Hibernate

* ORM (Object Relational Mapping) framework
  + An ORM will match your entity classes to tables in the database
  + Pros
    - Much faster than writing the JDBC
      * Hibernate does the SQL queries for you
      * You just have to configure it
      * Reduces boilerplate code
        + Boilerplate is code that is not so much about logic but necessary code for thing to run.
    - It an OOP approach to writing applications
      * In general it is beneficial to think in terms of objects while programming
        + No ResultSet you automatically get back objects from the database
    - Decouples your Java application from the database syntax
      * Once you set up hibernate you can swap what database and RDBMS.
      * Makes your application more independent
        + Avoid tight coupling
* Hibernate Object states
  + Every entity object in Hibernate can be in one of three states
    - Transient
      * This is an entity object with an id of 0
      * It is not saved in the database
    - Persistent
      * This entity has a non zero id
      * The entity is currently part of a session
    - Detached
      * This entity has an id
      * The session that the entity was part of has closed
* Core Interfaces of Hibenate
  + Configuration
    - AN OBJECT that is used to set up your mappings and relationships
    - Reads the hibernate.cfg.xml for information
  + SessionFactory
    - Creates sessions for a particular database
    - An application only needs one sessionfactory
    - They are big expensive objects
  + Session
    - Core interface for making CRUD operations
      * Key methods
        + Save
        + Update
        + Remove
        + Get
  + Transaction
    - An interface for creating transactions and committing the CRUD operations from a session
  + Query/criteria
    - Interfaces for performing queries on your database
* Ways to query in Hibernate
  + Criteria Interface
    - Very OOP approach
    - Get a Criteria object
    - You add restrictions to that object
    - Then execute that criteria
  + HQL
    - OOP version of SQL
  + Native SQL queries
    - Not recommended
    - You do not get back your information as a nice object
    - Ties your Java code to a specific version of SQL syntax
* Loading information into object
  + Eager Loading
    - When you get an object and nested objects within it all at once
    - You get the parent and its children in one query immediately
  + Lazy Loading
    - When you get an object the nested objects are not loaded in
    - Hibernate will not make a query for those nested objects until a nested object is actually being called in a method
  + Defaults in Hibernate
    - Nested collections = Lazy Loaded
    - Nested single object = Eager Loaded
* Annotations in Hibernate
  + @Entitiy – marks a class as an entity
  + @Table – link this entity to a table
  + @Id – mark this field as a primary key
  + @GeneratedValue – mark this field as being auto incremented
  + @Column- connect a field to a column in a table
  + @JoinColumn – this field is a foreign key to another table
  + Multiplicities
    - @OneToMany
    - @ManyToMany
    - @ManyToOne