Project Overview

The purpose of this assignment is to explore JSP/JSTL presentation layer technologies, JDBC database interaction, and (for graduate students) to implement the Model-View-Controller (MVC) pattern. JDBC interaction will be C(reate), R(read), U(pdate) and D(elete). In all subsequent assignments we will use JPA as a persistence technology, but for this assignment you must use JDBC.

This assignment consists of 2 NetBeans web application projects using the Glassfish application server and MySQL database connectivity.
Undergraduate students must submit one project of their choosing, but are encouraged to complete both to further their understanding of these concepts.
Graduate students must submit both projects.
You may work with either or both of the MySQL sample databases:

1. World database -> http://dev.mysql.com/doc/world-setup/en/index.html
2. Employee database -> http://dev.mysql.com/doc/employee/en/index.html

Database files for both will be posted with this assignment. You may use the same database for both projects if you choose, or you may use World for one, and Employee for the other. If you followed the defaults for Windows MySQL installation, the World database should be installed and populated already.

Database Requirements

Create and populate the database(s) as needed. If using the world database, it must be named world. If using the employees database, it must be named employees.

Create a MySQL user named itmd4515 with full rights to the database(s). The user must have a password of itmd4515. Do not use the root user. Use this user and password for all your connection pools.

Requirements (MVC Project)

Create a NetBeans Web Application project named “UID MP1 MVC” but use a web context of /uidmp1mvc. For example, my NetBeans project will be called “Spyrison MP1 MVC” with a web context of /spyrisonmp1mvc.

Use NetBeans to create a Glassfish JDBC Connection Pool named uidMp1MvcPool connecting to the database for this application. Use NetBeans to create a Glassfish JDBC Resource named jdbc/uidMp1Mvc. For example, mine would be spyrisonMp1MvcPool and jdbc/spyrisonMp1Mvc. Make sure that your glassfish-resources.xml file is included with your submission (it should be in the Server Resources section of your project).

Implement the MVC pattern as follows:

- JavaBean classes (POJOs) for the Models
- Data Access by container managed beans (as noted in Week 3 class and demo)
- Servlet Technology for the Controllers
- JSP/JSTL Technology for the Views

Follow the separation of concerns principle. Perform database interaction in your data access layer. Process user requests (both GET and POST) through your controller servlets. Invoke the data access layer from your controller servlets. Place information in the appropriate scope(s), and dispatch requests to the appropriate views. Use your views as a presentation layer to display information from your models.

Provide a welcome or index page that allows traversal to all CRUD operations. Presentation is left up to you. You may use multiple pages, or you may design a layout of your choosing, but all CRUD operations must be implemented.

Consider server-side validation of user input in your controller design, and display appropriate messages and navigation if user input fails to validate.

An example:

- Welcome

- Create a new record
- Read all records
    - For each record, provide navigation to Update or Delete that record. Upon successful operation, user is returned to "read all records" page.

Include appropriate navigation between the pages, including the ability to return "home." Do not leave dead-end pages with no navigation. Consider usability in your design. Basic CSS is highly encouraged.

If you find it beneficial to use additional libraries in your project, you are free to do so provided you follow the general project requirements and fully document their use in your README. I encourage you to explore and be creative.

Installation, Compile & Run-Time Requirements

This application was developed using a Windows desktop machine configured with—

- NetBeans IDE 7.3.1 (Build 201306052037)
- Java: 1.7.0_17; Java HotSpot(TM) Client VM 23.7-b01
- Java Runtime: Java(TM) SE Runtime Environment 1.7.0_17-b02
- System: Windows 8 version 6.2 running on x86; Cp1252; en_US (nb)
- MySQL - Software version: 5.5.27 - MySQL Community Server (GPL)
- GlassFish Server Open Source Edition 4.0 (build 89)

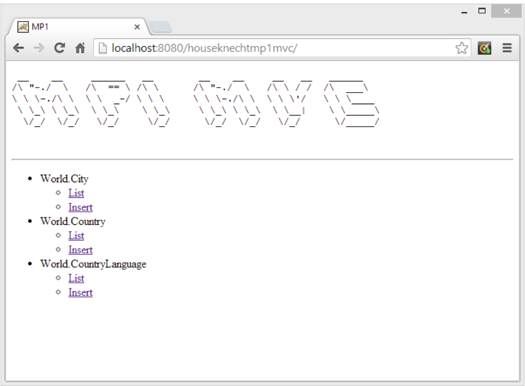
Per the glassfish-resource.xml file included in the source files for this project assumes the following configuration has been established on the Glassfish Server:

- JDBC Resource: jdbc/houseknechtMp1Mvc
- JDBC Connection Pool: houseknechtMp1MvcPool
    - URL: jdbc:mysql://localhost:3306/world
    - User: itmd4515
    - Password: itmd4515

- Note – itmd4515 has been given full privileges/authority to the world database.

The world database can be configured using the world_innodb.sql script included in the Setup sub-directory.

Story board & Screen Captures

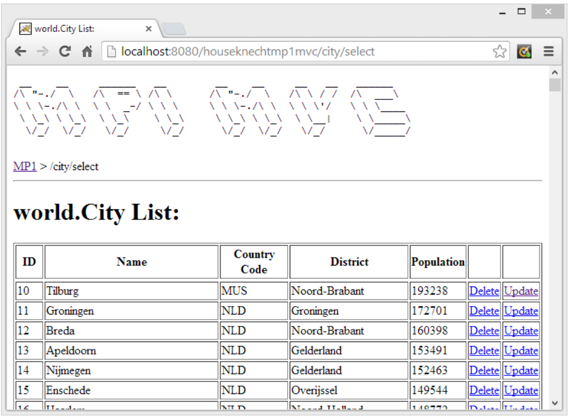


localhost:8080/houseknechtmp1mvc/

- World.City
  - List
  - Insert
- World.Country
  - List
  - Insert
- World.CountryLanguage
  - List
  - Insert

On startup, the application displays a simple welcome page (depicted above). The page lists 3 tables targeted by application. For the 3 tables listed, the user can then click on the "List" or "Insert" link. Each of these pages follows a similar format regardless of the table selected. The List pages present the contents of the target table using a simple columnar table layout. The Insert pages present a simple form allowing the user to define the properties of the new record (depicted below).

List Page Layout:



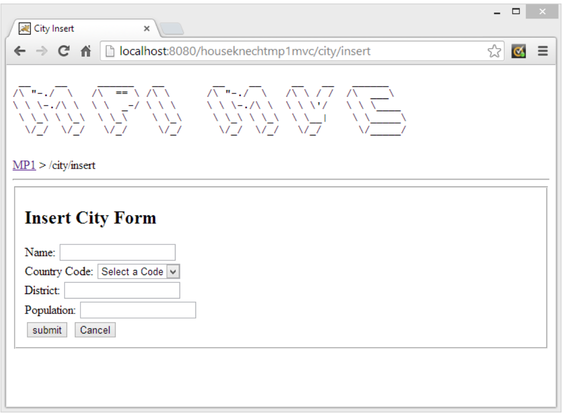
localhost:8080/houseknechtmp1mvc/city/select

MP1 > /city/select

world.City List:

ID	Name	Country Code	District	Population		
10	Tilburg	MUS	Noord-Brabant	193238	Delete	Update
11	Groningen	NLD	Groningen	172701	Delete	Update
12	Breda	NLD	Noord-Brabant	160398	Delete	Update
13	Apeldoorn	NLD	Gelderland	153491	Delete	Update
14	Nijmegen	NLD	Gelderland	152463	Delete	Update
15	Enschede	NLD	Overijssel	149544	Delete	Update

Insert/Update Page Layout:



localhost:8080/houseknechtmp1mvc/city/insert

MP1 > /city/insert

Insert City Form

Name:

Country Code: Select a Code

District:

Population:

From the List pages, the user can click on the “Delete” or “Update” links to modify individual records. If the user clicks “Delete” then the selected record is deleted. When this occurs, the screen refreshes and a message is displayed at top of the page identify the record that was deleted.

On the Update and Insert, Validation is performed on the server. No client side validation has been implemented in this iteration of the application. If the user enters invalid value on either the insert or update form pages, then the page will refresh and a message will be displayed at the top of the page.