BOTTEGA IT minds

Program szkolenia:

Java Persistence API - zagadnienia zaawansowane

Informacje:

Nazwa:	**Java Persistence API - zagadnienia zaawansowane**
Kod:	**jpa-pro**
Kategoria:	Zaawansowana Java
Odbiorcy:	developerzy
Czas trwania:	2 dni
Forma:	50% wykłady / 50% warsztaty

Wydajność jest częstym problemem systemów opartych o ORM dlatego program szkolenia dobrano pod kątem optymalizacji wykorzystania JPA.

Zalety szkolenia:

- Zwracamy szczególną uwagę na wydajność
- Prezentujemy typowe jak i mniej typowe pułapki
- Wskazujemy rozwiązania każdego z omawianych problemów
- warsztaty w formie pracy nad projektem wymagającym usprawnień

BOTTEGA IT minds

Szczegółowy program:

1. Modelowanie encji z wykorzystaniem technik Obiektowych

1.1. Elementy Domain Driven Design

1.2. Mapowanie powiązań, sterowania kierunkiem właściciela

1.3. Mapowania dziedziczenia - klasy bazowe dla Agregatów

1.4. Wprowadzania do modelu ValueObjects zapewniających immutability i zwiększających siłę wyrazu modelu (adnotacja Embeded)

1.5. Enkapsulaja modelu

1.6. Określanie wyraźnej granicy grafu obiektu - określanie jednostki zmiany

2. Architektura dostępu do danych w aplikacjach warstwowych

2.1. Abstrakcja źródeł danych

2.2. Wzorce DAO i Repository

2.3. Wpływ na testability

2.4. Wpływ na przenośność i skalowalność systemu

2.5. Base DAO/Repository

2.5.1. Wspólne operacje na encjach

2.6. Umieszczenie dostępu do danych w architekturze aplikacji

2.7. Projekt architektury pod kątem testowalności

2.8. Architektura CqRS

3. Zaawansowanie mapowania

3.1. Wykorzystanie typów zagnieżdżonych do mapowania Value Objects

3.2. Modelowanie niezmienników

3.3. Dobór struktur danych w powiązaniu One-to-Many

3.3.1. Set

3.3.2. Bag

BOTTEGA IT minds

3.3.3. List

3.3.4. Uniaknie zbędnych operacji odtwarzania i pobierania kolekcji

3.4. Mapowanie kolekcji

3.4.1. Tabele linkujące w powiązaniu One-to-Many

3.4.1.1. Kiedy stosować - kompozycja a agregacja

3.4.2. Strategie pobieranie w kontekście modelu biznesowego

3.5. Optymalizacja operacji

3.5.1. Pobieranie Proxy

3.5.2. Dynamiczny Update

3.6. Operacje kaskadowe a granica Agregatu

3.7. Problem identyfikacje encji - dobór podejścia

3.7.1. klucze główne

3.7.2. klucze naturalne

3.7.3. UUID

3.7.4. referencja w pamięci

3.8. Dziedziczenie

3.8.1. Kiedy dziedziczenie nie ma sensu w modelu biznesowym

3.8.2. Implementacje dziedziczenia - wady i zalety każdego z podejść

3.8.3. Pojedyncza tabela

3.8.4. Tabela per klasa

3.8.5. Tabele dla super klas

3.8.6. Alternatywa dla dziedziczenia

3.8.6.1. Party Archetype

3.8.6.2. Role Object Pattern

4. Zdarzenia i callbacks

BOTTEGA IT minds

4.1. Dobór techniki do problemu

4.2. Typowe zastosowania

5. Wydajność dostępu do danych

5.1. Pułapki wydajności JPA

5.1.1. „n+1 Select problem” - wykrywanie i zapobieganie

5.1.2. Wykrywanie N+1SP

5.1.2.1. Wykrywanie automatyczne przy pomocy AOP

5.1.3. Sposoby naprawy N+1SP

5.1.3.1. JOIN FETCH

5.1.3.2. Batch size

5.2. Pułapki Lazy Loadingu oraz zbyt chciwego pobierania danych

5.2.1. Nadmierne pobieranie danych

5.2.2. Racjonalne wykorzystanie Lazy Loadingu

5.3. Optymalne mapowanie encji

5.3.1. Nadmiarowość pobierania danych

5.3.2. Problemy z leniwym ładowaniem pól

5.4. Pobieranie konkretnych atrybutów

6. Operacje typu batch

6.1. Pułapki

6.1.1. L1 Cache

6.1.2. L2 Cache

6.1.3. Wersjonowanie

7. JPA czy native SQL – dobór odpowiedniego narzędzia do konkretnego problemu

7.1. Podejście pragmatyczne: refaktoryzacja z JPA na SQL przy pomocy DAO i wstrzykiwania zależności

BOTTEGA IT minds

7.2. Dostęp do danych w kontekście architektury Command-query Responsibility Segregation

7.2.1. Dedykowany model do odczytu - pobieranie danych odpowiednich do prezentacji

7.2.2. Uaktualnianie modelu do odczytu

7.3. Wykorzystanie myBatis

8. Hibernate cache – niezastąpione rozwiązanie.

8.1. Idea działania i konfiguracja

8.2. Cache pierwszego poziomu

8.2.1. Wygaszania w przypadku operowania na dużych ilościach danych

8.3. Cache drugiego poziomu

8.3.1. Najlepsze praktyki

8.4. Cache zapytań - pułapki

8.4.1. Kiedy nie stosować

8.5. Mapowanie encji zorientowanie na cacheowanie

9. Entity Manager - tryb rozszerzony

9.1. Naturalna granica jednostki pracy

9.2. Transakcje aplikacyjne

9.3. Model konwersacji

9.4. Manualne opróżnianie kontekstu

10. Testowanie

10.1. Racjonalizacja architektury w kontekście piramidy testów

10.2. Testowanie operacji na JPA

10.3. Unikanie mockowania źródeł danych w testach jednostkowych

10.4. Preferowanie testów end-to-end gdy pobieramy dane

10.5. Zwiększanie pokrycia poprzez testy jednostkowe

BOTTEGA IT minds

11. Transakcje w Java EE

11.1. Anomalie przy współbieżnym dostępie do danych

11.2. Poziomy separacji w celu unikania anomalii

11.3. Deklaratywne (zarządzane przez kontener)

11.4. Koncepcja transakcji sterowanych wyjątkami

11.5. Zarządzane przez Bean

11.6. Zarządzane przez klienta – praktyczne wykorzystanie w przypadku łączenia z operacjami nie tylko bazodanowymi

11.7. Zagadnienie propagacji transakcji - Praktyczne przykłady na każdy z typów

11.8. Transakcje rozproszone w środowisku JEE

- 11.8.1. Konfiguracja
- 11.8.2. Zasada działania „double commit”

11.9. Wydajność transakcji

12. Blokowanie

12.1. Optymistyczne - dobór podejścia do scenariusza

- 12.1.1. Wersjonowanie
- 12.1.2. Read
- 12.1.3. Write - ochrona agregatów (grafów encji)

12.2. Pesymistyczne

12.3. Pułapki i najlepsze praktyki

13. Systemu klasy multi-tenant

13.1. Podejścia do modelu danych

- 13.1.1. Wspólne dane
- 13.1.2. Dedykowane tabele
- 13.1.3. Dedykowane Schemy

13.2. Podejścia do logiki - przegląd