Matthew Lonis

Jeff Whitmer

November 28th, 2016

CSCI-A 290

Final Project – Part 2b

This application will be a “master” application that will not only include A290 buffet and certain parts of the RPN calculator that we worked on in class, but it will also showcase three other sub-forms that I will create from scratch. I want this to be a “master” application that shows off multiple things so I can use it on a resume and get the most out of one application.

**1. Overall**

This section will describe overall elements that will be common to all form pages so I reduce redundancy going forward.

Every form and sub-form will implement an exit (or close) functions that will close the form correctly. Almost all forms and sub-forms will need to handle button click events. For all forms, I will attempt to implement a sound that plays when a button is clicked to add to a more immersive experience within the app. Windows, by defaults, contains .wav files that are used throughout Windows 10 in the C://Windows/media/ folder. I will attempt to implement this, however, if the process becomes too difficult I will discard this idea in the final project. All forms (other than the options form) will maintain the same property colors, style, etc. for all buttons, forms, and various other objects. I will use flat themed buttons and all forms will include the exit, maximize, and minimize buttons in the top right area of the form.

The options form will be modal form that cannot be closed unless the user clicks the close or okay buttons. In the options form, if the background is changed, the background for all forms will be changed. The main page will group all its navigation buttons near the top right. All other forms will maintain this design, specifically speaking, the close button will always be top right of the form. The only way I will change this is if a form is designed with a vertical design, in which case the close button will be at the bottom of the form. All child forms will appear center of the parent form to maintain fluid navigation. All text boxes will have a label placed near the text box as to show an obvious relation.

**2. Main Screen**

Since the main screen will be a source for further navigation within my app, there are no functions or methods that need to be deployed. The main form will need to handle button click events. These events, beyond the close button, will need to create child forms that will appear on top of the main form and link the parent form to the child form. All sub-forms opened from the main form will be able to reference the parent form and call on its variables and properties. The main form will include a custom icon that will be the same for all sub-forms. All buttons will be grouped towards the top right. There will be a close, A290 Buffet, calculator, fuel mileage calculator, tire size calculator, unit converter, and options button; in that order.

**3. A290 Buffet**

Will be identical to A290 Buffet.

**4. Calculator**

The calculator will include the same mathematical functions as RPN calculator, however I will be adding a square root function/button and a squared function/button. The calculator will handle the same events as RPN (including but not limited to: text box text change [to check if input is a number], button clicks) but will also encompass button clicks for the two new buttons I am adding. The design of my calculator will be a vertical design.

**5. Fuel Mileage Calculator**

The fuel mileage calculator will need two methods. The first method will be to determine fuel mileage based on given distance and fuel consumed inputs. The second method will be to determine cost of a trip by multiplying fuel mileage by inputted cost of fuel. I want to create a vehicle object that will store the vehicle make, model, year, distance, and fuel amount and have the associated above methods with this object. This form will need to handle text box changes (to detect if input is a number), button clicks, and form changes (when fuel mileage is calculated I want a label to update and say “Your [vehicle year] [vehicle make] [vehicle model] consumes [fuel mileage] miles per gallon.”

All text boxes will have appropriate labels to identify the purpose of the text box and will be near of one another. I will use a two column vertical approach where the left column is the text boxes and various other user interactions and the right one will be for navigation buttons and a calculate button.

**6. Tire Size Calculator**

The tire size calculator will need five methods for each tire object that will calculate tire diameter, width, sidewall length, circumference, and revolutions per mile. This form will also need a function to compare the sizes of two tires (where the arguments are two tire objects). The tire object I wish to create will include the three given number for a tire (for example if you have input for a tire that is 285/75R16, there will be three variables to store these three numbers separately) and include the associated five methods I described above.

The form will need to be able to handle button click events, user text box changes (to detect if input is a number), and a form that opens a child form that is a tire comparison form. The form will be designed with a vertical layout, much like the calculator, where user inputs numbers at the top, a calculate button is below that and finally, five labels each representing diameter, width, sidewall, circumference, and revs/mile to display the results of the calculation.

For the tire comparison form, it will be identical to the tire size calculator form, however it will take two tire size inputs and the results will show tire size 1, tire size 2, and the difference between the tires in three columns (see part 2a for a visual explanation).

**7. Unit Converter Calculator**

The unit converter will need functions that convert one number to another number based on given strings. There will be two strings given to this function. One string represents the current unit for the given number and the second string will represent the desired unit conversion. The unit converter will need to handle several events. One event will be a drop down menu for a user to select a given set of unit types (using a combo box). Another event will be for the text box changes to detect if given input was a number. Again, this form will be a vertical design much like tire size calculator and calculator forms.

**8. Options**

This will be very similar to the options page in the A290 Buffet where the user will be able to change the background color (using a group box), enable or disable the close on exit dialog (using a check box), and potentially other options. This form will be a special form page that doesn’t have a maximize or minimize button and cannot be closed by the user unless the user selects the ‘okay’ or ‘close’ button.