Project Part II, CST8288 – Fall 2017

School of Advanced Technology Information and Communications Technology

---

Purpose

Project 2 is somewhat of a continuation of your "Symphony" project 1. Part 2 involves the use of JDBC (java database connectivity, MySQL), Java Servlet(s), a "Servlet Engine" (Jarkarta Tomcat), and lastly, the implementation of a design pattern to effectively "factor out" or "Decouple" the database logic from the domain layer logic in your application. This project will:

- Exercise your skills in developing Java Servlets and deploying in the Tomcat Servlet Engine environment.
- Give you an opportunity to access a database via Java using JDBC by extending an existing framework (must use the provided framework code, no "auto generated" code will be accepted) and implementing a Data Access Object design pattern (DAO) for the persistence layer (Domain object(s) must be created from existing state stored in the database).
- Provide a further introduction to multi-tiered applications in the Java world (your Servlet and DB connectivity)
- Introduce you to WEB applications in a multi-tiered environment using Java, particularly the server side components.
- Work in your teams that were originally created for the "Symphony" project.

Program Description

It is imperative that your application present only **1 composer per page** with a **maximum** of **10 rows of information** per page. You will be required to display more than 10 rows to demonstrate the ability of you application to "page" correctly thru compositions and composers (you will need to add additional rows of information to the DB). You must include at the bottom of the page a button for the "next 10 rows or the next composer". Your application should display a "page" similar to the diagram below: (You are free to create your own view as long as it meets the aforementioned criteria providing the details below.) You must also support multiple concurrent users (your servlet must be "threadsafe").

Composer: M. Clementi

Sonatina in C	
	Op. 36, No. 3
	Op. 36, No. 3
	Op. 36, No. 3

Next Composer

Your application must use JDBC to access a database containing the Symphony tables and must build full domain objects within the servlet. The following Sql files are provided for your use (however you will need to create additional data to force compositions for a composer to span more than 1 page). Use the sql script **createtables.sql** to setup the table, use the sql script **loadtables.sql** to load the table with data (insert). Your database access logic must be factored out into a "Data Access Object" (these DAO design patterns will be discussed in lab/lecture) as demonstrated in the marina sample. You must use the full DAO design pattern as presented in the marina example and the provided DAO framework for building composer objects, composition objects and movement objects. Using just the DAO class is not enough, this does not constitute the complete design pattern, rather you must build Composition objects and movement objects (and composer objects?). A composer can have 0,1 or many compositions which may have 0,1,or many movements. You will need to create the appropriate Domain and DAO classes for your application domain objects (Composers, Compositions and Movements).

Your application should make use of MYSQL, however here is an example of the drivers and URLS:

Here are some examples of the URL's and DRIVER's for different database management products (you may use MYSQL, ORACLE, …):

The following URL and DRIVER would be used for the **MySQL database**:

String database URL **jdbc:mysql://localhost/mysql**
String databaseDriver **com.mysql.jdbc.Driver**

The following URL and DRIVER would be used for the **Oracle 8i database**:

String database URL **jdbc:oracle:thin:@locahost:1521:<SID>**
String databaseDriver **oracle.jdbc.driver.OracleDriver**

The following URL and DRIVER would be used for the **IBM Cloudscape database**:

String databaseURL **jdbc:db2j:Symphony**
String databaseDriver **com.ibm.db2j.jdbc.DB2jDriver**

Program Documentation

You code must adhere to the published Java standards discussed in class and provide the necessary JavaDocs for you application.

Submission Requirements

Your submission should follow the email submission standards for this project. The project attachment (zip file) should be named CST8288Project2<initials for Member1>-<initials for Member2>.zip and contain the following items:

- SymphonyList.html — Html file for access your servlet/jsp.
- SymphonyComposition.java — Your Java servlet (minimum requirement)
- SymphonyComposition.class — Your class file (minimum requirement)
- <???> — Additional classes you create as required by your implementation
- docs — The "docs" folder containing your JavaDocs for your application.
- and all other require source and/or classes for your application (including the framework)

This project software component is due week 14. You will be required to demonstrate your project by sending your instructor a "link" to your opening web page via email. Your instructor will then "demo" the features required.