Lab1.htm; August 14, 2010

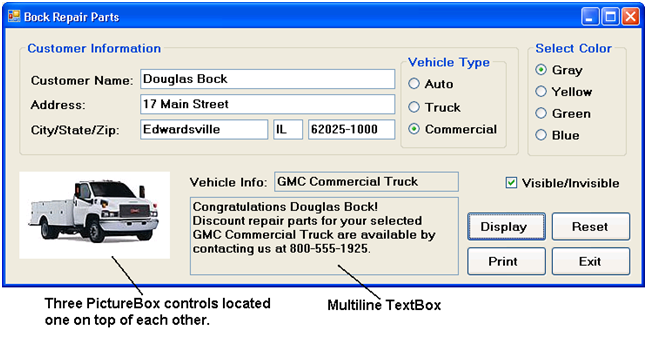
**Bock Repair Parts Sales Lab 2**

In this lab assignment you must develop a computer application to collect and display information about customers and products for the Bock Repair Parts Sales, LLC.

**Getting Started.**

* Start Visual Studio.
* Click the Create Project hyperlink or use the File-New Project menu option.
* In the New Project dialog box type a name for the project—name the project **Lab2-SectionTime-YourLastName-YourFirstName**– for example, if your name is Doug Bock and you are in the 2:00 pm class section, name the project **Lab2-200-Bock-Doug**.
* Change the form’s **File Name** property from **Form1.vb** to **Lab2.vb**.
* Click the Save All button on the button toolbar. The Save Project dialog box will display. In the Location drop-down combo box save the project to the **My Documents** folder.
* Continue your work on the project. Periodically you should click the Save All button so you don’t lose any work.

**Design Requirements.** Develop a form that is similar to the one shown below. Your form may have minor differences. The form enables the application user to enter information about customers and to select the type of vehicle and color to use when displaying information. There are two output TextBox controls and three PictureBox controls.



**Form Design.**

* Name all controls following the naming standard taught during class meetings.
* Set the appropriate property to display the form's title bar as shown in the figure to display the words **Bock Repair Parts** along with your name, for example: **Bock Repair Parts - by John Smith**.
* Set the appropriate property to cause the form to startup centered in the display monitor.

**Customer Information GroupBox.**

* Add a GroupBox control as shown. Set the appropriate property so that the GroupBox displays the words **Customer Information** as shown in the figure.
* Add three Label controls as shown for use as prompts to display the words: **Customer** **Name:**, **Address:**, and **City/State/Zip:**.
* Add TextBox controls to enable the application user to enter the customer name, address, city, and state. Give each TextBox an appropriate name.
* Add a MaskedTextBox control so the application user can enter the zip code. Give the masked textbox an appropriate name. Set the appropriate property for the MaskedTextBox control so that the control displays a 9-digit zip code mask.
* Add a label that displays the words **Vehicle Info:** and two read-only TextBox controls as shown – these TextBox controls will display output as described later in this document. Give each TextBox an appropriate name.
  + Set the appropriate property so that each TextBox can only be used to display output – the application user cannot type into either TextBox.
  + Set the appropriate property so that the Tab key does not tab to either of these TextBox controls.
  + Set the appropriate property of the second TextBox so that it can display multiple lines of output (identified as a Multiline TextBox in the figure).
* Add a CheckBox control as shown. The control should be checked when the project initially starts up.

**Vehicle Type GroupBox.**

* Add a GroupBox control as shown inside of the Customer Information GroupBox.
* Set the appropriate property so that the GroupBox displays the words **Vehicle Type** as shown in the figure.
* Add three RadioButton controls in this GroupBox and set the appropriate property to display the words **Auto**, **Truck** and **Commercial**.
* The auto RadioButton control should be the checked as the default on form startup.
* Selecting one of these RadioButton controls should cause two things to happen:
  + Display information about the selected vehicle type in the vehicle info TextBox.
  + Display a picture of the selected vehicle in a PictureBox control.
* The information to display in the read-only vehicle info TextBox control and PictureBox controls is given in this table.

|  |  |  |
| --- | --- | --- |
| **Radio Button** | Automobile Information | Graphic File Name |
| Auto | Ford Escort sedan | FordAuto.jpg |
| Truck | Dodge Truck | DodgeTruck.jpg |
| Commercial | GMC Commercial Truck | Commercial.jpg |

**Color GroupBox.**

* Add a GroupBox control and set the appropriate property so that the GroupBox displays the word **Select Color** as shown in the figure.
* Add four RadioButton controls to the GroupBox as shown.
* Set the appropriate property so that on startup, the Gray RadioButton is the default.
* Give each RadioButton an appropriate name.

**Buttons.**

* Add four Button controls as shown.
* Give each Button control an appropriate name.
* Set the appropriate property so that the Button controls display the text with hot keys – you select the hot keys to use.

**PictureBox controls.**

* Add three PictureBox controls and name each control appropriately. Each control will display a single image as described in the table above.
* Set the properties necessary to display the appropriate graphic image in each PictureBox control. The graphic images are named as described in the above table and are located on server **Y:\Business\Departments\CMIS\bockd\CMIS142\Graphics.**
* Set the appropriate property so that each graphical image displays in and fills the picture box control to a size approximately the same size as that shown in the figure below.
* Finish the design by drag/dropping the PictureBox controls on top of one another so that only one control is visible at design time – which control is visible at design time does not matter.

**Other Form Design Requirements**.

* Set the appropriate property to map the keyboard **Enter** key to the **Display** button control.
* Set the appropriate property to map the keyboard **Esc** key to the **Reset** button control.
* Add a tool tip component control to the form. Add the following tool tips for the four button controls:
  + Display button control – **Display congratulations message**.
  + Reset button control – **Reset the form**.
  + Print button control – **Print the form**.
  + Exit button control – **Exit the application**.
* Set the tab order as appropriate. The application should not tab to the output TextBox controls.
* Controls must be aligned and centered properly, proper amount of gray space around controls, and no misspellings.

**Coding Requirements.**

* **Display Button Coding**. When the Display Button is clicked, display a congratulations message to the read-only TextBox as shown in the figure.
  + The message should concatenate the customer name TextBox control’s Text property value and the vehicle info TextBox control’s Text property value into the message as shown in the example in the figure.
  + Display the message on separate lines so as to provide a pleasing appearance.
* **Reset Button Coding.** Write coding statements to accomplish the following tasks when the application user clicks the Reset Button:
* Clear all input and output TextBox and MaskedTextBox controls (except the vehicle info TextBox control).
* Check the gray RadioButton.
* Check the auto RadioButton (this will cause the vehicle info TextBox control to display the message from the table given earlier to the TextBox).
* Check the visible/invisible CheckBox.
* Set focus to the customer name TextBox control.
* **Print Button Coding.** Write code to print the form to a print preview window.
* **Exit Button Coding.** Write the code needed to exit the application.
* **CheckBox Coding.**  When the visible/invisible CheckBox control is unchecked, the vehicle info Label, vehicle info TextBox, and congratulations TextBox controls should become invisible. When checked, these three controls should be visible. This control should be checked when the application starts up.
* **RadioButton Coding – Vehicle Type.** As explained earlier, display the appropriate vehicle information to the vehicle info TextBox and the appropriate PictureBox image depending on the RadioButton that is checked.
* **RadioButton Coding – Select Color.** When one of the RadioButtons is checked, change the form’s background color. Write additional code using the Form’s Load event and a module-level variable to enable restoration of the gray background color.
* **Remarks.** Add remarks to the top of the program that gives the program name (class name), programmer name (you), and date programmed. Add remarks to each sub procedure as appropriate.

**Test the Lab.** Test your lab to ensure that it works correctly. Use the assessment grading form shown below—it is the same form that will be used to grade your project.

**CAUTION: DO NOT run your project from a flash drive – this can cause the project to become corrupted and you will have to recreate the project – always copy the project to the drive C:\TEMP folder or to My Documents and test run your project.**

**What to Do When You’re Finished, How to Save and Submit the Lab.**

Ok, you’ve finished the lab. **Do NOT use the File menu, Save As option**. Instead, follow these steps.

* First, CLOSE Visual Studio—you cannot copy the project to another location if it is open.
* Locate the folder that contains the project. It should be located in the **My Documents** folder where you first saved the project.
* Copy the entire folder to a flash drive.
* Bring the flash drive to the University to either a computer lab or to the computer classroom. Copy the entire folder to drive Y: to the submission folder for your class. You can copy the folder to drive Y: at the beginning of the class period when the project is due.

**Visual Basic Project Assessment Plan – Lab 2 (25 points possible).**

**Before startup.**

* Submitted late – see the course syllabus for a description of the late penalty.
* Submitted on time but some of the files necessary to run the project are missing – you must resubmit the project, see your instructor if you need assistance submitting the project – your resubmission will be considered a late submission.
* Project should be named correctly: **Lab2-SectionTime-YourLastName-YourFirstName**.
* Form’s File Name property is correct (Lab2.vb, NOT Form1.vb).
* Form’s Title bar displays correct information.
* Form has a good appearance: controls aligned, control size appropriate, no misspelled words, not too much gray space around controls.
* Program code has the required remarks statements to identify the program, programmer name, and date programmed.
* Each sub procedure has remarks statements to identify what the sub procedure does.
* All form controls are named properly following the naming convention taught in the notes and in class.
* Delete empty sub procedures that have no executable code that were created accidentally.
* The form has only a single ToolTip control (only one ToolTip control is needed for each form).
* Each GroupBox control displays the appropriate Text value.
* The form's Display Button control is mapped to the keyboard Enter key.
* The form's Reset Button control is mapped to the keyboard Esc key.

**Startup.**

* Form starts up centered on the screen. Form is an appropriate size.
* The Auto RadioButton control is checked.
* The Gray RadioButton control is checked.
* The Visible/Invisible CheckBox control is checked.
* The PictureBox control for the Auto displays on startup.
* Information for the Ford Escort sedan displays in the vehicle info read-only TextBox controls on startup.
* The tab order for the form is correct.
* The Congratulations TextBox control is read-only and is skipped in the tab order.
* The controls in the Customer Information GroupBox are blank (empty) at startup.
* The Zip MaskedTextBox control has the correct Mask property setting.
* Tool tips for the Button controls display correctly.
* When the Alt key is pressed, all hot keys are underlined for the Button controls.

**Display Button Click Event.**

* The output TextBox control displays the congratulations message as shown in the figure (-3 if does not work).

**Reset Button Click Event.**

* Clears TextBox and MaskedTextBox controls (except for the vehicle info TextBox control that always display data).
* Checks the Auto RadioButton and Gray RadioButton controls
* Checks the Visible/Invisible CheckBox controls.
* Set focus to the customer name TextBox control.

**RadioButton CheckedChanged Events – Select Color.**

* Sets the form’s background color to the selected color (-1 for each RadioButton that does not work properly; maximum of -2 points).
* Selecting the Gray RadioButton control restores the background color to the default shade of gray assigned normally to a Form control by using the techniques specified in the notes (-2 if does not work properly).

**RadioButton CheckedChanged Events – Vehicle Type.**

* Displays correct selected vehicle info in the TextBox (-2 if does not work).
* Displays correct automobile image (-2 if does not work).

**CheckBox CheckedChanged Event.**

* When checked, vehicle info Label, vehicle info TextBox and congratulations TextBox controls display.
* When unchecked, vehicle info Label, vehicle info TextBox and congratulations TextBox controls are not visible.

**Print Button Click Event.**

* Displays form in a print preview window.

**Exit Button Click Event.**

* Exits the program correctly.

**Other Errors.**

* During program execution, your computer program should compile and execute without generating any error messages – if the program will not compile or generates error messages during execution, you will lose up to 20 points depending on the severity of the error.
* If the program will not compile so that it can be executed, automatic deduction of 10 points regardless of the error.

**End of Lab**