**MenuStrips Using Multiple Document Interface (MDI)**

Many applications require multiple forms. The MDI application is a multiple document interface application. This type of Windows application has a parent form and one or more child forms. The parent form is a container for the child forms.

A child form functions completely within the parent form and cannot be moved outside the parent form. The child windows can be tiled, cascaded, and dragged to arrange them within the parent form.

**Creating and MDI Application**

First set the appropriate Form1 properties to create a parent form. These Form control properties are:

* **IsMdiContainer** is set to True for a parent form
* **WindowState** can be normal, minimized or maximized. A parent form is typically set to maximized to fill the entire screen.

A parent form should also include menus and commands for controlling the child forms.

Next a form is added to the project. Adding an new form to a project adds a form class, which means that objects of that class to be instantiated in the Form1 module. In other words, statements can be added to the Form1 module, in this case the parent form, to create and display child forms using the new form class. To add a form to a project, select **Project 🡪 Add Windows Form**.

Type a descriptive form name in the Name box (be sure to keep the .vb extension) and ten select **Add**. For this application, the MyTextEditor application, NewDocument is the child form name because it will contain a document. A Design window for the new form is displayed in the IDE.

The child form can be modified just as any form in the Design windo. Code specific to the child form is added to the application by switching to the child’s code window.

**Using Child Forms**

Program code for a parent form includes statements that declare a child form, display the child form, and close the child form. For example, the following Form1 (parent form) statements instantiate a new form, make the new form a child by assigning it to the parent, and then display the child form:

* Dim childForm As New NewDocument () ‘instantiate a new form
* childForm.MdiParent = Me ‘assign a parent
* childForm.Show() ‘display the child form

The first statement above declares a variable named childForm using NewDocument as the type. The NewDocument form added to the project is a class, which can be used to instantiate a new form. Next, the Form control has the property MdiParent, which can be set at run time only. The property is set to Me to assign the child form to the current form. Form1, which has been designated as the parent form. In the last statement, the Show() method is used to display the child form.

When no longer needed, a child window should be removed from the MDI application interface. The form control includes the following method:

* **Close()** closes the form. The form cannot be reopened after closing a new one must be instantiated and then displayed.

The close() method can be called from the child code or from the parent code by first determining the active child. To close the active child, the ActiveMdiChild property for the parent form is used. For example, the following statement closes the active child window:

* Me.ActiveMdiChild.Close() ‘parent form closes active child

**TextEditor Assignment Part 1 of 5**

1. **Create a New Project**: create a Windows application named TextEditor.
2. **Create the Parent Form**
   1. Select the form and set the Text property to Text Editor, the IsMdiContainer to True, and WindowState to Maximized.
   2. Add a MenuStrip to the form, name the first MenuItem File, the second MenuItem New, the third MenuItem Close and the last MenuItem Exit.
3. **Add a Child Form** 
   1. Select Project 🡪Add Windows Form. A dialog box is displayed.
      1. In the Template list, click Windows Form, if it is not already selected.
      2. In the Name box, type NewDocument. Vb.
      3. Select Add. A form named New Document.vb is displayed in a Design window. Note the file added to the project as shown in the Solution Explorer window.
   2. Set the Text property of the NewDocument form to New Document.
4. **Write the Application Code**
   1. Display the Form1 Design window and then display th eForm1Code window.
   2. Add comments that include your name and today’s date.
   3. Create a NewToolStripMenuItem\_Click event procedure and then add statements to display a new document window with an appropriate title bar:

* Static docNum As Integer = 0 ‘document number
* Dim doc As New NewDocument ‘create new document window
* ‘Assign document parent
* doc.MdiParent = Me
* ‘create title bar that includes document number
* doc.Text = “New Document “ & docNum
* ‘Show Window
* Doc.Show()
  1. Create a CloseToolStripMenuItem\_Click event procedure and then add a statement to close the active window:

Me.ActiveMdiChild.Close()

* 1. Create a ExitToolStripMenuItem\_Click event procedure and than add a statement to end the application:

then add a statement to end the application:

Application.Exit()

1. **Run the Application**
   1. Save the modified TextEditor project and then run the application.
   2. Select File 🡪 New. A New Document 1 Window is displayed. Drag the window in the TextEditor window to move it.
   3. Select File 🡪 New. A new Document 2 window is displayed.
   4. Click the New Document 1 window to make it the active window. Select File🡪Close. The window is closed.
   5. Close the New Document 2 Window.
   6. Select File 🡪 Exit to quit the TextEditor application.