Name Session:

Programming II

Lab Exercise 5/5/2020

Part I - Building a Word Processor

Project Overview

In this project we will create a simple word processor. Our word processor will have an area for typing text and a menu bar. Our menu bar will have File, Format, Edit, and Color menu items. We will use a RichTextBox control to hold the users’ text. We will use the OpenFileDialog, SaveFileDialog, FontDialog, ColorDialog, and MainMenu controls to add functionality to our word processor.

# Build the GUI

1. Create a new Windows Application and call it *Word Processor*.
2. Resize the Form to 1024 x 768
3. Add a RichTextBox control to your Form.
4. Add a MenuStrip control to your Form
5. Type &File where is says to *Type Here*.
6. Add the following menu items under File:

* &Open
* &Save
* E&xit

1. Click on the Type Here to the right of your File menu and type Format.
2. Add the following menu items under Format:
   * Font
   * Caps
   * Center Justify
   * Left Justify
   * Right Justify
3. Click on the Type Here to the right of your Format menu and type Edit.
4. Add the following menu items under Edit:
   * Copy
   * Paste
   * Cut
5. Click on the Type Here to the right of your Edit menu and type Color.
6. Add the following menu items under Color:
   * Font
   * Background
7. Add the following dialog controls to your Form:
   * OpenFileDialog
   * SaveFileDialog
   * FontDialog
   * ColorDialog

# The Code

The following controls are associated (if you did them in the correct order) with menu options:

MenuItem2 Open

MenuItem3 Save

MenuItem4 Close

MenuItem6 Font

MenuItem7 Caps

MenuItem8 Center Justify

MenuItem9 Left Justify

MenuItem10 Right Justify

MenuItem12 Copy

MenuItem13 Paste

MenuItem14 Cut

MenuItem16 Font

MenuItem17 Background

1. Add a click event handler for each of the above menu items by going to View Code and selecting the Class Name (i.e. MenuItem2) and the Method Click from the Dropdown boxes at the top of the code window.
2. Add the following code to each of the MenuItem event handlers.

//File Open

OpenFileDialog1.ShowDialog();

RichTextBox1.LoadFile(OpenFileDialog1.FileName);

this.Text = "C#.NET WP - " + OpenFileDialog1.FileName;

//File Save

SaveFileDialog1.ShowDialog();

RichTextBox1.SaveFile(SaveFileDialog1.FileName);

this.Text = "C#.NET WP - " + SaveFileDialog1.FileName;

//Exit

this.Close();

//Font

FontDialog1.ShowDialog();

RichTextBox1.SelectionFont = FontDialog1.Font;

//Convert to all caps

RichTextBox1.SelectedText = RichTextBox1.SelectedText.ToUpper();

//Center Justify

RichTextBox1.SelectionAlignment = HorizontalAlignment.Center;

//Left Justify

RichTextBox1.SelectionAlignment = HorizontalAlignment.Left;

//Right Justify

RichTextBox1.SelectionAlignment = HorizontalAlignment.Right;

//Copy

RichTextBox1.Copy();

//Paste

RichTextBox1.Paste();

//Cut

RichTextBox1.Cut();

//Font Color

ColorDialog1.ShowDialog();

RichTextBox1.ForeColor = ColorDialog1.Color;

//Background Color

ColorDialog1.ShowDialog();

RichTextBox1.BackColor = ColorDialog1.Color;

1. Double click on the Form to get a Form1\_Load event handler and add the following code to that event (Customize for your use):

RichTextBox1.Text = "";

this.Text = "Mr. Messa’s Word Processor";

1. Now test your work processor by creating a file and saving it. Also try opening another RTF file.

# Part II - Word Processor Improvements

Add the following features as a menu item under Tools

1. Add a Word Counter menu item that counts the words in your RichTextBox control. Note: you may use either the wordCount or wordCount2 function.
   1. To the WordCountToolStripMenuItem\_Click, add the following code:

string words;

int count;

string message;

words = RichTextBox1.Text;

//count = wordCount(words);

count = wordCount2(words);

message = "You document contains " + count + " words";

MessageBox.Show(message);

* 1. Add the following code to the wordCount function

int count;

count = 0;

for (int letter = 0; letter <= w.Length - 1; letter++)

{

if (w[letter] == ' ')

count++;

}

return count + 1;

* 1. Add the following code to the wordCount2 function

int count;

string[] words = w.Split(' ');

count = words.Length;

return count;

1. Add a feature that calculates the average word length of your document. Use this feature to determine the grade level of you document based on average word length.
   1. Add the following code to the GradeLevelToolStripMenuItem\_Click event handler

string words;

int count;

string message;

double average;

string gradeLevel;

words = RichTextBox1.Text;

//count = wordCount(words);

count = wordCount2(words);

average = (double)(words.Length - (count - 1)) / count;

gradeLevel = grade(average);

message = "Average word length: " + average.ToString("f2") +

Environment.NewLine + "Grade Level: " + gradeLevel;

MessageBox.Show(message);

* 1. Add the following code to the grade function

if (avg < 3.5)

return "Elementary";

else if (avg < 4.5)

return "Middle School";

else if (avg < 6.5)

return "High School";

else

return "Adult";

1. Add a feature that counts the number of sentences.
   1. Add the following code to the sentenceCountToolStripMenuItem\_Click event handler.

string message;

int count;

count = sentenceCount(RichTextBox1.Text);

message = "Sentences in the document: " + count;

MessageBox.Show(message);

* 1. Add the following code to the sentenceCount function

int count = 0;

for (int i = 0; i < w.Length; i++)

{

if (w[i] == '.' || w[i] == '?' || w[i] == '!')

count++;

}

return count;

**When your word processor is completed and tested, send me a screenshot of it in its running state.**