



KOTLIN REST APPLICATIONS

ANDY BOWES

THE JVM THING

20TH JULY 2017



BENEFITS OF KOTLIN

- Existing Java knowledge is transferable
 - Build & deploy pipeline is unchanged
 - Gradle & Maven Integration
- Expressiveness
 - Removes Java boilerplate
 - Clear & Concise Code



SPRING INTEGRATION

- Kotlin & Spring just works
 - All the familiar annotations & patterns
- Added dedicated support in Spring Framework 5.0
 - Pivotal are committed to Kotlin
 - Sebastien Deleuze champions the language
- Spring has tagged many methods as `@NotNull`
- Added Kotlin extension functions



SPRING

- No need to declare your bean class as open anymore
- Classes with these annotations are automatically 'open'
 - `@Component`, `@Async`, `@Transactional`, `@Cacheable`
 - `@Controller`, `@RestController`, `@Service` or `@Repository` are automatically opened since these annotations are meta-annotated with `@Component`.



SPRING BOOT

```
@SpringBootApplication  
open class SimpleApplication  
  
fun main(args: Array<String>) {  
    SpringApplication.run(Application::class.java, *args)  
}
```



SPRING - JPA DATA REPOSITORIES

```
@Entity
data class Movie(@Id val id: Long,
    val title: String,
    val director: String)

interface MovieRepository : CrudRepository<Movie, Long> {
    fun findByTitle(title: String): List<Movie>
    fun findByDirector(director: String): List<Movie>
}
```



SPRING WEB MVC

```
@RestController
class MovieController (val repository:MovieRepository) {

    @GetMapping("/")
    fun findAll() = repository.findAll()

    @GetMapping("/{title}")
    fun findByTitle(@PathVariable title:String)
        = repository.findByTitle(title)

    @GetMapping("/director/{director}")
    fun findByDirector(@PathVariable director:String)
        = repository.findByDirector(director)
}
```



SPRING INITIALIZR

- Simplifies the creation of Spring Boot projects
- Available online via <https://start.spring.io/>
- Kotlin is now supported by Spring Initializr
- Incorporated in IntelliJ



SPRING INITIALIZR DEMO

- Live Demo
 - Creation of simple application
 - Data classes
 - MongoDB Access
 - MVC Controller



DEPLOYMENT

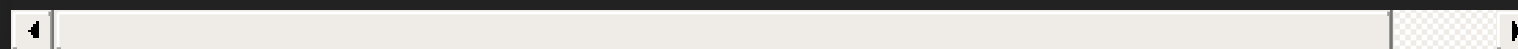
- Kotlin compiles to Java byte-code
- Builds typical Java JAR file
- Add Kotlin Dependencies
 - kotlin-stdlib
 - additional libs for Java 7 & Java 8 features



DOCKER DEPLOYMENT

- Identical to Java Spring application
- Example Dockerfile

```
FROM openjdk:alpine
EXPOSE 8080
ADD build/libs/myapplication-0.0.1.jar app.jar
RUN sh -c 'touch /kmdb.jar'
ENTRYPOINT [ "sh", "-c", "java $JAVA_OPTS -jar /app.jar"
```





SPRING WEBFLUX

- Spring WebFlux
 - Reactive Alternative to Spring Web MVC
- Spring Framework 5
 - Reactor
- Reactive Streams
 - Mono - Single elements
 - Flux - Streams of elements
- MongoDB or Redis Database Support



REACTIVE CONTROLLER MAPPING

- Use Routing DSL to provide mapping of URIs to functions

```
{  
  ("/movies" and accept(APPLICATION_JSON)).nest {  
    GET("/", movieHandler::findAll)  
    GET("/{id}", movieHandler::findOne)  
    GET("/actor/{actor}", movieHandler::findByActor)  
  }  
}
```



REACTIVE HANDLER FUNCTIONS

@Component

```
class MovieHandler(val repository: MovieRepository) {  
    fun findOne(req: ServerRequest) =  
        ok().json().body(repository.findOne(req.pathVariable("id")))  
    fun findAll(req: ServerRequest) =  
        ok().json().body(repository.findAll())  
    fun findByActor(req: ServerRequest) =  
        ok().json().body(repository.findByActor(req.pathVariable("actor")))  
}
```

@Repository

```
class MovieRepository(val template: ReactiveMongoTemplate,  
    val objectMapper: ObjectMapper) {  
    fun findAll(): Flux<Movie> = template.find<Movie>(Query().with(Sort.by("title")))  
    fun findOne(id: String): Mono<Movie> = template.findById<Movie>(id)  
    fun findByActor(id: String): Flux<Movie> = template.findByActor<Movie>(id)
```



ALTERNATIVE FRAMEWORKS

- Spring is not the only option
- Integrates with other Java frameworks
 - Spark
 - Vertx
- Kotlin specific frameworks
 - Ktor



SPARK

- Lightweight, expressive framework
 - Web or REST
 - HTTP or Web Sockets
- Kotlin support added April 2017



SPARK EXAMPLE

```
fun main(args: Array<String>) {  
    val userDao = UserDao()  
  
    path("/users") {  
        get("/:id") { req, res ->  
            userDao.findById(req.params("id").toInt())  
        }  
        get("/email/:email") { req, res ->  
            userDao.findByEmail(req.params("email"))  
        }  
        post("/create") { req, res ->  
            userDao.save(name = req.qp("name"), email = req.qp("email"))  
            res.status(201)  
            "ok"  
        }  
    }  
}
```



Spark/Kotlin Documentation:

I have only worked with Kotlin for a few hours while writing this tutorial, but I'm already a very big fan of the language. Everything just seems to make sense, and the interoperability with Java is great.



KTOR

- Lightweight framework
- Functional interface via lambda functions
- Asynchronous by design
- Hosted
 - Servlet 3.0+ (e.g. Tomcat)
- Standalone
 - Netty or Jetty



KTOR EXAMPLE

- Simple JSON REST Service

```
fun Application.main() {  
    install(DefaultHeaders)  
    install(CallLogging)  
    install(GsonSupport) {  
        setPrettyPrinting()  
    }  
    ...  
    routing {  
        get("/v1") {  
            call.respond(model)  
        }  
        get("/v1/item/{key}") {  
            val item = repository.getItem(key)  
            if (item == null)  
                call.respond(HttpStatusCode.NotFound)  
            else  
                call.respond(item)  
        }  
    }  
}
```



SUMMARY

- Kotlin is a viable option for server-side development
- Easy transition from Java
 - Use identical build & deploy process
- Latest frameworks have inbuilt Kotlin support
- Produces elegant & clean code
- Try It



USEFUL LINKS

- Spring Framework/Kotlin
 - <https://spring.io/blog/2017/01/04/introducing-kotlin-support-in-spring-framework-5-0>
 - <https://kotlinlang.org/docs/tutorials/spring-boot-restful.html>
- Spring WebFlux/Kotlin
 - <https://github.com/mixitconf/mixit>
- Spark <http://sparkjava.com/>
- Ktor <http://ktor.io/>