# The Java Persistence API Version 2.0

# Course Summary

**Description**

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) and take advantage of a prepared JPQL query console to explore the two schemas on which the course's case studies are based. The course then moves into advanced topics including JPA-2.0 mapping options, the Criteria API, lifecycle hooks, JSR-303 validation, locking, and caching. Students will complete the course with a firm understanding of JPA architecture and plenty of hands-on exercise in entity mapping, persistence operations, and JPQL.

Course software includes two schemas: a fairly simple human-resources model (6 tables, 253 rows) for early chapters and a more sophisticated pharmacy schema (14 tables, 4255 rows) for the latter half of the course. The pharmacy schema is based on an open-source project and made available to Capstone Courseware by special permission.

This version of the course supports JPA 2.0 with a choice of two providers: EclipseLink 2.3, which is pre-configured for course exercises, and Hibernate® 4.0. 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 at least here and there, to illustrate portability and for comparison's sake over 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.

**Objectives**
**At the end of this course, students will be able to:**

- 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 persistence operations.
- Carry out create/retrieve/update/delete (CRUD) operations on JPA entities using entity managers.
- Implement entity relationships of all cardinalities, including unidirectional and bidirectional relationships.
- Implement special ORM cases such as composite primary keys, inheritance relationships, 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.
- Make well-informed decisions about locking strategies, and understand the role of the JPA cache in enterprise applications.

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

# The Java Persistence API Version 2.0

## Course Summary (cont'd)

### Topics

- Introduction to JPA
- Object/Relational Mapping
- Entity Managers
- JPQL
- Advanced Mappings
- The Criteria API
- Lifecycle and Validation
- Locking and Caching

### Audience

This course is designed for developers interested in implementing persistence tiers for enterprise applications.

### Prerequisites

A strong Java programming background is essential for this course. Knowledge of relational database concepts and SQL is recommended but is not strictly required. Prior experience with JDBC will be a plus but is not required.

### Duration

Four days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

# The Java Persistence API Version 2.0

## Course Outline

**I. Introduction to JPA**
- A. Object/Relational Mapping
- B. Mismatches Between Relational and Object Models
- C. The Java Persistence API
- D. JPA History
- E. JPA Architecture
- F. Entity Metadata
- G. The Entity Manager
- H. JPA Providers

**II. Object/Relational Mapping**
- A. Annotations
- B. JavaBean Standards
- C. Property, Field, and Mixed Access
- D. Table and Column Mapping
- E. Primary Keys and Generation
- F. Type Mappings
- G. Temporal and Enumerated Types
- H. Embedded Types
- I. Entity Relationships
- J. @ManyToOne Relationships
- K. @OneToOne Relationships
- L. @OneToMany Relationships
- M. @ManyToMany Relationships
- N. Eager and Lazy Loading

**III. Entity Managers**
- A. Putting Entities to Work
- B. persistence.xml
- C. Entity State and Transitions
- D. Managing Transactions
- E. Persistence Operations
- F. Creating Queries
- G. Named Queries
- H. Query Parameters
- I. Native Queries

**IV. JPQL**
- A. The Java Persistence Query Language
- B. Query Structure
- C. Path Expressions
- D. Filtering
- E. Scalar Functions
- F. Operators and Precedence
- G. between, like, in
- H. is null, is empty
- I. Ordering
- J. Aliases
- K. Grouping
- L. Aggregate Functions
- M. Joins
- N. Constructors

**V. Advanced Mappings**
- A. Inheritance Strategies
- B. Single-Table Strategy
- C. Joined-Table Strategy
- D. Table-Per-Concrete-Class Strategy
- E. Querying Over Inheritance Relationships
- F. Secondary Tables
- G. Composite Primary Keys
- H. @IdClass and @EmbeddedId
- I. Derived Identifiers
- J. @ElementCollection
- K. Default Values
- L. @Version Fields
- M. Cascading and Orphan Removal
- N. Detachment and Merging

**VI. The Criteria API**
- A. History of the Criteria API
- B. Criteria Query Structure
- C. The MetaModel API and Query Type Safety
- D. Tuples
- E. Joins
- F. Predicates
- G. Building Expressions
- H. Ordering
- I. Grouping
- J. Encapsulating Persistence Logic
- K. Façades
- L. Range Queries

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

# The Java Persistence API Version 2.0

## Course Outline (cont'd)

**VII. Lifecycle and Validation**

A. Lifecycle Events
B. Method Annotations
C. Entity Listeners
D. JSR-303 Validation
E. Constraint Annotations
F. Validation Modes
G. Validation Groups

**VIII. Locking and Caching**

A. Concurrency
B. Optimistic Locking
C. Optimistic Read Locking
D. Optimistic Write Locking
E. Pessimistic Locking
F. Caching
G. Persistence Context as Transactional Cache
H. Shared (2nd-level) Cache
I. Locking and Caching "Do's and Don'ts"

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically