

# Program szkolenia:

## Java Persistence API (Hibernate)

### Informacje ogólne

<b>Nazwa:</b>	<b>Java Persistence API (Hibernate)</b>
<b>Kod:</b>	<b>JPA</b>
<b>Kategoria:</b>	Java Enterprise Edition
<b>Grupa docelowa:</b>	Projektanci, programiści
<b>Czas trwania:</b>	2-3 dni
<b>Forma:</b>	50% wykłady / 50% warsztaty

Szkolenie przedstawia rzetelne podstawy JPA, standardowe problemy i sposoby ich efektywnego rozwiązania oraz zagadnienia zaawansowane, takie jak aspekty wydajności.

Szkolenie może zostać dopasowane do kompleksowego rozwiązania osadzonego w JEE lub opartego o lekkie podejście POJOs w Spring.

Szkolenie przygotowuje uczestników do integracji mechanizmu persystencji w dowolnej architekturze aplikacji webowych i standalone.

Podczas szkolenia zwracamy szczególną uwagę na aspekty optymalizacji, zarówno na poziomie narzędzia jak i architektury systemu. Jak wynika z naszego doświadczenia nawet zaawansowani użytkownicy wpadają w pułapki wydajnościowe.

Materiał został poszerzony o specyficzne zagadnienia Hibernate – najpopularniejszej implementacji JPA.

### Zalety szkolenia:

- » Poruszamy zagadnienia zaawansowane
- » Przedstawiamy alternatywne sposoby korzystania z JPA
- » Zwracamy szczególną uwagę na wydajność

## Program szkolenia:

### 1. Podstawy modelowania obiektowego i mapowania relacyjno-obiektowego

### 2. Konfiguracja Hibernate

2.1. W środowisku Java EE

2.2. W środowisku niezarządzanym

2.3. Spring lub Seam

### 3. Mapowanie encji

3.1. Najlepsze praktyki

3.2. Typy, klucze proste i ograniczenia

3.3. Powiązania

3.4. Strategie pobierania zagregowanych danych – praktyczne rady

3.5. Strategie wykonywania operacji kaskadowych – praktyczne rady

### 4. Mapowanie zaawansowane

4.1. Zagnieżdżenia

4.2. Efektywne mapowanie Value Objects jako klas Embedable

4.3. Klucze złożone

4.4. Wykorzystanie map do realizacji persystencji

4.5. Kolekcje typów prostych

### 5. EntityManager i kontekst persystencji (Persistent Context)

5.1. Cykl życia

5.2. API EntityManager

5.3. Tryb rozszerzony

5.4. Imperatywne zarządzanie synchronizacją kontekstu persystencji

### 6. Tworzenie efektywnych zapytań

## 6.1. Metody tworzenia prostych zapytań

### 6.1.1. Język JPQL

### 6.1.2. Criteria API w JPA 2.0

## 6.2. Zaawansowane wymagania biznesowe za pomocą dynamicznego składania zapytań

## 6.3. Techniki pisania wydajnych zapytań

### 6.3.1. Ograniczenie pobierania nie potrzebnych danych

### 6.3.2. Ograniczenie ilości zapytań do bazy

### 6.3.3. Prekompilowanie zapytań JPQL – nazwane zapytania

## 7. Odzworowanie dziedziczenia

### 7.1. Trzy strategie (wady i zalety)

### 7.2. Dobór strategii do problemu

### 7.3. Zapytania polimorficzne

## 8. Transakcyjność operacji JPA

### 8.1. Problem transakcyjności operacji JPA

### 8.2. Optymistyczne i pesymistyczne blokowanie

### 8.3. Tryb rozszerzony EntityManager

## 9. Optymalizacja

### 9.1. Techniki optymalnego mapowania

### 9.2. Cache (zapytań, encji)

### 9.3. Pułapki lazy loadingu (n+1 select problem)

### 9.4. Unikanie pobierania nadmiernych danych – rozwiązania stosowane do problemu

## 10. Architektury warstwy dostępu do danych

### 10.1. Podejście płaskie

### 10.2. Warstwa Data Access Objects (DAO)

### 10.3. Repozytoria Domain Driven Design (DDD)

10.4. Systemy rozproszone

## 11. Zagadnienia zaawansowane

11.1. Mechanizmy JPA - przykłady praktycznego wykorzystania

11.1.1. Wywołania zwrotne

11.1.2. Klasy nasłuchujące

11.1.3. Praktyczne wykorzystanie mechanizmów

11.2. Zasada działania JPA

11.3. Szczegóły implementacji Hibernate – zwiększenie świadomości używanego narzędzia

11.4. Hibernate Shreads

11.5. Hibernate Search - Lucene

11.6. Wsparcie dla konwersacji - Tryb rozszerzony

11.7. Przydatne rozszerzenia Hibernate

11.7.1. Dostęp do Hibernate Session

11.7.2. Criteria API w Hibernate – produktywnie tworzenie dynamicznych zapytań

11.7.3. Dodatkowe generatory kluczy

11.7.4. Dodatkowe operacje kaskadowe