Lab 5 Due next lab period – week of May 23-26

Purpose:

1. Create a Windows GUI application using Visual Studio and VB.NET
2. Learn UI control event handling for `RichTextBox` and `WebBrowser` controls
3. Practise `VB.NET` programming; implement the Try/Catch block

What to submit

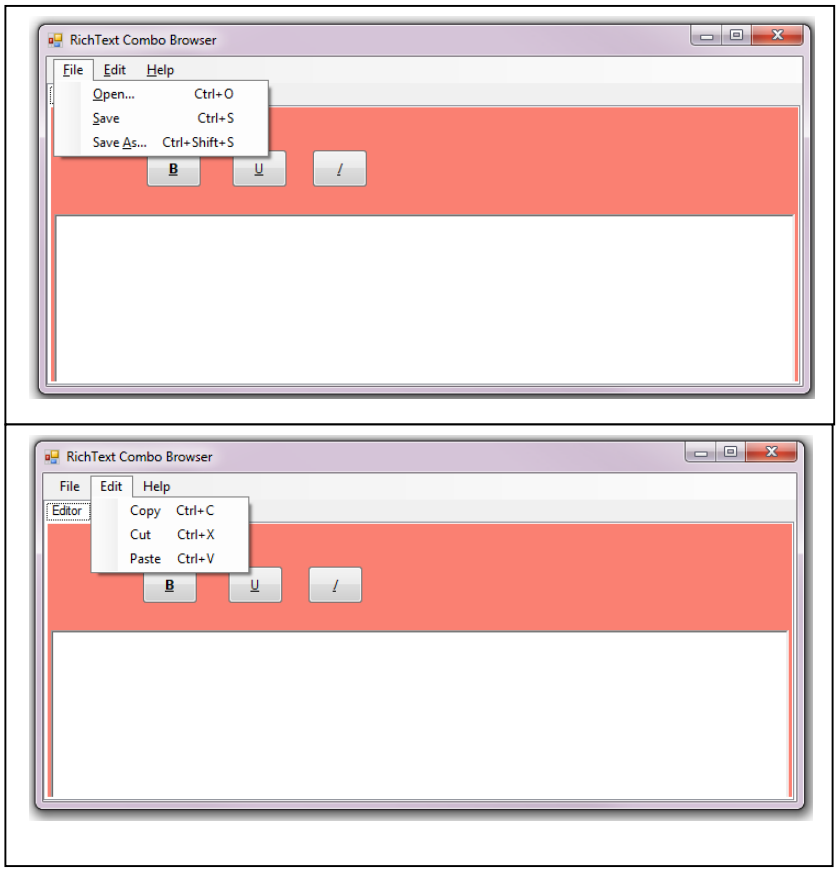
1. Demo will be done in lab, in your lab section.
2. Upload your project compressed into a .zip file to the Dropbox on D2L.

Evaluation (24 marks):

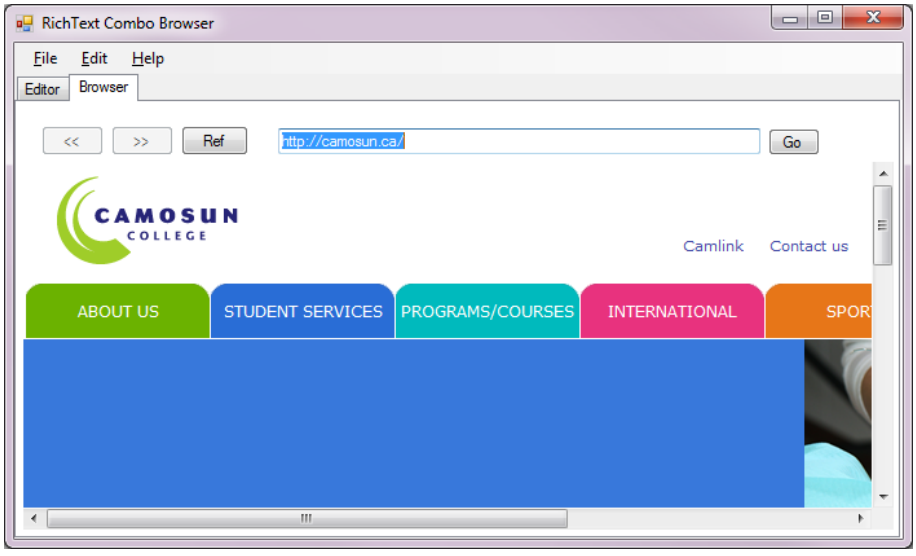
1. Be prepared to demonstrate your solution at the start of next lab. If you are not ready at the start of lab or are not present, a mark of 0 will be given.
2. Your solution will be assessed based on (3 marks each):
    - proper form layout of the controls (no overlapping controls, sufficient white space)
    - coding the editor buttons
    - coding the menu bar operations
    - coding the tab control
    - coding the web browser control
    - coding the associated web browser buttons
    - VB.NET code style – variables named appropriately, comments used in the code where needed, code is readable and understandable
    - Correctly answering questions about your solution.
3. **All work must be individual.** Any plagiarism will result in a mark of 0 and possibly additional penalties.

Description

The requirement for this lab is a Windows form GUI application which contains a simple rich-text editor and a functioning web browser. The user can switch between the two components using a `TabControl`. The `RichText` editor is contained in the first tab and the browser is in the second. The menu bar shown is also to be implemented and is shown below:



The browser tab:

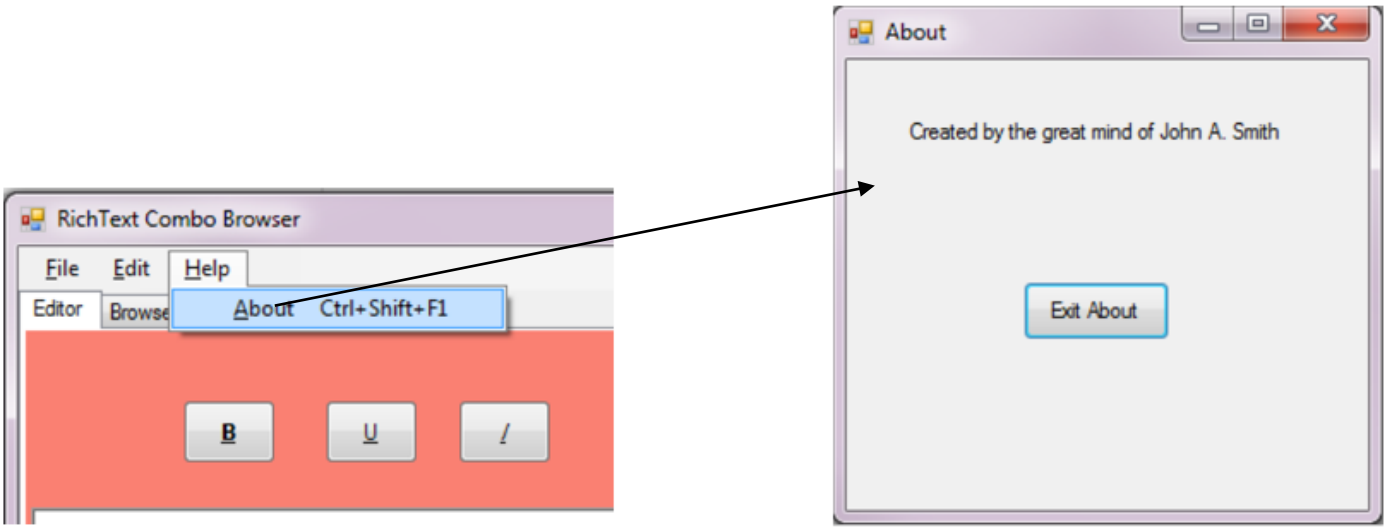


The editor component uses the `RichTextBox` control, which is similar to the `TextBox` control but supports rich-text formatting such as different fonts and styles, as well as hyperlinked text.

Above the editor window appear three style buttons: B for bold, I for italics, and U for underlining. When the user selects any text in the `RichTextBox` control, that text can be rendered into bold, italics and/or underline by clicking on the corresponding style button. The buttons should have no effect if no text has been selected. You might expect that clicking one of the buttons makes subsequent typed text appear in that style as you type but this requirement is not expected here – so note the difference in the UI behaviour compared to other word processors. Note that the styles are not mutually exclusive (e.g. text can be bolded and underlined). And once styled, there is no provision for unstyling the text.

Also, the rich text editor should respect the key shortcuts for bold (control-b), italics (control-i), and underline (control-u), which function similarly to clicking the buttons.

The Help | About menu brings up a second, smaller sized Form control having a label and a button (Exit About). Place your name in the label text. Use the Form control's `Show` and `Hide` methods appropriately in the VB code. Set the `TopMost` property of the About form to `True` to make it always appear on top of other form



The web browser component uses the `WebBrowser` control. Above the browser is a set of simple navigation buttons ("back", "forward", "refresh", "go"), and a textbox for the entry of a new URL to load into the browser.

When the user hits the Enter key from within the URL textbox, or when the user clicks the "go" button (to the right of the textbox—not shown in the above image), a new URL is loaded into the browser based on the text in the URL textbox. If the user neglects to provide the protocol portion of the URL, the text should be defaulted to add in the http:// protocol portion into the string. Only the http protocol need be supported here.

Select a starting home URL as http://www.camosun.ca for the browser control.

The menu bar component has three top level items: File, Edit and Help.

In the File menu there are three submenu items: Open, Save and Save As.

The Open action is used by the editor to open an existing RTF file and is used by the browser to open an HTML file. If the user attempts to open an unsupported type file in either editor or browser, the application should display a help message to the user. For the browser, opening a file will use the file:// protocol instead of the http:// protocol, you should copy the file location to the URL input text field complete with the file:// protocol for the user to see.

The Save action is used only by the editor. If the user has previously saved or loaded the current file, then the contents are simply updated without prompting the user. If the user is working on a new file, then this menu item should function the same as the "Save As" menu option. When the user first opens a file, or immediately after the file is saved, the "Save" menu item should be "greyed-out", indicating that the user cannot update the file at this time.

The Save As action is used by both the editor and the browser. For the editor, the user is shown a standard `SaveDialogBox` instance. For the browser the user is shown the Explorer's `SaveAsHTML` dialog box. The user is prompted by either dialog box for a file name, and the current page is saved. If the file cannot be saved for some reason, a message box is displayed directing the user to try again. The user should be allowed to continue to try until they either succeed or cancel the operation by clicking the "Cancel" button in the dialog.

The Edit menu has three submenu items: Copy, Cut and Paste. These items should work as expected but only for the editor tab. If the browser tab is active, then these three items are greyed out in the menu.

The Help menu needs just one item: About. When the user selects this item, a second form appears as described above.

Menu hotkeys to be implemented include: Open – ctrl o; Save – ctrl s; Save As – ctrl shift s; Copy – ctrl c; Cut – ctrl x; Paste – ctrl v; About – ctrl shift F1

Alt keys need to be defined for activating the menu items: file = alt F; edit = alt E; help = alt H

The application should handle any Window resizing operations. The buttons, etc above the tabs should be fixed height and the horizontal centering should be maintained as the window becomes wider or narrower. The editor and browser components always occupy the remaining window space.

Bonus: Add the ability to undo edits in the rich text editor. You can handle undos of formatting edits only for this bonus, or if you're feeling adventurous, see if you can handle undos of text input too (this may be significantly harder). **Hint**: Try using a Stack.

Steps

If you are using the lab computers to complete this lab work, there is a Microsoft domain trust issue that will prevent the WebBrowser and FileDialog controls from working properly if your project is saved to your personal network student account (H:).

Instead, you can use these controls without fuss by making the VB project save the files onto the C: under your personal Documents folder, instead. Click Tools | Options | Projects and Solutions and change the three directory locations to the c:\ directory where your documents are saved locally - look for your student number. You may alternatively use your memory stick in place of the lab PC's local drive.

1. Add TabControl:

The `TabControl` is simple to operate. Drag the control from the Toolbox onto the form, activate one of the tabs, then drag new controls onto the active tab. `TabControl` uses the property `TabPages` collection to maintain each of the tabs' properties. The `SelectedIndexChanged` event fires when a tab is selected. Additional information at http://msdn.microsoft.com/en-us/library/4akwf92a(v=vs.100).aspx

2. Add RichTextBox to one of the Tabs
    a. You'll need to add:
        i. RichTextBox
        ii. Three Buttons

The `RichTextBox` control uses a number of events including `LoadFile`, `SaveFile`, `SelectionChanged` and properties `SelectionFont`. Below is sample `VB.NET` code for setting the underline property for the selected text within a `RichTextBox` control.

```
' Set the underlines style of the selected text.
Dim f As Font
Dim s As FontStyle

Dim i As Integer        ' Loop counter.
Dim start As Integer    ' Where the selected text starts.
Dim length As Integer   ' How many characters are in the selection.

start = Me.RichTextBox1.SelectionStart     ' Save this values so they can be restored later.
length = Me.RichTextBox1.SelectionLength

If length <> 0 Then

    For i = start To start + length - 1    ' Iterate through each character in the selection.

        ' Set the selected text to that character.

        Me.RichTextBox1.SelectionStart = i ~~~This is an i not 1
        Me.RichTextBox1.SelectionLength = 1

        ' Examine this character's font and style.

        f = Me.RichTextBox1.Font
        s = Me.RichTextBox1.SelectionFont.Style

        ' Switch on this character's underlines style.

        ' Bitwise OR operation to add the underlines "bit".
        s = s Or FontStyle.Underline

        Me.RichTextBox1.SelectionFont = New Font(f.FontFamily, f.Size, s)

    Next i

    ' Restore the original selected start and length.
    Me.RichTextBox1.SelectionStart = start
    Me.RichTextBox1.SelectionLength = length
    Me.RichTextBox1.Focus()
```

Set the Underline bit

Some VB code is missing here...

Additional information at http://msdn2.microsoft.com/en-us/library/3tdc88y7(VS.100).aspx

3. Add the WebBrowser
    a. You can use a SplitContainer in the Tab to hold the buttons above the WebBrowser

The `WebBrowser` control uses methods `GoBack`, `GoForward`, `Navigate`, and `Refresh`. Additional information at http://msdn2.microsoft.com/en-us/library/2te2y1x6(VS.100).aspx

4. Add the MenuStrip control events:

The `MenuStrip` control uses a `Click` event for each menu and submenu item. Properties include `Checked`, `ShortCut` and `Enabled`. Set the `OpenFileDialog1` object `Filter` property to "All Files|*.rtf|Text files|*.txt" so that those types of files can be seen by the file dialog. Set the `FileName` property to "Testdoc.rtf". Additional information at http://msdn2.microsoft.com/en-us/library/ms171650(VS.100).aspx

If you do not want to use the `LoadFile` and `SaveFile` methods of the `RichTextBox` (which are easy to use), you can use the Stream and `StreamReader` objects instead. The `OpenFileDialog` and `SaveFileDialog` will require the use of a `StreamReader` object which will hold the contents of the opened file. `StreamReader` class is under the `System.IO` Namespace so you will need to add **Imports System.IO** at the very top of your VB.NET source file above any class definition. Additional information at http://msdn2.microsoft.com/en-us/library/system.windows.forms.openfiledialog.aspx and http://msdn2.microsoft.com/en-us/library/system.windows.forms.savefiledialog.aspx