Design: Data Layer

Karim Sultan

Objectives

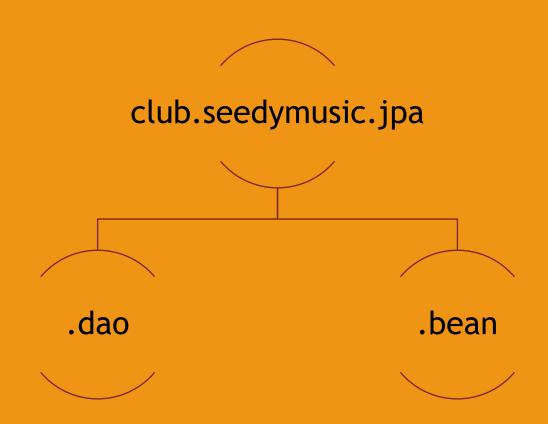
To keep the data layer extremely simple

 Only uses 3 POJO objects To use the Java Persistence API of JEE

 Implemented with Hibernate To enable connection pooling

 Managed by Hibernate over Tomcat

Package Structure

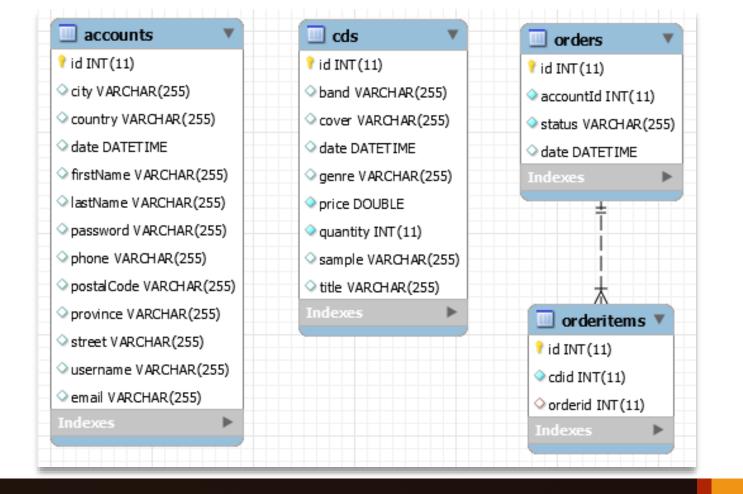


Data Objects

Transfer Data Access Objects Database Objects (DAO) (Beans) Hibernate CdDAO Cd Manager MySQL 5.7 AccountDAO Account OrderDAO Order

Model: Beans

<<Java Class>> <<Java Class>> <<Java Class>> Account ⊕ Cd ● Order club.seedymusic.jpa.bean club.seedymusic.jpa.bean club.seedymusic.jpa.bean DEFAULT VALUE: String DEFAULT VALUE: String a id: int a id: int □ id: int accountld: int username: String title: String status: String password: String band: String date: Date firstName: String genre: String Order() lastName: String price: double getld():int street: String a cover: String setId(int):void city: String a sample: String getAccountId():int province: String quantity: int setAccountld(int):void a country: String date: Date getStatus():String postalCode: String Cd() setStatus(String):void a phone: String getDate():Date getOrderItems():Set<OrderItem> email: String setDate(Date):void setOrderItems(Set<OrderItem>):void date: Date getId():int getDate():Date Account() setId(int):void setDate(Date):void getId():int getTitle():String setId(int):void setTitle(String):void getDate():Date getBand():String setDate(Date):void setBand(String):void -orderItems 0..* getUsername():String getGenre():String setUsername(String):void setGenre(String):void <<Java Class>> getPassword():String getPrice():double OrderItem setPassword(String):void setPrice(double):void club.seedymusic.jpa.bean getFirstName():String getCover():String □ id: int setFirstName(String):void setCover(String):void a cdid: int getLastName():String getSample():String OrderItem() setLastName(String):void setSample(String):void OrderItem(int) getStreet():String getQuantity():int getld():int setStreet(String):void setQuantity(int):void setId(int):void getCity():String getCdid():int setCity(String):void setCdid(int):void getProvince():String equals(Object):boolean setProvince(String):void hashCode():int getCountry():String setCountry(String):void getPostalCode():String setPostalCode(String):void getPhone():String setPhone(String):void getEmail():String setEmail(String):void



Database

Catalog Component Catalog Web Service Catalog DAO Catalog Bean MySQL DB API call API calls can add, get, update, list CDs instantiate Perform transaction Via Hibernate Return result Catalog Bean Catalog DAO Catalog Bean MySQL DB Catalog Web Service

How to Use: Catalog Component

order Component order DAO MySQL DB order Web Service order Bean Account Bean API call API calls can be for order/account [For oder] instantiate [For Account] instantiate Perform transaction Via Hibernate Return result Order/Account Bean order DAO order Web Service order Bean Account Bean MYSQL DB

How to Use: Order Component

Hibernate Jars

- Hibernate requires these JARs (including C3P0 connection pooling):
 - antlr-2.7.7.jar
 - c3p0-0.9.2.1.jar
 - dom4j-1.6.1.jar
 - hibernate-c3p0-4.3.11.Final.jar
 - hibernate-commons-annotations-4.0.5.Final.ja
 - hibernate-core-4.3.11.Final.jar
 - hibernate-entitymanager-4.3.11.Final.jar
 - hibernate-jpa-2.1-api-1.0.0.Final.jar
 - jandex-1.1.0.Final.jar
 - javassist-3.18.1-GA.jar
 - jboss-logging-3.1.3.GA.jar
 - jboss-logging-annotations-1.2.0.Beta1.jar
 - jboss-transaction-api_1.2_spec-1.0.0.Final.jar
 - mchange-commons-java-0.2.3.4.jar
- They are checked into github under WEB-INF/lib

TODO: Example Java Code

- Example Java Code can be found on Github
 - JSPs:
 - /test/testDAO.jsp
 - /test/testDAOList.jsp
 - /test/testDAOGet.jsp
 - Servelets:
 - /test/TestDAO.java
 - /test/TestDAOList.java
 - /test/TestDAOGet.java

```
<?xml version="1.0" encoding="utf-8"?>
2 - <!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
    <hibernate-configuration>
        <session-factory>
            cproperty name="hibernate.connection.driver class">com.mysql.jdbc.Driver
            cyroperty name="hibernate.connection.url">jdbc:mysql://localhost:3306/seedy db/property>
9
            cproperty name="hibernate.connection.username">seedy</property>
10
            cproperty name="hibernate.connection.password">seedy2017</property>
11
12
            cproperty name="hibernate.dialect">org.hibernate.dialect.MySQLDialect/property>
            cproperty name="show_sql">true</property>
13
14
            cproperty name="format_sql">true
15
            cproperty name="hbm2ddl.auto">update</property>
16
            <mapping class="club.seedymusic.jpa.bean.Cd" />
17
        </session-factory>
    </hibernate-configuration>
```

Configuration File Format

Test Framework

Test via

/test/testCDDAO.jsp
/test/testAccountDAO.jsp
/test/testOrderDAO.jsp

Test controller servlets /test/TestCDDAO
/test/TestAccountDAO
/test/TestOrderDAO