**Designing File Menu**

**Setting Up the User Interface**

Open Visual Studio and create a new Windows Forms Application project.

Design a basic form with the following elements:

**Adding a TextBox**

1.**Toolbox:** Locate the Toolbox in Visual Studio. It usually appears on the left-hand side of your screen.

2.**Search:** If it's not expanded, find the "Common Controls" section in the Toolbox. You can either scroll down or use the search bar to find the "TextBox" control.

3.**Drag and Drop:** Click and drag the TextBox control from the toolbox and drop it onto your form.

4.**Resize:** Resize the TextBox to occupy a large portion of your form. You can do this by clicking and dragging the little squares on its edges.

5.**Properties:**

Go to the "Properties" window (usually located on the right-hand side).

Find the property named "Multiline" and set it to "True". This is essential for allowing multiple lines of text.

Optionally, you can change the "Font" property to make the text more readable.

**Adding a MenuStrip**

1. **Toolbox**: Locate and expand the "Menus & Toolbars" section in the Toolbox.
2. **Drag and Drop**: Drag the "MenuStrip" control onto your form. It will usually dock itself at the top of the form.
3. **Creating "File" Menu**:
   * Click on the MenuStrip component where it says "Type Here".
   * Type "File" and press Enter to create a top-level "File" menu item.
4. **Creating Submenu Items**:
   * Click below the "File" item, where it says "Type Here" again.
   * Type "Open" and press Enter.
   * Repeat the process and type "Save".

**Implementing File Open Functionality**

1. Double-click on the "Open" menu item to create its event handler code.
2. Inside the event handler code, add the following logic:

Imports System.IO ' Add this line on the top of the form

Dim openDialog As New OpenFileDialog()

openDialog.Filter = "Text Files (\\*.txt)|\\*.txt|All Files (\\*.\\*)|\\*.\\*" ' Filter for text files

If openDialog.ShowDialog() = DialogResult.OK Then

Dim filePath As String = openDialog.FileName

Dim fileContents As String = My.Computer.FileSystem.ReadAllText(filePath)

Textbox1.Text = fileContents 'Put the file contents in our TextBox

End If

**Code Breakdown:**

* We create a new OpenFileDialog object.
* We set a filter (\*.txt) to allow users to see only text files by default (optional).
* We call ShowDialog() to display the dialog box.
* If the user clicks "OK" and selects a file, the code retrieves the selected file path.
* We use My.Computer.FileSystem.ReadAllText to read the entire text file content into a string variable.
* Finally, we update the Text property of our TextBox control to display the file contents.

**Implementing File Save Functionality**

1. Double-click on the "Save" menu item to create its event handler code.
2. Inside the event handler code, add the following logic:

Dim saveDialog As New SaveFileDialog()

saveDialog.Filter = "Text Files (\*.txt)|\*.txt|All Files (\*.\*)|\*.\*"

If saveDialog.ShowDialog() = DialogResult.OK Then

Dim filePath As String = saveDialog.FileName

File.WriteAllText(filePath, TextBox1.Text)

End If

**Code Breakdown:**

* We create a new SaveFileDialog object.
* We set a filter (\*.txt) to allow users to see only text files by default (optional).
* We call ShowDialog() to display the dialog box.
* If the user clicks "OK" and enters a filename, the code retrieves the specified file path.
* We use My.Computer.FileSystem.WriteAllText to write the current text content of our TextBox control (everything the user has typed in) to the specified file path.

**Running and Testing the Application**

1. Build and run the application in Visual Studio.
2. Test the "Open" functionality by opening a text file. Verify that the file contents are displayed in the TextBox.
3. Test the "Save" functionality by typing some text in the TextBox and then using "Save" to save it to a new file.