

Presentata da **Moreno Frigo Turco**
Supportata dal team di **Bacaro Tech**



Spring Boot Java vs Kotlin




Introduzione

In questa presentazione esamineremo la struttura di due semplici microservizi Spring Boot: uno sviluppato in Java e l'altro in Kotlin.

Per la generazione dei progetti (come vedremo nelle prossime slide), ho utilizzato Spring Initializr.

Per il codice completo, potete consultare il mio GitHub.

Generazione Progetto Java



Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ **Java** ☐ Kotlin ☐ Groovy
☒ **Maven**

Spring Boot
☐ 3.5.0 (SNAPSHOT) ☐ 3.5.0 (M1) ☐ 3.4.3 (SNAPSHOT) ☒ **3.4.2**
☐ 3.3.9 (SNAPSHOT) ☐ 3.3.8

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ **Jar** ☐ War

Java ☐ 23 ☒ **21** ☐ 17

Dependencies ADD DEPENDENCIES... CTRL + B

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL Driver SQL
MySQL JDBC driver.

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.


Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

GENERATE CTRL + G

EXPLORE CTRL + SPACE

...

Generazione Progetto Kotlin



Project
☐ Gradle - Groovy ☒ Gradle - Kotlin
☐ Maven

Language
☐ Java ☒ Kotlin ☐ Groovy

Spring Boot
☐ 3.5.0 (SNAPSHOT) ☐ 3.5.0 (M2) ☐ 3.4.4 (SNAPSHOT) ☒ 3.4.3
☐ 3.3.10 (SNAPSHOT) ☐ 3.3.9

Project Metadata

Group

myt.dev

Artifact

kotlinbackend

Name

kotlinbackend

Description

Semplice Backend costruito con Springboot + Kotlin

Package name

myt.dev.kotlinbackend

Packaging

☒ Jar ☐ War

Java

☐ 23 ☒ 21 ☐ 17

Dependencies [ADD DEPENDENCIES... CTRL + B](#)

Spring Data JPA **SQL**

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL Driver **SQL**

MySQL JDBC driver.

Spring Web **WEB**

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Boot DevTools **DEVELOPER TOOLS**

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

[GENERATE CTRL + G](#)

[EXPLORE CTRL + SPACE](#)

[...](#)

Model

Il model è il punto di connessione tra il nostro backend e il database.

Al suo interno troviamo le entità, ovvero classi che rispecchiano la struttura delle tabelle del database utilizzato. Queste entità vengono annotate con specifiche configurazioni per definire le relazioni, i vincoli e i tipi di dati, facilitando l'interazione tra l'applicazione e il database in modo strutturato ed efficiente.

Service

Il service rappresenta il livello logico dell'applicazione, gestendo le operazioni e la business logic del backend.

Qui vengono implementati i metodi che manipolano i dati, interagendo con il repository per recuperare, salvare, modificare o eliminare informazioni dal database. Questo livello aiuta a separare la logica applicativa dalla persistenza, garantendo un'architettura più modulare, manutenibile e scalabile.

Controller

Il controller è il livello dell'applicazione che gestisce le richieste HTTP e funge da punto di ingresso per il backend.

Qui vengono definiti gli endpoint che permettono ai client di interagire con il sistema, instradando le richieste verso il service appropriato. Il controller elabora gli input, restituisce le risposte nel formato corretto (ad esempio JSON) e gestisce eventuali errori, garantendo una comunicazione chiara ed efficiente tra frontend e backend.

Conclusioni Finali

Come abbiamo visto nel codice, i due linguaggi sono molto simili, dimostrando che il passaggio da uno all'altro non comporta grandi difficoltà.

Abbiamo anche evidenziato alcuni vantaggi di Kotlin, come la gestione semplificata di alcune casistiche, ad esempio il Null Safety. Tuttavia, utilizzando un framework come Spring Boot, che è basato su Java, non sfruttiamo appieno la potenza di Kotlin.

Quindi, come potremmo valorizzarlo al massimo? La soluzione è utilizzare un framework nativo per Kotlin, come Ktor!

Ringraziamenti

Volevo ringraziare ancora il team del Bacaro Tech per lo spazio concesso a queste interessanti chicche su Kotlin.

Se i contenuti proposti vi hanno suscitato interesse, vi invitiamo a farcelo sapere, così da poter organizzare un altro incontro insieme.

Buon codice a tutti! 🍷

Grazie dell'attenzione!

**Per ulteriori contenuti come questo,
seguimi su**

