CEL SZKOLENIA:

This course offers a comprehensive and detail-oriented treatment of the Java Persistence API (JPA) for developers interested in implementing persistence tiers for enterprise applications. We cover JPA basics including simple object/relational concepts and annotations, persistence contexts and entity managers, and configuration via persistence.xml. We get a good grounding in the Java Persistence Query Language (JPQL), working with a JPQL console. The course then moves into advanced mapping techniques, the Criteria API, lifecycle hooks, validation, locking, and caching. Students will complete the course with a firm understanding of JPA architecture and plenty of hands-on experience.

This version of the course supports JPA 2.1 with a choice of two providers: EclipseLink 2.6, which is pre-configured for course exercises, and Hibernate® 5.0. Switching providers is just a matter of moving a few lines in and out of XML comments in the relevant persistence.xml file, and we encourage instructors to demonstrate both providers, to illustrate portability and for comparison of some finer points.

The course also supports either the Derby or Oracle® RDBMS. Derby is bundled with the course software and is pre-configured; a script is included to change over to Oracle configurations for all exercises and schema-creation scripts are available for both.

Hibernate is a registered trademark of Red Hat, Inc. Oracle is a registered trademark of Oracle Corporation. Other names may be trademarks of their respective owners. No association with or endorsement by Red Hat or Oracle is implied by the use of these terms in this document.

Learning Objectives

- Understand the value of object/relational mapping and JPA's role as a standard for ORM
implementations.
- Develop JPA entities using JPA annotations to align the Java classes, properties, and types to relational tables, columns, and types.
- Create entity managers and instantiate persistence contexts to perform create/retrieve/update/delete (CRUD) operations.
- Implement entity relationships of all cardinalities, including unidirectional and bidirectional relationships.
- Map composite primary keys, inheritance relationships, eager/lazy fetching, and cascading operations.
- Use JPQL to write object-oriented queries, and process query results.
- Use the Criteria API to define queries programmatically, and take advantage of type safety using the Metamodel API.
- Build reusable façades that encapsulate simpler and more complex persistence operations.
- Implement persistence lifecycle event handlers.
- Define JSR-303 validation constraints on JPA entities and see them enforced by the JPA provider.
- Manage concurrent operations via optimistic or pessimistic locking strategies.
- Understand the actions of the local and shared entity caches, and use them appropriately while avoiding over-caching pitfalls.

Plan szkolenia:
- Introduction to JPA
  - Object/Relational Mapping
  - Mismatches Between Relational and Object Models
  - The Java Persistence API
  - JPA Architecture
  - Entity Metadata
  - The Entity Manager
  - JPA Providers
- Single-Table Mapping
  - Annotations
  - JavaBean Standards
  - Property, Field, and Mixed Access
  - Table and Column Mapping
  - Primary Keys and Key Generation
  - Type Mappings
  - Temporal and Enumerated Types
  - Embedded Types
Converters

Mapping Associations
  @Embeddable Types
  Entity Relationships
  @ManyToOne Relationships
  @OneToMany Relationships
  @ManyToMany Relationships
  Eager and Lazy Loading

Entity Managers
  Putting Entities to Work
  persistence.xml
  Entity State and Transitions
  Managing Transactions
  Persistence Operations
  Creating Queries
  Named Queries
  Query Parameters
  Native Queries
  Stored-Procedure Queries

JPQL
  The Java Persistence Query Language
  Query Structure
  Path Expressions
  Filtering
  Scalar Functions
  Using Native Functions
  Operators and Precedence
  between, like, in
  is null, is empty
  Ordering
  Aliases
  Grouping
  Aggregate Functions
  Joins
  Fetch Joins
• Constructors
• Updates and Deletes

• Persistence Components
  • Encapsulating Persistence Logic
  • Design Considerations
  • Testability
  • Transaction Control
  • Exception Handling
  • Generic Types

• Advanced Mappings
  • Inheritance Strategies
  • Single-Table Strategy
  • Joined-Table Strategy
  • Table-Per-Concrete-Class Strategy
  • Querying Over Inheritance Relationships
  • Secondary Tables
  • Composite Primary Keys
  • @IdClass and @Embeddeld
  • Derived Identifiers
  • @ElementCollection
  • Default Values
  • @Version Fields
  • Cascading and Orphan Removal
  • Detachment and Merging

• The Criteria API
  • History of the Criteria API
  • Criteria Query Structure
  • The MetaModel API and Query Type Safety
  • Tuples
  • Joins
  • Predicates
  • Building Expressions
  • Ordering
  • Grouping
  • Encapsulating Persistence Logic
  • Façades
○ Range Queries
○ Updates and Deletes

○ Lifecycle
  ○ Lifecycle Events
  ○ Method Annotations
  ○ Entity Listeners

○ Validation
  ○ JSR-303 Validation
  ○ Constraint Annotations
  ○ Validation Modes
  ○ Validation Groups
  ○ Handling Validation Exceptions

○ Locking
  ○ Concurrency
  ○ Updates to the Same and Related Entities
  ○ Pessimistic Locking
  ○ Lock Types
  ○ Who Blocks Whom
  ○ Deadlocking and Timeouts
  ○ Optimistic Locking
  ○ The @Version Annotation
  ○ Optimistic Read and Write Locking
  ○ Error Handling
  ○ Combining Locking Strategies

○ Caching
  ○ Caching
  ○ Persistence Context as Transactional Cache
  ○ Shared (2nd-Level) Cache
  ○ Pros and Cons
  ○ Cache Configuration
  ○ Eviction
  ○ Effects of Locking on Caching

Wymagania:

○ A strong Java programming background is essential for this course -- consider Course 103
Knowledge of relational database concepts and SQL is recommended -- consider Course 301 Introduction to SQL -- but is not strictly required.

Prior experience with JDBC will be a plus but is not required.

Poziom trudności

Certyfikaty:

The participants will obtain certificates signed by Capstone Courseware.

Prowadzący:

Authorized Capstone Courseware Trainer.