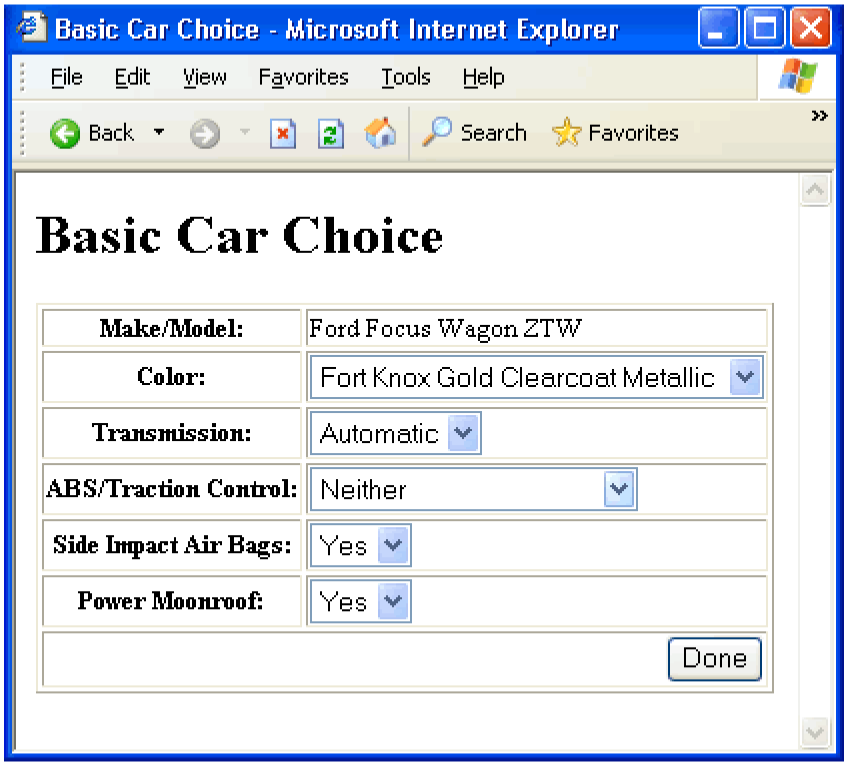
Project – Unit 5

We now have a client-server system where the end users, mostly non-programmers, can easily add new car models to the system without over writing each other's data entry.

We now need to extend this system so it can be used over the web. You will need to extend the functionality of Client (created in Unit 4) with the following features

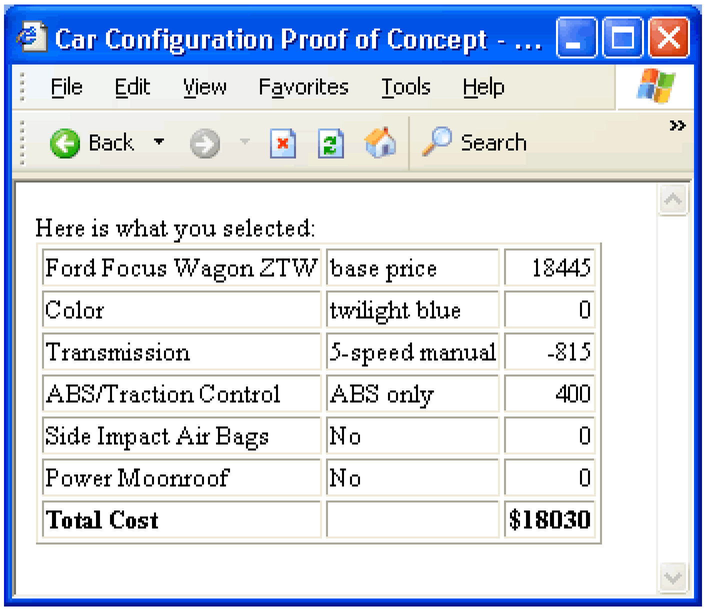
1. Show list of available models in drop down box.
2. User selects options for their cars and submit them as shown below:
    a. *I realize the picture below does not show the list of available models.*



Basic Car Choice

Make/Model:	Ford Focus Wagon ZTW
Color:	Fort Knox Gold Clearcoat Metallic
Transmission:	Automatic
ABS/Traction Control:	Neither
Side Impact Air Bags:	Yes
Power Moonroof:	Yes

3. Page will display the details of their choice as shown below.


Car Configuration Proof of Concept - ...
File Edit View Favorites Tools Help
Back
Search

Here is what you selected:

Ford Focus Wagon ZTW	base price	18445
Color	twilight blue	0
Transmission	5-speed manual	-815
ABS/Traction Control	ABS only	400
Side Impact Air Bags	No	0
Power Moonroof	No	0
Total Cost		**$18030**

Use these screen shots as a design reference on which to model the interface of your own page.

Our first goal here is to develop the layout of the forms page, as well as to set up choices for users to select. Then we need to update the page to reflect choices made on the forms page.

Also, as you begin development, be sure to use the same configuration options that you used in previous unit. Specifically, I'm referring to the choices regarding the moonroof, color, transmission, brakes, and airbags.

Technical Requirements

You have to design this task being mindful of technical requirements.

1. Create a Servlet that interacts with Client in Unit 4 to get the list of available models.
2. Create another Servlet that can interact with Client created in Unit 4 to get the data for the list of available OptionSets.
3. Create a JSP that shows the OptionSets and prints the selected choices showing the total cost.

Please keep in mind that data displayed is dynamic (i.e. read from the static object – LinkedHashMap)

Project 1 – Unit 6

In this unit you have to provide Create, Update and Delete operations for persisting the LinkedHashMap into a Database. The operation functions should be called whenever:

1. A new Automobile is added to the LinkedHashMap.
2. An Automobile is deleted from LinkedHashMap
3. An Automobile is updated.
4. An Automobile is deleted.

You will need to design and implement:

1. A database schema. (Be sure to apply rules of normalization discussed in class).
2. Create a set of classes for managing the Create, Update and Delete operations.
3. You should write a driver (in main) and test out the functionality in a console program on the server side.

What is not to be included:

- **Implement Database functionality in the server.**
- **Do not implement any functionality from the client side.**
- Do not save Option Choices.
- Client interaction for creating, updating and deleting are not to be implemented or tested. All functionality should be tested in a driver from Server side. In later versions (which will never be built in this course) the interactions with client can be setup.

How?
You will be given detailed examples in class on how to setup a database and connectivity.

- **Grading your Submission**

1. Program Specification/Correctness (25 points)
    a. No errors, program always works correctly and meets the specification(s).
    b. The code could be reused as a whole or each routine could be reused.
    c. Design is reusable and extensible.
    d. Interfaces and abstract classes are applied for both unit 5 and 6.
    e. DB schema is normalized.
    f. Database classes are setup in a way so the SQL is read from a text file. (Improves modifiability).
    g. Code is adequately tested and test runs are shown for both Unit 5 and Unit 6.

2. Readibility(5)
    a. No errors, code is clean, understandable, and well-organized.
    b. Code has been packaged and authored based on Java Coding Standards.

3. Documentation(5)
    a. The documentation is well written and clearly explains what the code is accomplishing and how.
    b. Detailed class diagram is provided.

4. Code Efficiency(10)
    a. No errors, code uses the best approach in every case. The code is extremely efficient without sacrificing readability and understanding.

Reclaiming lost points

If you lost points in Units 1 through 4 you should:

1. Modify your project and fix issues that were reported to you (for which points were taken off).
2. Create a change log that shows:
    a. What you were asked to change
    b. How many points were taken off.
    c. What changes you made in what files
    d. Show test cases for changes made (if applicable)
    e. How many points should be added back in your opinons.
3. TA’s will review this change log, grade and return the lost points, if they feel that issue(s) have been corrected.

Things I learned that I would apply in 2nd Mini. (Due by March 3rd 2014) – 50 points

Please create a well-organized list of design lessons learned from Project 1 and submit to cislabs04@gmail.com by March 3rd 2014.

You will earn one point for each item. (Ofcourse content quality is important in this context).