|  |  |  |
| --- | --- | --- |
| **CS 1400 Lab #14**  **RadioButtons and ComboBoxes**  **Version 1.0**  **Objectives:**  In a program using a Graphical User Interface (GUI), it is common to offer the user a list of choices and allow them to select one of the choices from a group of ***RadioButtons*** and/or a ***ComboBox*** and other multi-item lists. In this Lab you will learn how to do this with ***Radio Buttons*** and a ***ComboBox***.  **Study Material**   |  | | --- | | **RadioButtons**  ***RadioButtons*** allow the user to choose one item from a ***GroupBox*** of several choices. When you select a ***RadioButton***, any previously selected Button becomes un-selected. This means that the choices are mutually exclusive, i.e. you can only choose one ***RadioButton*** from the **GroupBox**, and all others are then deselected. If you want to be able to choose from several items from a ***GroupBox*** control provided by Visual Studio to do this.    Figure - Single Radio Button  The Fig. 1 shows a single ***RadioButton*** on a Form. A ***RadioButton*** appears as a small circle with a label to the right of the circle.  The text usually explains what the ***RadioButton*** represents. It is stored in the Text property of the ***RadioButton***. When a ***RadioButton*** is selected, a small dot    appears inside of the circle.    ***RadioButtons*** almost always appear in a group. A group is defined by whatever enclosing control holds the ***RadioButtons***. In this case I used a ***GroupBox***. This could also be the Form, a Panel or a ***GroupBox*** control. Fig. 2 shows a group of three ***RadioButtons*** inside of a ***GroupBox***. The ***GroupBox*** provides a Text property that helps describe what the ***RadioButtons*** are to be used for. In this case the Text property of the ***GroupBox*** has been changed to ***"Shipping Methods"***. The middle ***RadioButton*** has been selected.   Figure - GroupBox Control containing RadioButtons  If you have a RadioButton named ***RBtnSameDay*** you can get the text of the selected item by writing  string selectedItem = RBtnSameDay.Text;  You can test a ***RadioButton*** to see whether or not it has been selected (Checked) by looking at its ***Checked property***. If the RadioButton is the currently selected RadioButton in the GroupBox (GBoxShippingRates), its Checked property will be true. Otherwise it will be false. For example,  ***if (RBtnExpress.Checked) { ... }***  You can set the Checked property of a RadioButton by writing  ***RBtnExpres.Checked = true;***  **Handling the CheckedChanged Event**    Double click to change to true  When you select or de-select a RadioButton, a ***CheckedChanged*** event is fired. Create the skeleton for a ***CheckedChanged*** event handler by selecting a RadioButton. Then in the Properties Window click on the lightning bolt to see a list of possible events. Double click on ***CheckChanged***.  Here is an example of a ***CheckedChanged*** event handler:  private void ***RBtnStandard\_CheckedChanged***(object sender, EventArgs e) {     if (RBtnStandard.Checked)        MessageBox.Show("Selected Standard Shipping"); } | | **ComboBoxes**    ***ComboBoxes*** are similar in some functionality to RadioButtons, as they allow the user to choose one item from a dropdown ***ListBox*** of several choices. When you select one item, any previously selected item becomes un-selected. Only one item in the ComboBox list may be selected. A ComboBox is a  Figure - ComboBox Control Text Property Displayed  combination of a TextBox and a dropdown ListBox controls. Only one list item is displayed at one time in a ComboBox and other available items are loaded in a dropdown ListBox.  The Fig. 3 shows a ComboBox on a Form. The text shown in the TextBox is initialized in the Text property of the ComboBox.  The items in the Dropdown List are defined in the ***Items Property*** as shown in Fig. 4 a list for a ComboBox.   If you have a ComboBox named ComboBox1 you can get the text of the selected item by writing  ***string itemText = CBoxShippingRates.Text;***  You can also get the index into the list of selections by writing  Figure - ComboBox Dropdown ListBox  ***int selectedItem = CBoxShippingRates.SelectedIndex;***  The index of the selected item is stored in the ***SelectedIndex*** property of the ***ComboBox***.  **Handling the SelectedIndexChanged Event**    Figure - Inserting a SelectedIndexChanged Event Handler Method  When you select one of the items in a ***ComboBox***, that item appears in the ***TextBox*** part of the ComboBox and a ***SelectedIndexChanged*** event is generated. Create the skeleton for a SelectedIndexChanged event handler by selecting the ComboBox. Then in the Properties Window click on the lightning bolt to see a list of possible events. Double click on SelectedIndexChanged.  Here is an example of a SelectedIndexChanged event handler:  private void CBoxShippingRates\_SelectedIndexChanged(object sender, EventArgs e) {     string selectedItem = CBoxShippingRates.Text;     MessageBox.Show("You selected " + selectedItem); }  **Using a RichTextBox Control:**  A RichTextBox is used to display or enter text. It is often used to display the results of a user’s interaction with the GUI. For instance the results of clicking on RadioButton or selecting an item from a ComboBox.  The RichTextBox is |   **Programming Exercise** |
| **CS 1400 Lab 14**  **How would you like it Shipped?**  **Objective:**  This lab will give you some practice using RadioButtons and ComboBoxes.  **The Problem**  In this week’s programming Project, the user will be able to select one of a number of shipping methods for a parcel. Parcels may be shipped using Standard, Express, or Same-Day service. For this Lab, create a usable GUI Windows Application that uses both RadioButtons and a ComboBox to offer the user this set of choices. When the user makes a selection, show which choice was made in a MessageBox.  **File(s) to Submit:**  Place the complete project folder for this assignment in a zip file and name the zip file Lab\_14\_your-initials\_v1.0.zip. For example, I would name my file Lab\_14\_DAF\_V1.0.zip. Submit this assignment as Lab #14 on Canvas.  **Grading Checklist**   |  |  |  |  | | --- | --- | --- | --- | | # | Program | 1st Submission | 2nd Submission | | 1 | Meets & works to specifications | Correct | Xnot | 6 points | | 2 | Error Free, elegant & efficient | C | X | 4 points | | 3 | Pseudo-Code | C | X | -3 points | | 4 | Style Guidelines | C | X | -2 points | | 6 | Source Files(s) & Formatting | C | X | -2 points | | 7 | Project Prolog | C | X | -1 points | | 8 | Function Prologs | C | X | -1 points | | 9 | Zip Filename | C | X | -1 points | | 10 | Lab & Project Names | C | X | -1 points | | 11 | Zip File is invalid or will not unzip | Lab = 0 pts | Lab = 0 pts | |  | Total Points | 10 | 0 | 10 | 0-9 | |