Introduction to Hibernate

Originals of Slides and Source Code for Examples:
http://courses.coreservlets.com/Course-Materials/hibernate.html

Customized Java EE Training: http://courses.coreservlets.com/
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live Spring & Hibernate training, see courses at http://courses.coreservlets.com/.

Taught by the experts that brought you this tutorial.
Available at public venues, or customized versions can be held on-site at your organization.

- Courses developed and taught by Marty Hall
  - Java 5, Java 6, intermediate/beginning servlets/JSP, advanced servlets/JSP, Struts, JSF, Ajax, GWT, custom mix of topics
- Courses developed and taught by coreservlets.com experts (edited by Marty)
  - Spring, Hibernate/JPA, EJB3, Ruby/Rails

Contact hall@coreservlets.com for details
Topics in this Section

- Refresher in enterprise application architectures
- Traditional persistence
- Hibernate motivation
- Installation

Enterprise Application Architectures

Customized Java EE Training: http://courses.coreservlets.com/
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.
**N-Tier Architecture**

- **Application is made up of layers or tiers**
  - Each layer encapsulates specific responsibilities
  - Enables changes in one area with minimal impact to other areas of the application

- **Common tiers**
  - Presentation
    - ‘View’ in model-view-controller
    - Responsible for displaying data only. No business logic
  - Service
    - Responsible for business logic
  - Persistence
    - Responsible for storing/retrieving data
**DAO Design Pattern**

- **Data Access Object**
  - Abstracts CRUD (Create, Retrieve, Update, Delete) operations
- **Benefits**
  - Allows different storage implementations to be ‘plugged in’ with minimal impact to the rest of the system
  - Decouples persistence layer
  - Encourages and supports code reuse

**Implementing Business Logic**

- **Service Layer**
  - Thin domain layer
  - Procedural service layer
  - Fowler ‘Anemic Domain Model’
- **Domain Objects/Business Objects**
  - Thin service layer and complex OO domain model
  - Business logic primarily in the domain/business objects
  - Rich domain objects
- **Some combination of the two…**
Design Approaches

- [D1] Service layer contains all business logic (no real domain model)
- [D2] Complex OO domain model/thin service layer
- [D3] Service layer contains use case logic that operates over thin or moderately complex domain model

[D1] Procedural Approach

- Service layer communicates directly to data access layer
  - No object model
  - Data access layer returns data transfer objects (DTOs) to service layer
- Leverages commonly understood core technologies
  - JDBC, JavaBeans
- Requires more low level code to persist transfer objects to the data store
[D1] Procedural Approach

[Diagram showing layers of presentation, service, and persistence]

• Complex OO domain model/thin service layer
  – Rich object model utilizing standard design patterns, delegation, inheritance, etc.
  – Distinct API to domain model

• May result in more maintainable code but updates are harder
  – What objects have been modified and need to be saved in the database

• Need complex Data Mapper/Data Store since domain model and database schema are likely different
  – TopLink, JDO, Hibernate

[D2] Object Oriented Approach
[D2] Object Oriented Approach

- Presentation
- Service
- Persistence
- Database

[D3] Mixed Approach

- **Object model can be basic to moderately complex**
  - Simple model is just used as a data access/ORM layer
  - Model can take on business logic
    - Common behavior for different service-layer use cases
    - Service layer performs use-case operations over a set of cooperating business objects
  - Example: Entity Beans BMP/CMP
- **Uses advantages of both extremes**
- **Difficult to remain consistent within the same application**
Traditional Persistence

Customized Java EE Training: http://courses.coreservlets.com/
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.
JDBC Overview

- **JDBC API provides ability to**
  - Establish connection to a database
  - Execute SQL statements
  - Create parameterized queries
  - Iterate through results
  - Manage database transactions

- **Basic Steps to JDBC Operations**
  1. Load driver or obtain datasource
  2. Establish connection using a JDBC URL
  3. Create statement
  4. Execute statement
  5. Optionally, process results in result set
  6. Close database resources
  7. Optionally, commit/rollback transaction

**JDBC Example – Create Account**

```java
public Account createAccount(Account account) {
    Connection connection = null;
    PreparedStatement getAccountIdStatement = null;
    PreparedStatement createAccountStatement = null;
    ResultSet resultSet = null;
    long accountId=0;

    // Load driver
    try {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        catch (Exception e) {
            throw new RuntimeException(e);
        }
    } catch (Exception e) {

    //Get connection and set auto commit to false
    try {
        Connection connection =
            DriverManager.getConnection("jdbc:oracle:
            thin:lecture1/lecture1@localhost:1521:XE");
        connection.setAutoCommit(false);
    }

    return new Account(accountId);
```

...
// Get account id from sequence
getAccountIdStatement = connection.prepareStatement("SELECT ACCOUNT_ID_SEQ.NEXTVAL FROM DUAL");
resultSet = getAccountIdStatement.executeQuery();
resultSet.next();
accountId = resultSet.getLong(1);

// Create the account
createAccountStatement = connection.prepareStatement(AccountDAOConstants.CREATE_ACCOUNT);
createAccountStatement.setLong(1, accountId);
createAccountStatement.setString(2, account.getAccountType());
cREATEACCOUNTSTATEMENT.setDouble(3, account.getBalance());
createAccountStatement.executeUpdate();

// Commit transaction
connection.commit();

catch (SQLException e) {
    // In case of exception, rollback
    try{
        connection.rollback();
    }catch(SQLException e1){// log error
    }catch(SQLException e1){// log error
    throw new RuntimeException(e);
}
finally {
    // Close database resources
    try {
        if (resultSet != null)
            resultSet.close();
        if (getAccountIdStatement!= null)
            getAccountIdStatement.close();
        if (createAccountStatement!= null)
            createAccountStatement.close();
        if (connection != null)
            connection.close();
    } catch (SQLException e) {// log error
    }
}
EJB 2.x Overview

- **EJB API provides ability to**
  - Map object model to database tables
  - Hand off management of database connections
  - Take care of relationship management
  - Manage transactions
  - Use callback methods.
  - Search for desired objects
  - Access Control

- **Basic Steps to EJB Operations**
  1. Create your EJB
     - Home Interface
     - Remote Interface
     - Bean Class (implementation class)
  2. Setup deployment descriptors
     - ejb-jar.xml
     - Container specific EJB descriptor (<container>-ejb-jar.xml)
  3. In code, look up the EJB Home Interface
  4. Create an instance of the EJB off the Home Interface, using attributes passed in through the method call

EJB 2.x Home Interface

```java
public interface SavingsAccountHome extends EJBHome {

    public SavingsAccount create(String id, String firstName, String lastName, BigDecimal balance)
    throws RemoteException, CreateException;

    public SavingsAccount findByPrimaryKey(String id)
    throws FinderException, RemoteException;

    public Collection findByLastName(String lastName)
    throws FinderException, RemoteException;
}
```

Source: [http://java.sun.com/j2ee/tutorial/1.3-fcs/doc/BMP2.html#62922](http://java.sun.com/j2ee/tutorial/1.3-fcs/doc/BMP2.html#62922)
**EJB 2.x Remote Interface**

```java
public interface SavingsAccountRemote extends EJBObject {
    public void debit(BigDecimal amount)
        throws RemoteException;
    public void credit(BigDecimal amount)
        throws RemoteException;
    public String getFirstName()
        throws RemoteException;
    public String getLastName()
        throws RemoteException;
    public BigDecimal getBalance()
        throws RemoteException;
}
```


**EJB 2.x Bean Class**

```java
public class SavingsAccountBean {
    public String ejbCreate(String id, String firstName,
                           String lastName, BigDecimal balance)
        throws CreateException {
        if (balance.signum() == -1) {
            throw new CreateException(
                "A negative initial balance is not allowed."
            );
        }
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
        this.balance = balance;
        return id;
    }
}
```

EJB 2.x Bean Class

... public void ejbPostCreate() {
    // The ejbPostCreate method must have the same
    // input parameters and return type as the
    // ejbCreate method.
    //
    // If you want to set up a relationship you should
    // do so in the ejbPostCreate method.
}

public void ejbRemove() {}
public void ejbLoad() {}
public void ejbStore() {}
...

Source http://java.sun.com/j2ee/tutorial/1.3-fcs/doc/BMP2.html#62922

EJB 2.x Bean Class

... public void debit(BigDecimal amount) {
    balance = balance.subtract(amount);
}

public void credit(BigDecimal amount) {
    balance = balance.add(amount);
}

public String getFirstName() {
    return firstName;
}

public String getLastName() {
    return lastName;
}

public BigDecimal getBalance() {
    return balance;
}
...

Source http://java.sun.com/j2ee/tutorial/1.3-fcs/doc/BMP2.html#62922
**EJB 2.x ejb-jar.xml**

```xml
<enterprise-beans>
  <entity>
    <description>Savings Account Bean</description>
    <display-name>SavingsAccount</display-name>
    <ejb-name>SavingsAccount</ejb-name>
    <home>example.bean.SavingsAccountHome</home>
    <remote>example.bean.SavingsAccountRemote</remote>
    <ejb-class>example.bean.SavingsAccountBean</ejb-class>
    <persistence-type>Container</persistence-type>
    
    <cmp-version>1.x</cmp-version>
    <cmp-field>
      <field-name>id</field-name>
    </cmp-field>
    <cmp-field>
      <field-name>firstName</field-name>
    </cmp-field>
    <cmp-field>
      <field-name>lastName</field-name>
    </cmp-field>
    <primkey-field>
      <field-name>id</field-name>
    </primkey-field>
  </entity>
</enterprise-beans>
```

**EJB 2.x jonas-ejb-jar.xml**

```xml
<jonas-entity>
  <ejb-name>SavingsAccount</ejb-name>
  <jndi-name>SavingsAccount</jndi-name>
  <jdbc-mapping>
    <jndi-name>jdbc_conn1</jndi-name>
    <jdbc-table-name>SAVINGS_ACCOUNT</jdbc-table-name>
    <cmp-field-jdbc-mapping>
      <field-name>id</field-name>
    </cmp-field-jdbc-mapping>
    <cmp-field-jdbc-mapping>
      <field-name>firstName</field-name>
    </cmp-field-jdbc-mapping>
    <cmp-field-jdbc-mapping>
      <field-name>lastName</field-name>
    </cmp-field-jdbc-mapping>
    ...  
</jdbc-mapping>
</jonas-entity>
```
EJB 2.x jonas-ejb-jar.xml

... 

<cmp-field-jdbc-mapping>
  <field-name>balance</field-name>
  <jdbc-field-name>BALANCE</jdbc-field-name>
</cmp-field-jdbc-mapping>

<finder-method-jdbc-mapping>
  <jonas-method>
    <method-name>findByLastName</method-name>
  </jonas-method>
  <jdbc-where-clause>
    WHERE LAST_NAME = ?
  </jdbc-where-clause>
</finder-method-jdbc-mapping>

</jdbc-mapping>
</jonas-entity>

EJB 2.x Client

InitialContext context = new InitialContext();
SavingsAccountHome home =
    (SavingsAccountHome)context.getEJBHome();

SavingsAccount john =
    home.create("123", "Doe", "John", zeroAmount);

john.credit(new BigDecimal("88.50"));
john.debit(new BigDecimal("20.25"));
BigDecimal balance = john.getBalance();

Collection c = home.findByLastName("DOE");

Source http://java.sun.com/2ee/tutorial/13-fcs/doc/BMP2.html#62922
**Hibernate History**

- **Grass roots development (2001)**
  - Christian Bauer
  - Gavin King

- **JBoss later hired lead Hibernate developers (2003)**
  - Brought Hibernate under the Java EE specification
  - Later officially adopted as the official EJB3.0 persistence implementation for the JBoss application server.

- **EJB 3.0 Expert Group (2004)**
  - Key member which helped shape EJB3.0 and JPA

- **NHibernate**
  - .NET version release in 2005

**Hibernate Goals**

- **Prevent leakage of concerns**
  - Domain model should only be concerned about modeling the business process, not persistence, transaction management and authorization
  - Flaw of EJB2.x

- **Transparent and automated persistence**
  - Complete separation of concerns between domain model objects and the persistence mechanism.
  - Persistent solution does not involve writing SQL

- **Metadata in XML**
  - Object/Relational Mapping should provide human readable mapping format (not just a GUI mapping tool)

- **Reduction in LOC**

- **Importance of domain object model**
Why Hibernate?

- **Impedance mismatch**
  - Object-oriented vs. relational

- **Failure of EJB 2.x**
  - Entity Beans were extremely slow, complex

- **Java developers are not database developers**
  - Reduce the need for developers to know and fully understand database design, SQL, performance tuning
  - Increase portability across database vendors

- **Increase performance by deferring to experts**
  - Potential decrease in database calls
  - More efficient SQL statements
  - Hibernate cache usage

Why not Hibernate?

- **Overkill for small number of tables**
- **Complex legacy database schema**
- **Heavy batch processing**
- **Advanced queries / more SQL control**
- **Free, but tied to third party**
- **Complexity / ramp up / support**
- **Scaling concerns (Shards)**
- **Gavin King is somewhat opinionated 😊**
Who Uses Hibernate?

- Ubik-Ingénierie, ubik-ingeniere.com, Roubaix, France
- Fedelta POS, fedelapos.com, Brisbane, Australia
- Slickserv, slickserv.com, San Francisco, California, USA
- Company name, Location: Softgate Commerce, NY, USA
- Company name, Location: GPT Argentina, La Plata, Buenos Aires, Argentina
- Open Source Project: Tracker
- Company name, Location: TerraContact Inc., Montreal, Canada
- Company name, Location: LF Inc., Tampa, FL
- Company name, Location: Elastic Path Software, Vancouver, BC, Canada
- Company name, Location: argus Barcelona, Europe
- Company name, Location: AT&T Labs, Tampa, Florida
- Company name, Location: JTeam, Amsterdam, The Netherlands
- Company name, Location: 1Gonia, Paris, France
- Company name, Location: TDC Internet, Warsaw, Poland
- Company name, Location: PriceWaterhouseCoopers, Tampa, Florida
- Company name, Location: JHL Business Solutions Ltd., Hong Kong
- Company name, Location: Intracsoft International, Belgium, Brussels
- Company name, Location: Burgerweeshuis, Netherlands, Deventer
- Company name, Location: Cisco Learning Institute, Phoenix, AZ USA
- Company name, Location: Open Lab S.r.l., Florence I
- Company name, Location: DriveNow, Australia
- Sony Computer Entertainment Europe, SCEA, Studio Liverpool, Liverpool, United Kingdom
- Company name, Location: Church and People, New York
- Crnk Clothing, t shirts and apparel
- Mailvision, End-to-End SIP solutions, Israel. (http://www.mailvision.com)
- Pyromod Software Inc., Creator of BestCrosswords.com, Montreal, Canada. (http://www.pyromod.com)
- Travel Toucan Travel Site

Source hibernate.org

Hibernate Jobs (as of Aug 2008)

- From indeed.com
  - Claims to compile data from most major job sites

![Job Trends](https://via.placeholder.com/150)

<table>
<thead>
<tr>
<th>May'05</th>
<th>Sep'05</th>
<th>Jan'06</th>
<th>May'06</th>
<th>Sep'06</th>
<th>Jan'07</th>
<th>May'07</th>
<th>Sep'07</th>
<th>Jan'08</th>
<th>May'08</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.20</td>
<td>0.25</td>
<td>0.30</td>
<td>0.35</td>
<td>0.40</td>
<td>0.45</td>
</tr>
<tr>
<td>0.50</td>
<td>0.55</td>
<td>0.60</td>
<td>0.65</td>
<td>0.70</td>
<td>0.75</td>
<td>0.80</td>
<td>0.85</td>
<td>0.90</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Source indeed.com
JPA Jobs (as of Aug 2008)

- From indeed.com
  - Claims to compile data from most major job sites

![Job Trends Chart](image)

**Customized Java EE Training:**
http://courses.coreservlets.com/
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.
Hibernate – Step 1

- [http://www.hibernate.org](http://www.hibernate.org)
- Click on ‘Downloads’ link

Hibernate – Step 2

- Select the core binary release

<table>
<thead>
<tr>
<th>Binary Releases</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Package</strong></td>
</tr>
<tr>
<td>Hibernate Core</td>
</tr>
</tbody>
</table>

- Select download type

<table>
<thead>
<tr>
<th>Package</th>
<th>Release</th>
<th>Filename</th>
<th>Size</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>hibernate</td>
<td>3.3.1.GA</td>
<td>hibernate-distribution-3.3.1-GA-57.tar.gz</td>
<td>26763494</td>
<td>Platform-Independent</td>
</tr>
<tr>
<td>hibernate</td>
<td>3.3.1.GA</td>
<td>hibernate-distribution-3.3.1-GA-57-jar.zip</td>
<td>21462800</td>
<td>Platform-Independent</td>
</tr>
</tbody>
</table>

Totals: 1 2 60166294
Hibernate – Step 3

• Unzip the Download
  – hibernate-distribution-3.3.1.GA-dist.zip
  – Copy jars from locations under root of zip
    • hibernate3.jar
    • hibernate-distribution-3.3.1.GA/lib/required
  – Drop the jars from into the lib directory of your project (or other location you can add to your projects classpath)

• Obtain a Simple Logging Façade for Java (SLF4J) Implementation
  – http://www.slf4j.org/download.html
    • slf4j-simple-1.5.3.zip
  – Unzip and copy slf4j-simple-1.5.3.jar into lib directory of your project
    • slf4j-simple-1.5.3.jar under root directory of download

Hibernate – Step 4

• Within Eclipse

  Right click on project to get to “Configure Build Path”

  Under the “Libraries” tab, click “Add JARs” to add the Hibernate jars to the project
Oracle Express – Step 1

  - Download OracleXE.exe install
  - Also available for Linux
    - Debian, Mandriva, Novell, Red Hat and Ubuntu

Oracle Express – Step 2

- **Run OracleXE.exe**
  - Accept license agreement
  - Choose install location
  - Set SYSTEM password to 'system'
Oracle Express – Step 3

Confirm

Complete

Login and test http://127.0.0.1:8080/apex/

Starting and Stopping Oracle

• Oracle is automatically running upon install
• Start and stop Oracle using the StartDB and StopDb scripts
  – C:\oracleexe\EE\app\oracle\product\10.2.0\server\BIN\StartDB.bat
  – C:\oracleexe\EE\app\oracle\product\10.2.0\server\BIN\StopDb.bat
• In Windows, can use Start Menu Options
JavaDB Configuration

• JavaDB is a version of Derby that comes packaged with Java 6

• Configuration – set environment variables
  – DERBY_HOME
    • Value should be location of JavaDB root directory
    • Example: C:\Program Files\Sun\JavaDB
  – PATH
    • Append JavaDB bin directory to existing PATH variable
    • Example: C:\Program Files\Sun\JavaDB\bin

• Start Server by calling startNetworkServer script

• Stop Server by calling stopNetworkServer script (in another window)
Wrap-up

Customized Java EE Training: http://courses.coreservlets.com/
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

Preview of Next Sections

• Walk through a simple, but full, end to end example
Summary

- **Refresher in application architectures**
  - Service-based business logic
  - Rich domain model
  - Combination

- **Traditional persistence implementation**
  - Persistent implementations
    - Entity Beans
    - JDBC
  - JDBC example

- **Motivation**
  - Origination and history of Hibernate
  - Reasons for Hibernates development
    - Impedance mismatch
    - Failure of EJB 2.x
    - Java developers are not database developers
    - Performance benefits

- **Installation**

Questions?

Customized Java EE Training: [http://courses.coreservlets.com/](http://courses.coreservlets.com/)
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate/JPA, Java 5 & 6.
Developed and taught by well-known author and developer. At public venues or onsite at your location.