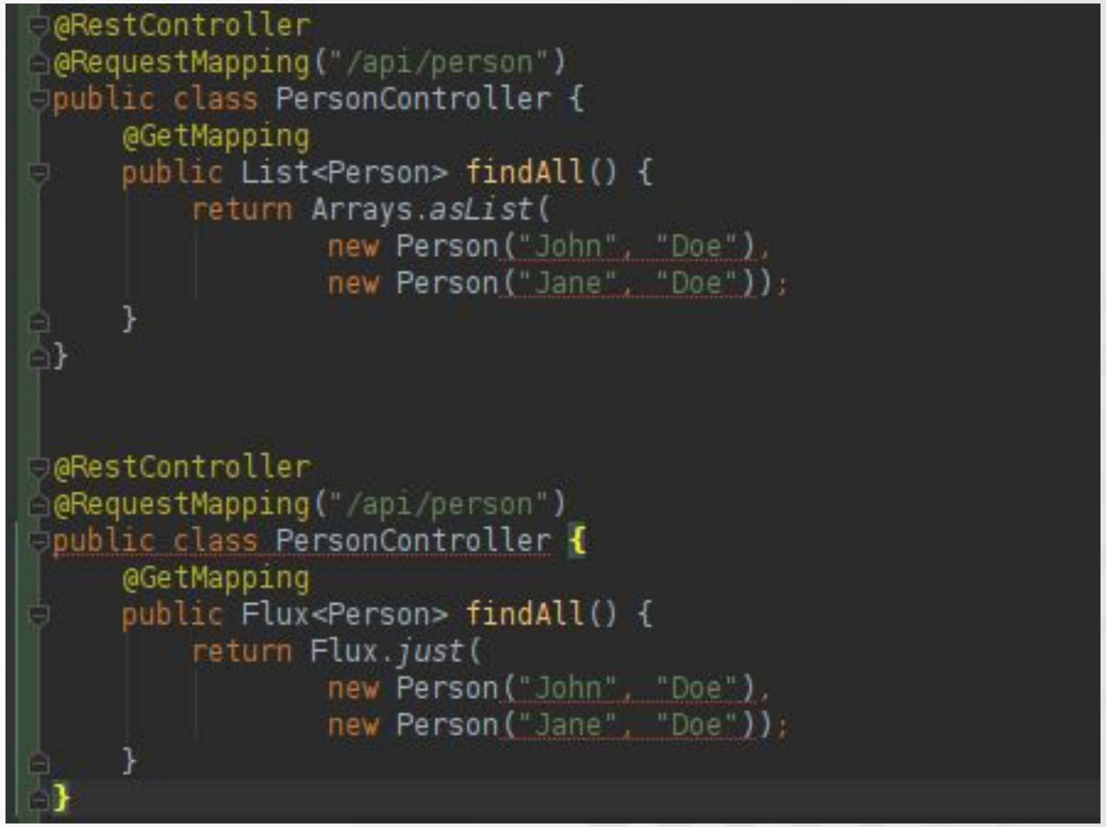
Building blocks

- Built over project Reactor - reactive streams



Programming models

- Annotated controllers
  - Consistent with Spring MVC and based on the same annotations from the spring-web module



```java
@RestController
@RequestMapping("/api/person")
public class PersonController {
    @GetMapping
    public List<Person> findAll() {
        return Arrays.asList(
                new Person("John", "Doe"),
                new Person("Jane", "Doe"));
    }
}


@RestController
@RequestMapping("/api/person")
public class PersonController {
    @GetMapping
    public Flux<Person> findAll() {
        return Flux.just(
                new Person("John", "Doe"),
                new Person("Jane", "Doe"));
    }
}
```

- Functional Controllers
  - Lambda based and

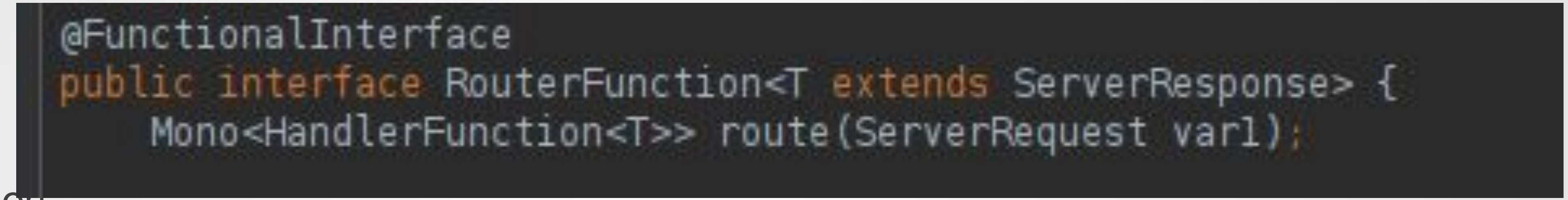
    functional programming models

PROGRAMMERS' WEEK

softvision

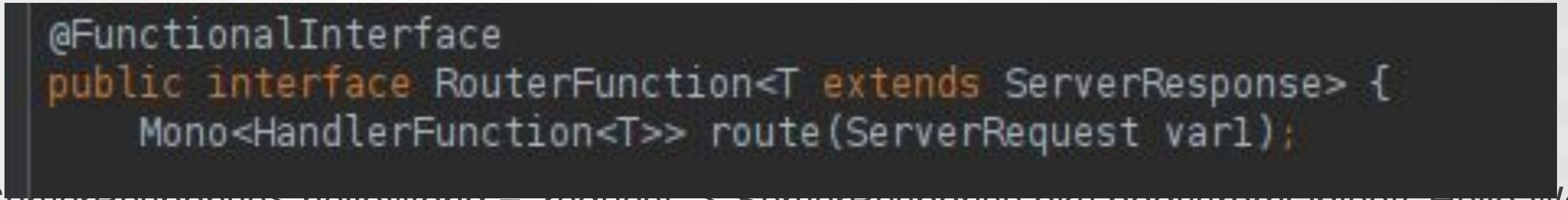
Functional Stype

- RouterFunction



```
@FunctionalInterface
public interface RouterFunction<T extends ServerResponse> {
    Mono<HandlerFunction<T>> route(ServerRequest var1);
```

- HandlerFunction



```
@FunctionalInterface
public interface RouterFunction<T extends ServerResponse> {
    Mono<HandlerFunction<T>> route(ServerRequest var1);
```

```
HandlerFunction<ServerResponse> helloWorld = request -> ServerResponse.ok().body(fromObject("Hello World"));

RouterFunction<ServerResponse> helloWorldRoute =
    RouterFunctions.route(RequestPredicates.path("/hello-world"),
    request -> Response.ok().body(fromObject("Hello World")));

RouterFunction<ServerResponse> helloWorldRoute2 =
    RouterFunctions.route(RequestPredicates.path("/hello-world"),
    this::helloWorld);
```

PROGRAMMERS' WEEK
softvision

References

- https://docs.spring.io/spring-framework/docs/5.0.0.BUILD-SNAPSHOT/spring-framework-reference/html/web-reactive.html
- https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html#webflux-controller
- https://github.com/poutsma/web-function-sample

PROGRAMMERS' WEEK
softvision