

# Java Persistence API

## Platformy Technologiczne

Michał Piotrowski  
Michał Wójcik

Katedra Architektury Systemów Komputerowych  
Wydział Elektroniki, Telekomunikacji i Informatyki  
Politechnika Gdańska

9 marca 2011

## JPA — Java Persistence API:

- mechanizm pozwalający na mapowanie obiektów Javy na tabele w relacyjnej bazie danych,
- wykorzystywany w:
  - Java SE,
  - Java EE (uproszczone używanie i konfigurowanie);
- nie wymaga tworzenia skomplikowanych DAO (Data Access Objects),
- wspomaga transakcje,
- niezależny od dostawcy bazy danych,
- możliwość uniknięcia zabawy z SQL.

Klasy encyjne (entity classes) — klasy przechowywane w bazie danych:

- zwykłe klasy POJO (Plain Old Java Object),
- pola klasy nie mogą być publiczne,
- każdy obiekt encyjny ma jednoznacznie identyfikujący go klucz główny
  - złożone klucze główne są reprezentowane przez oddzielne klasy, które muszą implementować metody `hashCode()` i `equals()`,
- oznaczone za pomocą anotacji.

## Wybrane adnotacje w JPA:

- `@Entity` — oznaczenie klasy encyjnej,
- `@Table` — oznaczenie tabeli:
  - `name` — nazwa tabeli w bazie danych;
- `@Column` — oznaczenie kolumny:
  - `name` — nazwa kolumny w bazie danych,
  - `nullable` — ustawienie czy pole jest wymagane;
- `@JoinColumn` — mapowanie kolumn przy łączeniu encji:
  - `name` — nazwa kolumny z kluczem zapożyczonym,
  - `referencedColumnName` — nazwa kolumny na którą wskazuje klucz,
- `@Id` — klucz główny,
- `@GeneratedValue` — autogenerowalny klucz,
- `@Transient` — nie podlega utrwaleniu,
- `@Temporal` — wymagane dla typów `Date` i `Calendar`,

Adnotacje powiązań pomiędzy encjami:

- @OneToOne:
  - cascade — usuwanie elementów zależnych od siebie,
  - mappedBy — ustala ustala pole lub własność będące właścicielem relacji;
- @OneToMany:
  - cascade,
  - mappedBy;
- @ManyToOne:
  - cascade;
- @ManyToMany:
  - cascade,
  - mappedBy.

Rodzaje powiązań:

- jednokierunkowe — tylko jedna encja ma pole wskazujące na drugą,
- dwukierunkowe — obie encje mają pola wskazujące na siebie nawzajem.

Persistence Unit — zbiór encji w aplikacji:

- `persistence.xml` — plik konfiguracyjny, w katalogu `META-INF`,
- wybrane elementy:
  - `persistence-unit` — opis konkretnego PU:
    - `name` — nazwa,
    - `transaction-type` — rodzaj transakcji: `RESOURCE-LOCAL`, `JTA`;
  - `provider` — dostawca implementacji,
  - `class` — nazwa klasy encyjnej wraz z pakietem,
  - `property` — konfiguracja połączenia z bazą (zwłaszcza `JAVA SE`).

Entity Manager — dostarcza metod operowania na transakcjach i encjach:

- persistence context — grupa unikalnych encji zarządzanych podczas działania aplikacji,
- persistent identity — posiada unikatowy identyfikator,
- stany encji:
  - new — nie jest jeszcze związana z persistence context, nie posiada persistent identity,
  - managed — związana z persistence context, posiada persistent identity,
  - detached — nie jest już związana z persistence context, posiada persistent identity,
  - removed — związana z persistence context, posiada persistent identity, oznaczona do usunięcia;
- EntityManagerFactory — udostępnia dostęp do zarządcy encji (w Java SE),
- nie może być wykorzystywany przez więcej niż jeden wątek.

## Zarządzanie encjami:

- **EntityManager:**

- `void persist(Object o)` — dodanie nowej encji (new -> managed),
- `EntityTransaction getTransaction()` — zwraca obiekt transakcji,
- `<T> T merge(T entity)` — synchronizuje stan encji z bazą,
- `void remove(Object o)` — usunięcie encji (managed -> removed),
- `void refresh(Object o)` — aktualizuje stan encji na podstawie bazy,
- `<T> T find(Class<T> entityClass, Object key)` — wyszukuje na podstawie klucza prywatnego;

- **EntityTransaction:**

- `begin()` — rozpoczyna transakcję,
- `commit()` — kończy transakcję, zapisuje zmiany do bazy,
- `rollback()` — usuwa transakcję z kolejki, odłącza zarządzane encje.

Java DB — wspierana przez SUN (Oracle) dystrybucja Apache Derby.  
Apache Derby:

- mała baza danych, około 2.6 MB,
- w całości zaimplementowana w języku Java,
- dystrybuowana na licencji Apache License, Version 2.0,
- oparta na standardach JDBC i SQL,
- wspiera także tryb klient/serwer.

- [1] *Apache Derby Tutorial*.  
<http://db.apache.org/derby/papers/DerbyTut/index.html>.
- [2] John O'Conner.  
Using the Java Persistence AP in Desktop Applications.  
<http://java.sun.com/developer/technicalArticles/J2SE/Desktop/persistenceapi/>.
- [3] Oracle.  
*The Java EE 6 Tutorial*, chapter VI Persistence.  
Oracle, March 2011.  
<http://download.oracle.com/javasee/6/tutorial/doc/>.