BO·TT·EGA
IT minds

Program szkolenia:

Kotlin i Spring Boot – zaawansowane aspekty języka i architektury

Informacje:

Nazwa:	**Kotlin i Spring Boot - zaawansowane aspekty języka i architektury**
Kod:	**kotlin-KotBoot**
Kategoria:	Kotlin
Odbiorcy:	developerzy, architekci
Czas trwania:	4 dni
Forma:	30% wykłady / 70% warsztaty

W trakcie szkolenia uczestnik nauczy się używać konstrukcji składniowych Kotlina oraz pozna idiomy i wzorce ich wykorzystania w kontekście tworzenia aplikacji backendowych z wykorzystaniem frameworka Spring.

Kotlin jest językiem coraz mocniej zyskującym na popularności, głównie za sprawą wygodnych konstrukcji składniowych i wysunięcia na pierwszy plan paradygmatu programowania funkcyjnego. Cechy te znacznie ułatwiają pisanie dobrego kodu zawierającego mniej błędów. Znajdując szerokie zastosowanie na różnych platformach, Kotlin staje się językiem uniwersalnym. Jest oficjalnie wspierany przez Google do tworzenia aplikacji androidowych, z powodzeniem stosowany do tworzenia rozwiązań backendowych, webowych (kotlin-js) oraz natywnych, w tym iOS (kotlin-native).

Zalety szkolenia:

- Architektura aplikacji
- Poprawne wykorzystanie konstrukcji języka, zgodnie z ich przeznaczeniem
- Elementy Domain-driven Design

BOTTEGA IT minds

Szczegółowy program:

1. Kotlin (2 dni)

- 1.1. Składnia języka
  - 1.1.1. Podstawy składni i różnice względem Javy
  - 1.1.2. Deklaracje zmiennych i stałych
  - 1.1.3. Typy podstawowe
  - 1.1.4. Kontrola przepływu
  - 1.1.5. Zwracanie wartości funkcji/lambdy i skoki
  - 1.1.6. Null-safety na poziomie systemu typów
  - 1.1.7. Równość obiektów w Kotlinie
  - 1.1.8. Obsługa wyjątków
- 1.2. Konstrukcje języka do programowania obiektowego
  - 1.2.1. Interface, Class and Object (Singleton)
  - 1.2.2. Companion Objects
  - 1.2.3. Konstruktory i atrybuty
  - 1.2.4. Enkapsulacja (modyfikatory dostępu)
  - 1.2.5. Data Classes i niemodyfikowalny stan
  - 1.2.6. Sealed Classes jako narzędzie do modelowania stanu
  - 1.2.7. Rozszerzanie możliwości klas przez extension functions
  - 1.2.8. Delegacja na poziomie klasy jako alternatywa do dziedziczenia
  - 1.2.9. Delegowane atrybuty
    - 1.2.9.1. Lazy
    - 1.2.9.2. Observable
    - 1.2.9.3. Implementacja własnych delegatów

BOTTEGA IT minds

- 1.2.10. Aliasy typów
- 1.2.11. Nadpisywanie operatorów
- 1.3. Biblioteka standardowa
  - 1.3.1. Zakresy
  - 1.3.2. Tablice i kolekcje
  - 1.3.3. Przydatne rozszerzenia do biblioteki standardowej Javy
- 1.4. Programowanie funkcyjne
  - 1.4.1. Założenia paradygmatu funkcyjnego
  - 1.4.2. Funkcje w Kotlinie
    - 1.4.2.1. Parametry domyślne
    - 1.4.2.2. Nazwane argumenty
    - 1.4.2.3. Deklaracja typów zwracanych
    - 1.4.2.4. Notacja infiksowa
    - 1.4.2.5. Funkcje wyższego rzędu
    - 1.4.2.6. Wyrażenia lambda
- 1.5. Optymalizacja funkcji wyższego rzędu przez inlinowanie wykonania
  - 1.5.1. Funkcyjne przetwarzanie kolekcji
  - 1.5.2. Programowanie bez zmiennych - funkcje let/use/with/apply
- 1.6. Programowanie współbieżne z wykorzystaniem Coroutines
- 1.7. Generyczne klasy i funkcje
  - 1.7.1. Składnia i różnice względem Javy
  - 1.7.2. Zapobieganie wymazywaniu typów dzięki reifikowanym typom generycznym
- 1.8. Współpraca Kotlina z Javą
- 1.9. Kotlin i jego wpływ na performance aplikacji
  - 1.9.1. Porównanie bytecode'u Javy i Kotlina dla podstawowych konstrukcji języka

BOTTEGA IT minds

2. SpringBoot (2 dni)

- 2.1. Paradygmat Inversion of Control - wsparcie frameworka
  - 2.1.1. Dependency Injection
    - 2.1.1.1. Wykorzystanie do zmniejszenia poziomu zależności
  - 2.1.2. Zdarzenia
    - 2.1.2.1. Zdarzeniowe architektury otwarte na rozbudowę
    - 2.1.2.2. Praktyczne przykłady generowania zdarzeń biznesowych z warstwy logiki
  - 2.1.3. Techniki Aspect Oriented Programming
    - 2.1.3.1. Podstawy teoretyczne
    - 2.1.3.2. Praktyczne przykłady wykorzystania w aplikacjach biznesowych
    - 2.1.3.3. Zastosowanie w Spring
- 2.2. Kontener wstrzykiwania zależności
  - 2.2.1. Konfiguracja
  - 2.2.2. Komponenty
    - 2.2.2.1. Deklaracja
    - 2.2.2.2. Cykl życia
    - 2.2.2.3. Zależności
    - 2.2.2.4. Zasięg komponentów
      - 2.2.2.4.1. Scoped Proxy
      - 2.2.2.4.2. Wstrzykiwanie krócej żyjących komponentów do dłużej żyjących
      - 2.2.2.4.3. Pre/post procesory
- 2.3. MVC
  - 2.3.1. Wzorzec Model View Controller w kontekście Springa
  - 2.3.2. Kontrolery

BOTTEGA IT minds

- 2.3.2.1. Mapowania restowe
- 2.3.2.2. Obsługa parametrów
- 2.3.2.3. Obsługa różnych typów zawartości (JSON, etc.) inegocjowanie
- 2.3.3. Model
- 2.3.3.1. Mapowanie modelu
- 2.3.3.2. Walidacja
- 2.3.4. Praca z widokiem i view resolvery
- 2.3.5. Obsługa wyjątków
- 2.3.6. Obsługa wywołań asynchronicznych
- 2.4. Dostęp do danych
- 2.4.1. Transakcje (konfiguracja, poziomy izolacji, warstwa abstrakcji Spring)
- 2.4.2. Integracja z JPA/Hibernate - podstawy
- 2.4.3. Spring Data - automatyzacja d ostępu do danych
- 2.5. Testowanie
- 2.5.1. Spring - wsparcie dla testów
- 2.5.2. Testowanie jednostkowe
- 2.5.3. Testowanie integracyjne ze wsparciem kontenera
- 2.5.4. Slice testing
- 2.5.5. Testy E2E i wykonywalne specyfikacje
- 2.6. Techniki umożliwiające implementację logiki biznesowej w oderwaniu od frameworka
- 2.6.1. Podstawy DDD
- 2.6.2. Podstawy architektury Ports and Adapters
- 2.7. SpringBoot i Kotlin - pułapki i dobre praktyki