PROG32758 – Quiz 4

Fall 2019

(1 bonus mark – 33% more, if marked in class ☺)

Partner up today! This is a nice review of just about everything we've done so far!

Note: I am only going to provide minimal instruction here, so read carefully and figure it out!

Also note: you will be pressed for time! Don't waste any on graphics until you're done!

The idea is to use the SQL JSTL library to display all the dealers from the SheridanUsedCars database in a JSP form - along with a max price field, post to a Controller servlet, use a PreparedStatement in the servlet to retrieve all the cars from that dealer below that price, iterate through the ResultSet to store each returned row's info in a POJO (and store each POJO in an ArrayList), store the ArrayList in a request attribute, forward to a view JSP, and iterate through the ArrayList attribute in the view JSP using just EL and JSTL.

<u>Exercise 1</u>

Create a new project. I'm going to provide you with some code in this first exercise, but please type it – do not copy and paste. It's really the only way to learn and know it.

Also, please note this first exercise mixes Model and View, and is a nice example of a simple solution that is HARD to maintain!

Inside, create a new JSP called form.jsp.

Include the JSTL SQL and Core libraries in your JSP as directives:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```

Create a connection to your SheridanUsedCars database (run the accompanied SQL script to create the database for you) using just JSTL as follows:

```
<sql:setDataSource var="db" driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/SheridanUsedCars"
user="root" password="root" />
```

Be sure to change your password if required!

Execute a query to retrieve all the manufacturers and their index numbers.

```
<sql:query dataSource="${db}" var="result">SELECT * FROM
manufacturer</sql:query>
```

Now create a regular HTML form with a dropdown which displays the name of the manufacturers as options and passes their manufacturerID as values. Inside the tags for your dropdown, iterate through the `${result}` you generated with your SQL query. This part is up to you to modify to suit your needs. I'll give you some of the code to help you out…

```
<c:forEach var="row" items="${result.rows}">
     ${row.manufacturerID} ${row.manufacturer}
</c:forEach>
```

Additionally, add a field called maxPrice to store text to your form.

Post the result to a servlet you create called Controller in an appropriate package.

Exercise 2

Note: this is actually the start of the preferred MVC way to do things. Generally, we never want to mix up model (DAO) code with view (JSP) code!!! It may be a little more work up front, but it will benefit you a lot and save you hard efforts to separate them out later!

In your Controller servlet, retrieve the manufacturerID and maxPrice your user entered.

Convert the maxPrice parameter to a double, but use a try catch block to ensure it all worked as expected.

Create a DAO class in an ca.sheridancollege.dao package.

Inside, create a connection to your database. Then create a second method which gets a connection and then creates and executes a PreparedStatement to retrieve all the cars matching the id and less than or equal to maxPrice. Remember to set your SQL parameters – do NOT concatenate them!

Execute your DAO method from the Controller!

Iterate through the ResultSet in the Controller using a while loop.

Exercise 3

Create a POJO called Car in a ca.sheridancollege.beans package which follows all the POJO rules. This means it must implement Serializable, have only private variables, use getters and setters, and provide overloaded constructors.

Additionally, create an ArrayList of Car.

As you iterate through the ResultSet from exercise 2, store the returned values in a new Car instance by setting the appropriate values. Store each car instance in your ArrayList.

After the loop has finished, store your ArrayList in a request attribute.

RequestDispatch forward to view.jsp.

Exercise 4

Create view.jsp.

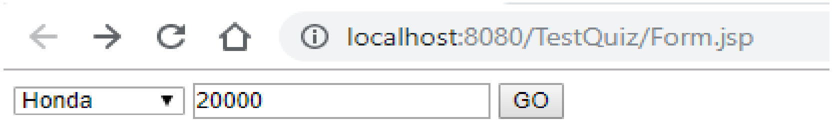
Add the JSP directive to include the JSTL core taglib. See exercise 1 if you're stuck.

Using only Expression Language and JSTL, iterate through your ArrayList and print out the relevant cars for your user in an appealing fashion.

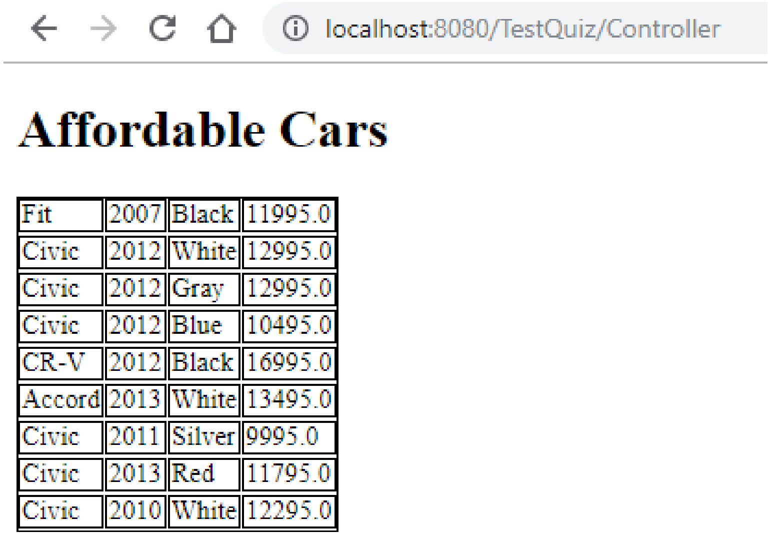
If you get this done, go to town on the graphics. I'd like to see alternating row background colours, but you are NOT allowed to use any Java beyond JSTL to do so. Hint: you may want to Google the core library set tag

<c:set…. for storage and incrementing, then test in the loop with a <c:if… tag. Play around!

Sample Run



Honda ▾ 20000



Affordable Cars

Fit	2007	Black	11995.0
Civic	2012	White	12995.0
Civic	2012	Gray	12995.0
Civic	2012	Blue	10495.0
CR-V	2012	Black	16995.0
Accord	2013	White	13495.0
Civic	2011	Silver	9995.0
Civic	2013	Red	11795.0
Civic	2010	White	12295.0

Good luck ☺