# Lab 5: Text Editor

Lab Assignment for NETD2202 – Net Development I

Prior to attempting this assignment you should…

1. Read Chapters 10, 21 and 25.
2. Attend Class
3. Watch Menus and Streams

This lab is to be completed individually.

Analyze the problem, code and test a solution. Submit your solution, no desk-check or Git URL is required.

**IMPORTANT** any submission submitted as anything other than a .zip file will be penalized 15%.

## Demonstrates

- Visual Studio (Ch. 1)
- Constants and Variables (Ch. 4)
- Use of controls (Ch. 2)
    - Setting Properties Statically
    - Setting Properties Dynamically
- Adherence to the Style Guide
- Events (Ch. 6)
- If/ElseIf (Ch. 5)
- Methods (Functions and Subroutines) (Ch. 6)
- System.IO (Ch. 21)
    - File.Exists
    - FileStream
    - StreamWriter
        - Write
    - StreamReader
        - ReadToEnd
- FileDialogs
    - SaveFileDialog
    - OpenFileDialog
- MenuStrip (Ch. 25)
    - ToolStripMenuItem

## Client Requirements

A client has contracted your company to create a text editor. Yes, you are asking yourself why another text editor, but the client is always right.

You will be creating a Windows Forms application that will need to be able to open, close, edit, save, save as, and create new files (.txt).

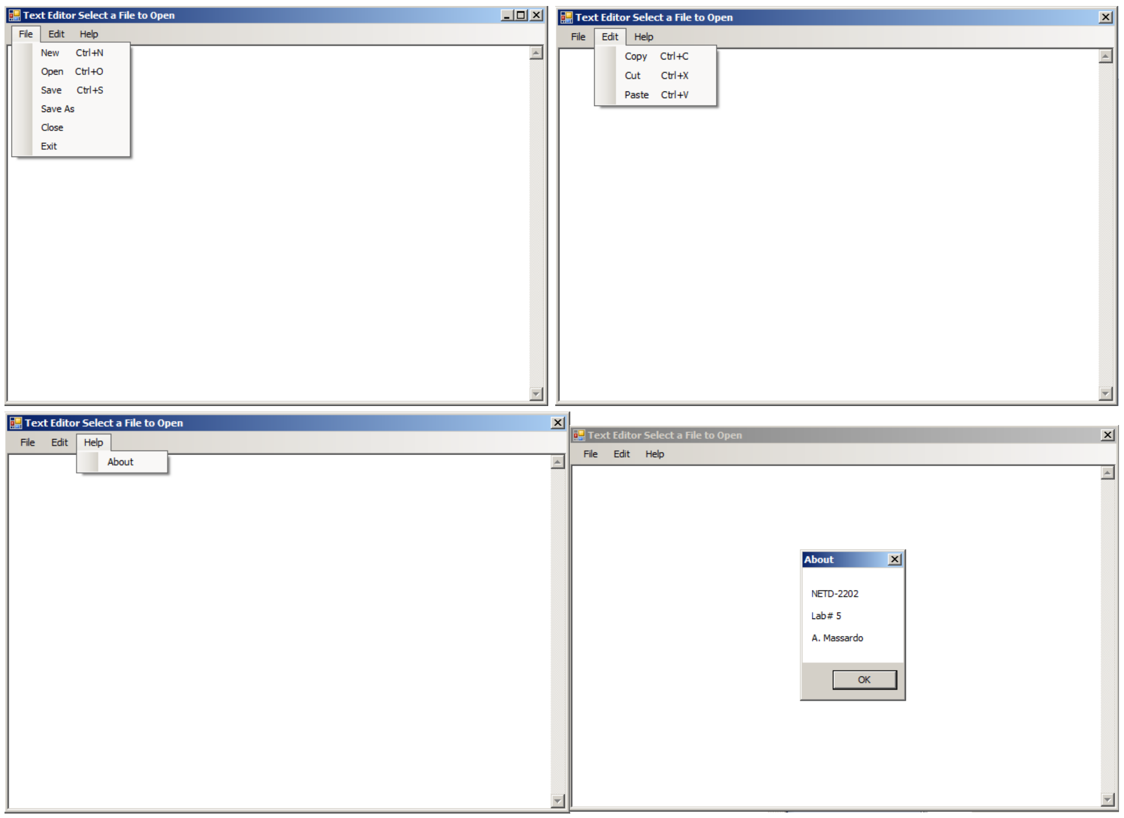
The application will also require the ability to copy, cut, and paste text.

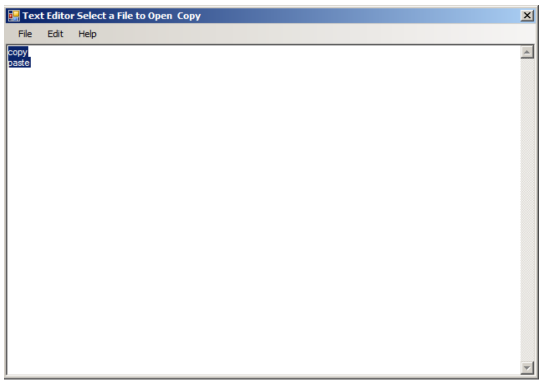
The solution should also allow the user to exit the program.

As part of the company audit process you will be expected to submit your code, a desk-check for the quality assurance department and a link to your GitHub account, specifically, to the new code.

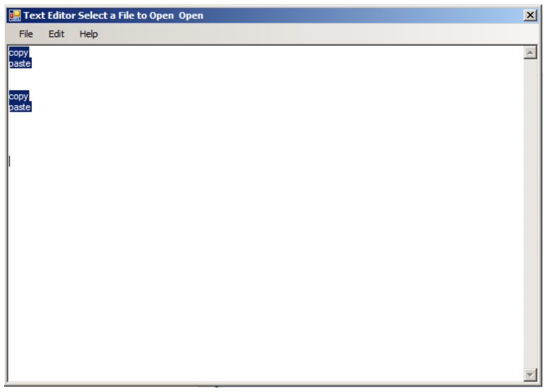
**IMPORTANT** any submission submitted as anything other than a .zip file will be penalized 15%.

## Instructions from the UX (User Experience) Department



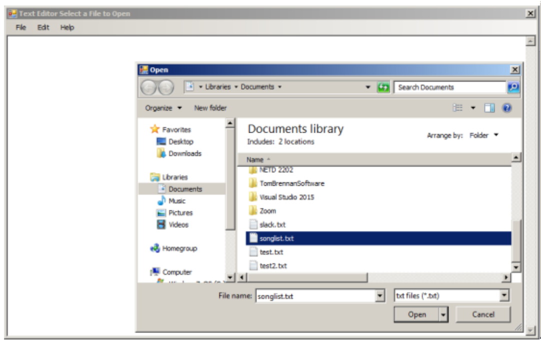


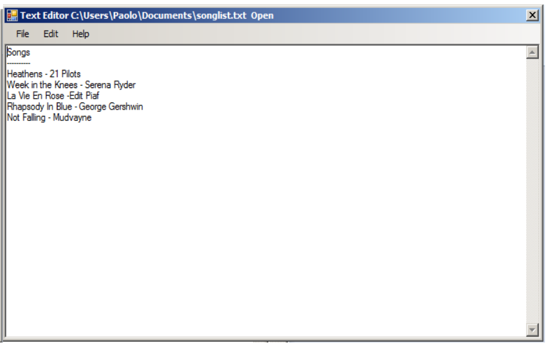
```
copy
paste
```



```
copy
paste

copy
paste
```





```
Songs
----------
Heathens - 21 Pilots
Week in the Knees - Serena Ryder
La Vie En Rose -Edit Piaf
Rhapsody In Blue - George Gershwin
Not Falling - Mudvayne
```

1. The form should…
   - Allow maximizing or minimizing of the screen.
   - Have its title bar set to something meaningful. Should be set dynamically based on current task.
   - Ensure its Accept and Cancel buttons not are set.
   - Open in the centre of the screen.
   - Be named as per our development style guide.

2. Menu items should…
   - Each menu item should…
     - Have its text set appropriately using access keys (&) to ensure efficiency.
     - Have a tool tip to ensure the user understands the purpose of the menu item.
     - Have short cuts for File (New, Open, Save, Save As, Exit), Edit (Copy, Cut, Paste), Help (About).
     - Be named as per our development style guide.

3. Input text box should…
   - Have its tab index set.
   - Text should be empty.
   - Have a tool tip to ensure the user understands the purpose of the control.
   - Allow for multiple lines.
   - Allow for scroll bars.
   - Set the text align to left.
   - Be named as per our development style guide.

4. ToolTip control should be named as per our development style guide

**User Input Flow**

1. The user should be allowed to Exit the solution by…
    - Clicking the menu item File>>Exit with the mouse on the screen.
    - Clicking the Control Box with the mouse on the screen.
    - Access Key Alt F and then Alt X.
        - Check if there have been any alterations before exiting (Optional)
2. The user should be allowed to Save As the document by …
    - Activate the Save Dialog Box
        - Clicking the menu item File>>Save As with the mouse on the screen.
        - Access Key Alt F and then Alt A.
3. The user should be allowed to Save a document by …
    - Activate the Save Dialog Box
        - Clicking the menu item File>>Save with the mouse on the screen.
        - Clicking the short cut Ctrl+s
        - Access Key Alt F and then Alt S
4. The user should be allowed to Open a document by …
    - Activate the Open Dialog Box
        - Clicking the menu item File>>Open with the mouse on the screen.
        - Clicking the short cut Ctrl+o
        - Access Key Alt F and then Alt O
    - Select the file to open
        - Check if there have been any alterations before opening (Optional)
5. The user should be allowed to create a New document by …
    - Clicking the menu item File>>New with the mouse on the screen.
    - Clicking the short cut Ctrl+N
    - Access Key Alt F and then Alt N
        - Check if there have been any alterations before creating a new document (Optional)
6. The user should be allowed to Copy text by…
    - Highlighting the text in the text box and…
        - Clicking the menu item Edit>>Copy with the mouse on the screen.
        - Clicking the short cut Ctrl+C
        - Access Key Alt E and then Alt C
7. The user should be allowed to Cut text by…
    - Highlighting the text in the text box and…
        - Clicking the menu item Edit>>Cut with the mouse on the screen.
        - Clicking the short cut Ctrl+X
        - Access Key Alt E and then Alt T

8. The user should be allowed to Paste text by …
   - Selecting the start point where the text will be place and…
     - Clicking the menu item Edit>>Paste with the mouse on the screen.
     - Clicking the short cut Ctrl+V
     - Access Key Alt E and then Alt P

## Instructions from the Tech Lead

1. All classes and form must be set to `Option Strict On`

2. Open…
   - Should use the `OpenFileDialog` to allow the user to select the file.
   - Should use FileStream object should be instantiated using the full file path, `FileMode.Open, and FileAccess.Read` in the constructor.
   - Should instantiate a StreamReader using an instantiated FileStream object, in its constructor, to designate the file it will be reading from.
   - The newly created StreamReader object should use its `ReadToEnd()` to load the data from the file to the text editor text box.
   - The StreamReader should be Closed (.Close) to ensure the resource is released.

3. Save…
   - If the file path is not known then the SaveFileDialog will be used to allow the user to browse and select the file path where the file will be saved.
   - If the file path is known (existing selected file) then there is no need to display the SaveFileDialog
   - There should be a single save method/sub that can be reused from different parts of the form.
     - The method should take 1 parameter…
       - full file path (including the file name)
     - A FileStream object should be instantiated using the full file path, `FileMode.Create, and FileAccess.Write` in the constructor.
     - The FileStream object should use an instantiated WriteStream object, in WriteStream constructor, to designate the file it will be saving to.
     - The method should use the WriteStream object to get the text from the text editor text box to Write and the Close

4. Save As…
   - Similar to Save, but it should always display the SaveFileDialog because you are creating a new file.
   - The method created for the Save can be reused.

5. New…
   - Should clear the screen and set any variables to default values that represent a new document. (e.g. No file path, no file name, etc.)

6. Cut, should have its own method that…
   - Removes the selected text.
   - Puts the selected text into the `My.Computer.Clipboard.SetText(textSelected)`
7. Copy, should have its own method that…
   - Puts the selected text into the `My.Computer.Clipboard.SetText(textSelected)`.
8. Paste, should have its own method that…
   - Inserts the text in the `My.Computer.Clipboard.GetText()` into the spot selected by the user.
9. Has changed check. (Optional – nice to have as it will impress the client)
   - Somehow it would need to store in memory the contents of a file when it either has been opened or save.
   - Then compare the what the user has in the text editor text box to what is stored in memory.
10. Questions: Tech Lead is Alfred at alfred.massardo@durhamcollege.ca.