|  |
| --- |
| College LaSalle |
| Project - Oriented Object Programming User and Technical Manual |
|  |
| Presented to: Mihai Maftei. |

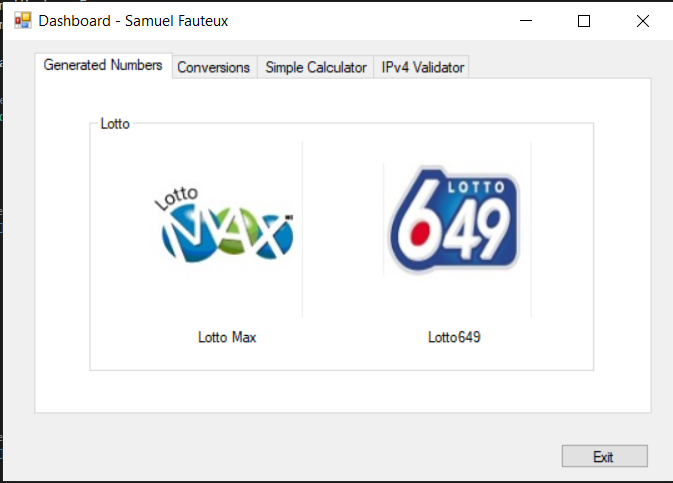
|  |
| --- |
| Your name: Samuel Fauteux  4/11/2021 |

1. **Start by adding a short description of your project, and the languages (technologies) used:**

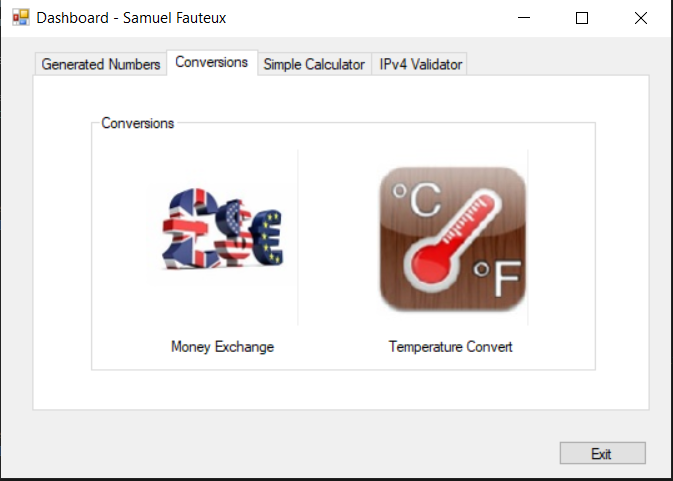
This C# program made with visual studio 2019 offers a graphical interface for multiple conversions, a lotto number generator for LottoMax and 649, a calculator and and IPv4 validator.

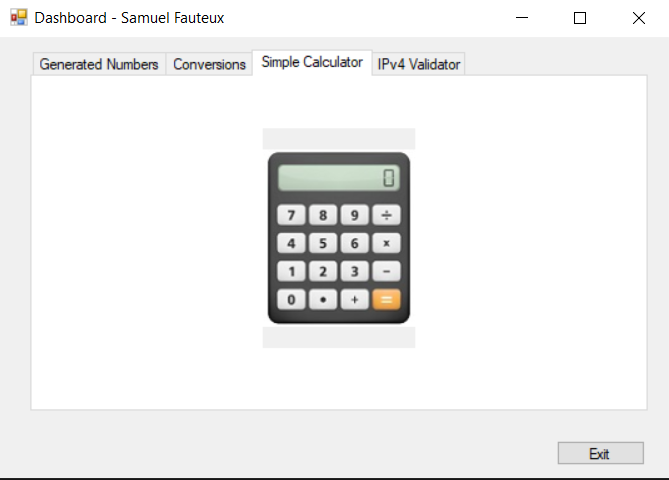
1. **Present the print screens of yours forms, and have a detailed description of the functionalities (step by step).**

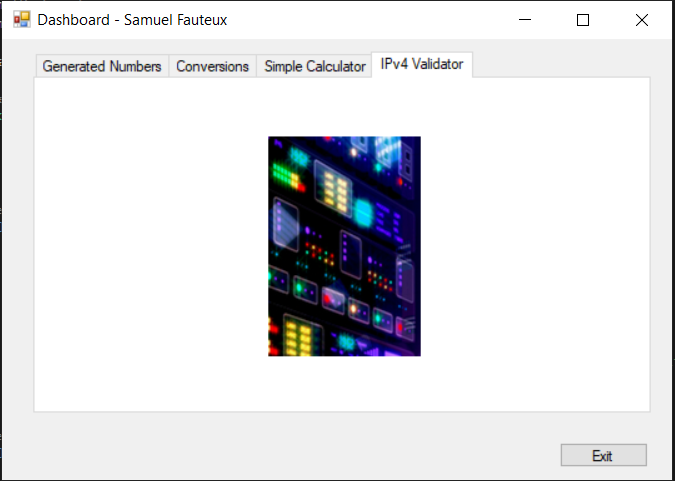
When you open the application, you are presented with this screen.



You can circle through the tabs to get those screens:







In any tab, clicking on one of the PictureBox will lead you to its related application:

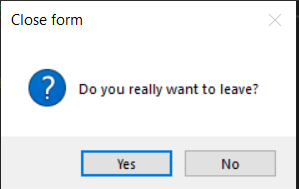
In Generated Numbers, both pictures will lead you to a screen letting you generate the lotto numbers of the lotto with the name you picked and read the latest result of each.

In Conversions, you can choose to either open the Money Exchange application or the temperature conversion one.

In Simple Calculator, you can open a Simple Calculator.

In IPv4 Validator, you can access the IPv4 validator.

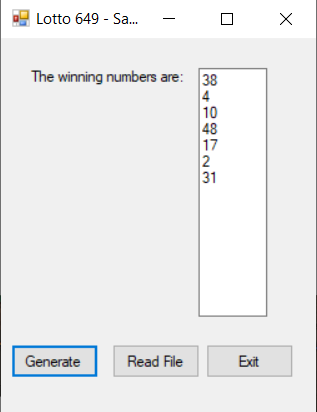
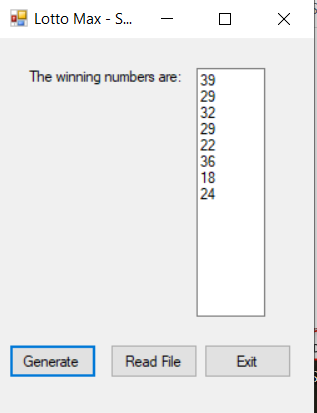
No matter on which tab you are, if you press the Exit button, you will be presented with this MessageBox:



If you click yes, the whole program close.

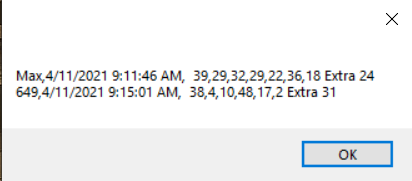
# Lotto

This is the Lotto form:

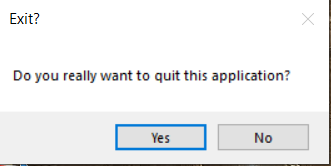


When you click on Generate, the chosen lotto numbers, saves it in a txt file and displays it in the box

When you click on Read File, you are presented with the last generated number of each, like this:

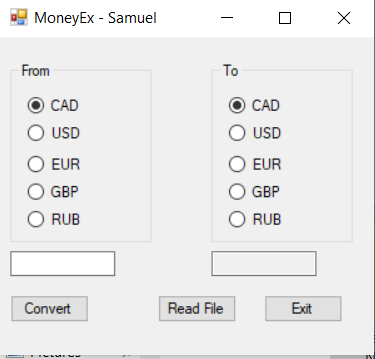


And when you click Exit, like in the main form, you are bothered before exiting only if you click yes



# Money Exchange

This is the Money Exchange application:

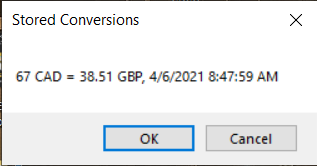


You can use the two set of radio buttons to change from which and to which money you want to go.

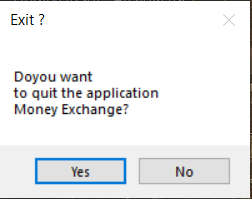
The left TextBox is where you put the amount to convert, the right one is the converted amount

The convert button, convert, saves the conversion and change the amount in the right textbox to match

The Read File button displays all the conversions saved, one by one, like this:

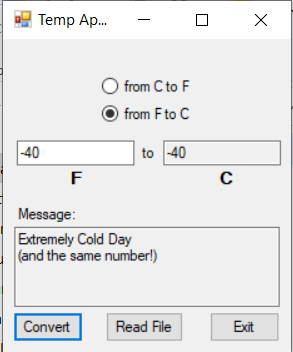
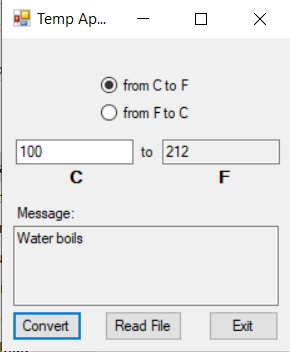


And as for the two previous form the exit button interact with you before letting you leave:



# Temperature Conversions

This is the Temperature Conversions form:



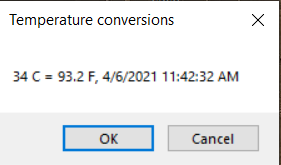
The radio buttons let you switch mode between Celsius to Fahrenheit and Fahrenheit to Celsius

The left TextBox is the input and the right one is the output.

The bottom one only displays something if you hit specific temperatures.

The Convert button takes the inputted temperature, convert it, save the conversion and display the output in the right TextBox

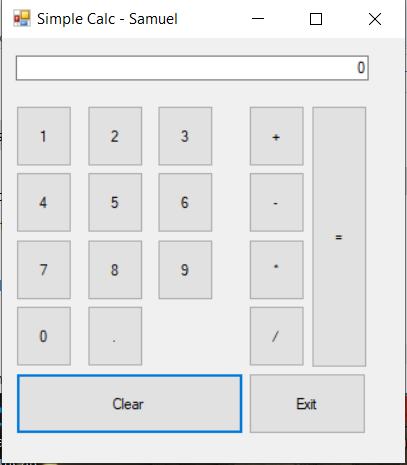
The Read File button shows you the saved conversions in a similar way to the previous application:



For once, the Exit button does its job quietly!

# Simple Calculator

This is my calculator:



If you press on one of the numbers, it is written in the top TextBox.

The dot is the decimal marker.

If you press on one of the operation buttons, the calculator screen will show 0 as a default value.

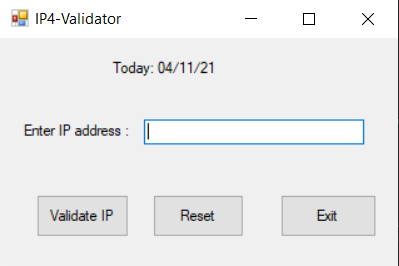
When pressing on =, the operation finishes its calculations and the whole thing is saved in a txt file before showing you the result on screen

The clear button clears the screen and the ongoing operation.

The Exit button once again let you exit without bothering you.

# IPv4 Validator

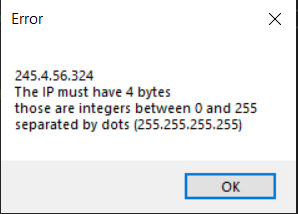
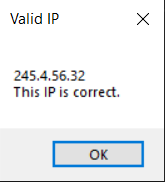
This is the IPvalidator form:



The top label shows today’s date.

The TextBox here, is waiting for an IPv4 to validate.

The Validate IP button verifies if the content of the TextBox is a valid IPv4 address and tell its answer with a text box, like so:



The Reset button empties the TextBox.

The Exit button once again does its job silently!

1. **Present the code of your application (forms) and**
2. **Present the classes and/or methods that you create or you did use in the project.**

# Form0

//Samuel Fauteux

//03/08/2021

/\* Lasalle Session 2 OOP Project 1:

\* This project has a dashboard that let the user chose between many function.

\* The Form0 is this dashboard

\* Form1+2 let the users generate a random lotto max or 649 numbers

\* Form3 is a simple calculator

\* Form4 let you change from one currency to the other

\* Form5 tun C° to f° and the opposite

\* Form6 display the IPv4 Validator

\*/

using System;

using System.Windows.Forms;

namespace LasS2project

{

public partial class Form0 : Form

{

/// <summary>

/// This is the default constructor, it initialize the components

/// </summary>

public Form0()

{

InitializeComponent();

}

/// <summary>

/// This is the exit button, it asks you if you want to leave before closing the application if you say yes

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Do you really want to leave?", "Close form", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

Close(); //we are in the main form so it actually closes the whole application

}

}

/// <summary>

/// Open the lotto form for LottoMax and hide this window

/// </summary>

private void pbLottoMax\_Click(object sender, EventArgs e)

{

Form1\_2 lottoMax = new Form1\_2(false);

lottoMax.Show();

Hide();

}

/// <summary>

/// Open the lotto form for Lotto649 and hide this window

/// </summary>

private void pb649\_Click(object sender, EventArgs e)

{

Form1\_2 lotto649 = new Form1\_2(true);

lotto649.Show();

Hide();

}

/// <summary>

/// Open the calculator form and hide this window

/// </summary>

private void pbCalculator\_Click(object sender, EventArgs e)

{

Form3 calculator = new Form3();

calculator.Show();

Hide();

}

/// <summary>

/// Open the money exchange form and hide this window

/// </summary>

private void pbMoneyExchange\_Click(object sender, EventArgs e)

{

Form4 exchangeApp = new Form4();

exchangeApp.Show();

Hide();

}

/// <summary>

/// Open the temperature convertor form and hide this window

/// </summary>

private void pbTemperature\_Click(object sender, EventArgs e)

{

Form5 temperatureConvertor = new Form5();

temperatureConvertor.Show();

Hide();

}

/// <summary>

/// Open the IPv4 validator form and hide this window

/// </summary>

private void pbIP\_Click(object sender, EventArgs e)

{

Form6 ipValidator = new Form6();

ipValidator.Show();

Hide();

}

}

}

# Form1+2

using System;

using System.Windows.Forms;

using System.IO;

namespace LasS2project

{

public partial class Form1\_2 : Form

{

const string saveFileName = "LottoNbrs.txt"; //all my save file are beside the .exe

bool generate8no = false; //this is the main difference between LottoMax and Lotto649 so I decided to rename it like that since it was why I created this variable but in the end I used it as a isLottoX

/// <summary>

/// this is the default constructor, I used it to test a few things but should not be called by the application once finished, it only initialyses the components

/// </summary>

public Form1\_2()

{

InitializeComponent();

}

/// <summary>

/// this is the constructor over loaded with a boolean value, it initializes the component, changes the title of the form and tell if we generat8no or not

/// </summary>

/// <param name="is649">defines if this form is generated for 649 or not</param>

public Form1\_2(bool is649)

{

InitializeComponent();

if(is649)

{

Text = "Lotto 649 - Samuel";

}

else

{

Text = "Lotto Max - Samuel";

generate8no = true;

}

}

/// <summary>

/// generates the random lotto numbers and the extra

/// </summary>

/// <param name="lottoNbs">the generated lotto numbers</param>

/// <param name="extra">the generated extra</param>

void GenerateLottoNb(out string lottoNbs, out byte extra)

{

Random r = new Random();

byte nbOfNbToGenerate = 7;

byte noMaxValue = 49;

if (generate8no)

{

nbOfNbToGenerate = 8;

noMaxValue = 50;

}

lottoNbs = "";

for (int i = 0; i < nbOfNbToGenerate - 1; ++i)

{

lottoNbs += r.Next(1, noMaxValue).ToString();

if (i < nbOfNbToGenerate - 2)

lottoNbs += ",";

}

extra = Convert.ToByte(r.Next(1, noMaxValue));

}

/// <summary>

/// Replace a specified line in the save file by an other one generated in this function based on the lotto info

/// </summary>

/// <param name="lineToModify">the line to replace</param>

/// <param name="lottoNbs">the lotto numbers</param>

/// <param name="extra">the lotto extra</param>

/// <param name="save">the entire content of the save file as a string</param>

void ReplaceLineInSaveFile(string lineToModify, string lottoNbs, byte extra, string save)

{

string modifiedLine = "";

try

{

modifiedLine = lineToModify.Split(' ')[0].Split(',')[0] + "," + DateTime.Now + ", " + lottoNbs + " Extra " + extra;

}

catch (IOException)

{

MessageBox.Show("The file " + saveFileName + " exist but is invalid, deleting it will fix this problem.", "Error", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);

}

catch (NullReferenceException)

{

MessageBox.Show("The file " + saveFileName + " exist but is invalid, deleting it will fix this problem.", "Error", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);

}

save = save.Replace(lineToModify, modifiedLine);

FileStream fs = new FileStream(saveFileName, FileMode.Create, FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

sw.Write(save);

sw.Close();

}

/// <summary>

/// save the lotto informations in a txt file

/// </summary>

/// <param name="lottoNbs">the lotto numbers</param>

/// <param name="extra">the lotto extra</param>

void Save(string lottoNbs, byte extra)

{

bool fileDoesExist = true;

if (!File.Exists(saveFileName))

fileDoesExist = false;

FileStream fs = new FileStream(saveFileName, FileMode.OpenOrCreate, FileAccess.ReadWrite);

//Creates the file with some basic informations if it doesn't already exist

if (!fileDoesExist)

{

StreamWriter sw = new StreamWriter(fs);

sw.WriteLine("Max," + DateTime.Now + ", NAN Extra NAN");

sw.WriteLine("649," + DateTime.Now + ", NAN Extra NAN");

sw.Close();

}

fs.Close(); //if The file had to be created, it will already be closed

fs = new FileStream(saveFileName, FileMode.Open, FileAccess.Read);

StreamReader sr = new StreamReader(fs);

string save = sr.ReadToEnd();

sr.BaseStream.Position = 0; //it should be it's default value

string lineToModify = sr.ReadLine();

//Current goal: replace the first line of the if and else with well understood sr elements, I already moved the sr close //SUCCESS!

if (generate8no)

{

sr.Close();

ReplaceLineInSaveFile(lineToModify, lottoNbs, extra, save);

}

else

{

lineToModify = sr.ReadLine();

sr.Close();

ReplaceLineInSaveFile(lineToModify, lottoNbs, extra, save);

}

}

/// <summary>

/// Generate the lotto numbers and the extra, save them in a txt file and display them

/// </summary>

private void btnGenerate\_Click(object sender, EventArgs e)

{

GenerateLottoNb(out string lottoNbs, out byte extra);

Save(lottoNbs, extra);

txtLottoNb.Text = lottoNbs.Replace(",", Environment.NewLine) + Environment.NewLine + extra;

}

/// <summary>

/// Display the content of the lotto save file (the latest LottoMax and latest 649) in a MessageBox

/// </summary>

private void btnReadFile\_Click(object sender, EventArgs e)

{

if (File.Exists(saveFileName))

{

FileStream fs = new FileStream(saveFileName, FileMode.Open, FileAccess.Read);

StreamReader sr = new StreamReader(fs);

MessageBox.Show(sr.ReadToEnd());

}

else

{

MessageBox.Show("This file doesn't exist yet!");

}

}

/// <summary>

/// Asks if the user really wants to quit and if so, show the main form before closing this one

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Do you really want to quit this application?", "Exit?", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

Common.ShowForm0();

Close();

}

}

}

}

# Form3

using System;

using System.Linq;

using System.Windows.Forms;

namespace LasS2project

{

public partial class Form3 : Form

{

const string saveFileName = "Calculator.txt";

string firstOperand = ""; //the first number of the calculation and often, the result of the last one //note: things are cleared after the =

string secondOperand = ""; //the latest number the user enterd in the calculator

char currentOperation = ' '; //the opération to perform between the fisrt operand and the second one

string theOperationSoFar = ""; //the operation done before pressing =

/// <summary>

/// This is the default constructor, it initialize the components

/// </summary>

public Form3()

{

InitializeComponent();

}

/// <summary>

/// check if the operand starts with a zero and if so, if it is required

/// </summary>

/// <returns>true if it start with an unnecesary 0</returns>

bool CheckIfOperendStartWithUnnessesary0()

{

if (txtCalcScreen.Text[0] != '0')

return false;

try

{

if (txtCalcScreen.Text[1] == '.')

return false;

}

catch(IndexOutOfRangeException)

{

//the string is not 2char long

}

return true;

}

/// <summary>

/// writes 1 in the calculator screen when the button 1 is pressed

/// </summary>

private void btn1\_Click(object sender, EventArgs e)

{

if(CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "1";

}

else

{

txtCalcScreen.Text += "1";

}

}

/// <summary>

/// writes 2 in the calculator screen when the button 2 is pressed

/// </summary>

private void btn2\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "2";

}

else

{

txtCalcScreen.Text += "2";

}

}

/// <summary>

/// writes 3 in the calculator screen when the button 3 is pressed

/// </summary>

private void btn3\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "3";

}

else

{

txtCalcScreen.Text += "3";

}

}

/// <summary>

/// writes 4 in the calculator screen when the button 4 is pressed

/// </summary>

private void btn4\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "4";

}

else

{

txtCalcScreen.Text += "4";

}

}

/// <summary>

/// writes 5 in the calculator screen when the button 5 is pressed

/// </summary>

private void btn5\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "5";

}

else

{

txtCalcScreen.Text += "5";

}

}

/// <summary>

/// writes 6 in the calculator screen when the button 6 is pressed

/// </summary>

private void btn6\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "6";

}

else

{

txtCalcScreen.Text += "6";

}

}

/// <summary>

/// writes 7 in the calculator screen when the button 7 is pressed

/// </summary>

private void btn7\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "7";

}

else

{

txtCalcScreen.Text += "7";

}

}

/// <summary>

/// writes 8 in the calculator screen when the button 8 is pressed

/// </summary>

private void btn8\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "8";

}

else

{

txtCalcScreen.Text += "8";

}

}

/// <summary>

/// writes 9 in the calculator screen when the button 9 is pressed

/// </summary>

private void btn9\_Click(object sender, EventArgs e)

{

if (CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text = "9";

}

else

{

txtCalcScreen.Text += "9";

}

}

/// <summary>

/// writes 0 in the calculator screen when the button 0 is pressed

/// </summary>

private void btn0\_Click(object sender, EventArgs e)

{

if (!CheckIfOperendStartWithUnnessesary0())

{

txtCalcScreen.Text += "0";

}

}

/// <summary>

/// add a dot to mark the decimal numbers if there is not already one, else warn the user

/// </summary>

private void btnDot\_Click(object sender, EventArgs e)

{

if (txtCalcScreen.Text.Contains('.'))

{

MessageBox.Show("A number cannot have 2 dots", "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

txtCalcScreen.Text += ".";

}

}

/// <summary>

/// remove the dot at the end of a string

/// </summary>

/// <param name="text">the string to remove a dot from</param>

/// <returns>the same string without a dot at the end</returns>

string RemoveTrailingDot(string text)

{

int textLastIndex = text.Length - 1;

if (text[textLastIndex] == '.')

text = text.Remove(textLastIndex);

return text;

}

/// <summary>

/// assing the text on the calculator screen in the right operend in order to calculate things

/// </summary>

void AssingTxtboxValueInRightOperand()

{

if (firstOperand == "")

{

firstOperand = txtCalcScreen.Text;

firstOperand = RemoveTrailingDot(firstOperand);

theOperationSoFar = firstOperand;

}

else

{

secondOperand = txtCalcScreen.Text;

secondOperand = RemoveTrailingDot(secondOperand);

}

}

/// <summary>

/// check which operation the user tries to perform and perform it

/// </summary>

void ApplyDesiredOperation()

{

Double.TryParse(firstOperand, out double x);

Double.TryParse(secondOperand, out double y);

switch(currentOperation)

{

case '+': theOperationSoFar += "+"; Add(x, y); break;

case '-': theOperationSoFar += "-"; Substract(x, y); break;

case '\*': theOperationSoFar += "\*"; Multiply(x, y); break;

case '/': theOperationSoFar += "/"; Divide(x, y); break;

default: MessageBox.Show("This operation wasn't recognised.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); break;

}

theOperationSoFar += secondOperand;

}

/// <summary>

/// add x and y together

/// </summary>

/// <param name="x">first operand as a double</param>

/// <param name="y">second operand as a double</param>

void Add(double x, double y)

{

Calculator calc = new Calculator(x, y);

firstOperand = calc.Add().ToString();

}

/// <summary>

/// substract x from y

/// </summary>

/// <param name="x">first operand as a double</param>

/// <param name="y">second operand as a double</param>

void Substract(double x, double y)

{

Calculator calc = new Calculator(x, y);

firstOperand = calc.Substract().ToString();

}

/// <summary>

/// multiply x and y together

/// </summary>

/// <param name="x">first operand as a double</param>

/// <param name="y">second operand as a double</param>

void Multiply(double x, double y)

{

Calculator calc = new Calculator(x, y);

firstOperand = calc.Multiply().ToString();

}

/// <summary>

/// divide x by y and yeild an error if there was one

/// </summary>

/// <param name="x">first operand as a double</param>

/// <param name="y">second operand as a double</param>

void Divide(double x, double y)

{

Calculator calc = new Calculator(x, y);

string result = calc.Divide();

if (result == "Error2")

{

MessageBox.Show("You divided by 0\nThe result of this operation was not saved.", result, MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

firstOperand = result;

}

}

/// <summary>

/// Apply the current operation and set the next one to be +

/// </summary>

private void btnAddition\_Click(object sender, EventArgs e)

{

AssingTxtboxValueInRightOperand();

if (secondOperand != "")

{

ApplyDesiredOperation();

}

currentOperation = '+';

txtCalcScreen.Text = "0";

}

/// <summary>

/// Apply the current operation and set the next one to be -

/// </summary>

private void btnSubstraction\_Click(object sender, EventArgs e)

{

AssingTxtboxValueInRightOperand();

if (secondOperand != "")

{

ApplyDesiredOperation();

}

currentOperation = '-';

txtCalcScreen.Text = "0";

}

/// <summary>

/// Apply the current operation and set the next one to be \*

/// </summary>

private void btnMultiplication\_Click(object sender, EventArgs e)

{

AssingTxtboxValueInRightOperand();

if (secondOperand != "")

{

ApplyDesiredOperation();

}

currentOperation = '\*';

txtCalcScreen.Text = "0";

}

/// <summary>

/// Apply the current operation and set the next one to be /

/// </summary>

private void btnDivision\_Click(object sender, EventArgs e)

{

AssingTxtboxValueInRightOperand();

if (secondOperand != "")

{

ApplyDesiredOperation();

}

currentOperation = '/';

txtCalcScreen.Text = "0";

}

/// <summary>

/// set all the class variable and the screen of the calculator to their default value

/// </summary>

void ResetValues()

{

firstOperand = "";

secondOperand = "";

currentOperation = ' ';

theOperationSoFar = "";

txtCalcScreen.Text = "0";

}

/// <summary>

/// finalize the operation, save it and display it in the calculator screen

/// </summary>

private void btnEqual\_Click(object sender, EventArgs e) //you cannot apply an operation on the result, othewise, everything is fine

{

AssingTxtboxValueInRightOperand(); //I don't think performing operations on the result was asked anyways so instead of putting flags I will move on

if (firstOperand == "")

{

firstOperand = "0";

theOperationSoFar = "0";

}

if (secondOperand != "")

{

ApplyDesiredOperation();

}

theOperationSoFar += "=" + firstOperand;

Common.Append(saveFileName, theOperationSoFar);

string temp = firstOperand;

ResetValues();

txtCalcScreen.Text = temp;

firstOperand = temp;

}

/// <summary>

/// reset values

/// </summary>

private void btnClear\_Click(object sender, EventArgs e)

{

ResetValues();

}

/// <summary>

/// show the main form and close this one

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

Common.ShowForm0();

Close();

}

}

}

# Form4

using System;

using System.Windows.Forms;

using System.Text.RegularExpressions;

namespace LasS2project

{

public partial class Form4 : Form

{

const string saveFileName = "MoneyConversions.txt";

/// <summary>

/// This is the default constructor, it initialize the components

/// </summary>

public Form4()

{

InitializeComponent();

}

/// <summary>

/// When this button is clicked, it verifies that the entered number fit the prerequisite or shows an error in a MessageBox

/// If the value format is accepted, it convert this amount in CAD before converting it in another if needed/requested

/// And finally it saves this conveersion in a txt file

/// </summary>

private void btnConvert\_Click(object sender, EventArgs e)

{

string textInput = txtInputAmmount.Text;

Regex twoDecimals = new Regex(@"^[0-9]+\.?[0-9]{0,2}$");

if (twoDecimals.IsMatch(textInput) && textInput[textInput.Length - 1] != '.')

{

string txtline = textInput;

Double.TryParse(textInput, out double value);

//turns the value in CAD

if (rbFromUSD.Checked)

{

value \*= 1.26; //all those change rates are from google 04/03/2021

txtline += " USD = ";

}

else if (rbFromEUR.Checked)

{

value \*= 1.48;

txtline += " EUR = ";

}

else if(rbFromGBP.Checked)

{

value \*= 1.74;

txtline += " GBP = ";

}

else if(rbFromRUB.Checked)

{

value \*= 0.016;

txtline += " RUB = ";

}

else

{

txtline += " CAD = ";

}

//Convert CAD in something else

string toType;

if(rbToUSD.Checked)

{

value /= 1.26;

toType = " USD, ";

}

else if(rbToEUR.Checked)

{

value /= 1.48;

toType = " EUR, ";

}

else if(rbToGBP.Checked)

{

value /= 1.74;

toType = " GBP, ";

}

else if(rbToRUB.Checked)

{

value /= 0.016;

toType = " RUB, ";

}

else

{

toType = " CAD, ";

}

txtOutputAmmount.Text = Math.Round(value, 2).ToString();

txtline += txtOutputAmmount.Text + toType + DateTime.Now;

Common.Append(saveFileName, txtline);

txtInputAmmount.Focus();

}

else

{

MessageBox.Show("The inputed value is not in an accepted format!\n(the value have a max of two decimal and no symbol, e.g. 32.43)", "Invalid Format", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

/// <summary>

/// Shows the conversion log

/// </summary>

private void btnReadFile\_Click(object sender, EventArgs e)

{

Common.ShowLog(saveFileName, "Stored Conversions");

}

/// <summary>

/// Asks if the user want to quit before quitting

/// if yes, show the main form and close this one

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

if(MessageBox.Show("Doyou want\nto quit the application\nMoney Exchange?", "Exit ?", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

Common.ShowForm0();

Close();

}

}

}

}

# Form5

using System;

using System.Windows.Forms;

namespace LasS2project

{

public partial class Form5 : Form

{

const string saveFileName = "TempConversions.txt";

/// <summary>

/// This is the default constructor, it initialize the components

/// </summary>

public Form5()

{

InitializeComponent();

}

/// <summary>

/// Exchage the content of the to and from label when the conversion is switched to this mode

/// </summary>

private void rbCToF\_CheckedChanged(object sender, EventArgs e)

{

lbFrom.Text = "C";

lbTo.Text = "F";

}

/// <summary>

/// Exchage the content of the to and from label when the conversion is switched to this mode

/// </summary>

private void rbFToC\_CheckedChanged(object sender, EventArgs e)

{

lbFrom.Text = "F";

lbTo.Text = "C";

}

/// <summary>

/// assign a message in the message textbox is the temperature as a precise value

/// </summary>

/// <param name="tempC">the temperature</param>

void AssingMessage(double tempC)

{

if (tempC == 100)

{

txtMessage.Text = "Water boils";

}

else if (tempC == 40)

{

txtMessage.Text = "Hot Bath";

}

else if (tempC == 37)

{

txtMessage.Text = "Body temperature";

}

else if (tempC == 30)

{

txtMessage.Text = "Beach weather";

}

else if (tempC == 21)

{

txtMessage.Text = "Room temperature";

}

else if (tempC == 10)

{

txtMessage.Text = "Cool Day";

}

else if (tempC == 0)

{

txtMessage.Text = "Freezing point of water";

}

else if (tempC == -18)

{

txtMessage.Text = "Very Cold Day";

}

else if (tempC == -40)

{

txtMessage.Text = "Extremely Cold Day" + Environment.NewLine + "(and the same number!)";

}

else

{

txtMessage.Text = "";

}

}

/// <summary>

/// makes the conversions, saves them and display the result if the format is correct, else warn the user with a MessageBox

/// </summary>

private void btnConvert\_Click(object sender, EventArgs e)

{

if(Double.TryParse(txtFrom.Text, out double value))

{

string line = txtFrom.Text;

if(rbCToF.Checked)

{

AssingMessage(value);

line += " C = ";

value = value \* 9 / 5 + 32;

line += value + " F, " + DateTime.Now;

}

else

{

line += " F = ";

value = (value - 32) \* 5 / 9;

line += value + " C, " + DateTime.Now;

AssingMessage(value);

}

Common.Append(saveFileName, line);

txtTo.Text = value.ToString();

}

else

{

MessageBox.Show("Format unrecognised!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

/// <summary>

/// Show the temperature conversion log in a MessageBox

/// </summary>

private void btnReadFile\_Click(object sender, EventArgs e)

{

Common.ShowLog(saveFileName, "Temperature conversions");

}

/// <summary>

/// Show the main for and close this on

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

Common.ShowForm0();

Close();

}

}

}

# Form6

using System;

using System.Windows.Forms;

namespace LasS2project

{

public partial class Form6 : Form

{

/// <summary>

/// this is the default constructor, it initializes the components and put today date in the corresponding label

/// </summary>

public Form6()

{

InitializeComponent();

lbToday.Text += DateTime.Now.ToString("MM/dd/yy"); //This is what was requested?

}

/// <summary>

/// checks if the provided IPv4 is valid and tells it to the user with a MessageBox

/// </summary>

private void btnValidate\_Click(object sender, EventArgs e)

{

bool isValidIP = true;

string[] ipParts = txtIP.Text.Split('.');

if (ipParts.Length == 4)

{

foreach (string ipPart in ipParts)

{

if (!Byte.TryParse(ipPart, out \_))

{

isValidIP = false;

break;

}

}

}

else

{

isValidIP = false;

}

if(isValidIP)

{

MessageBox.Show(txtIP.Text + "\nThis IP is correct. ", "Valid IP");

}

else

{

MessageBox.Show(txtIP.Text + "\nThe IP must have 4 bytes\nthose are integers between 0 and 255\nseparated by dots (255.255.255.255)", "Error");

}

}

/// <summary>

/// clear the text field

/// </summary>

private void btnReset\_Click(object sender, EventArgs e)

{

txtIP.Text = "";

}

/// <summary>

/// Show the main for and close this on

/// </summary>

private void btnExit\_Click(object sender, EventArgs e)

{

Common.ShowForm0();

Close();

}

}

}

1. **Present the difficulties that you have, what was the hardest and the easiest part of your project.**

The calculator gave me a few hard times and I started before being comfortable with StringWriter and StringReader, these two introduced in the final meant that I didn’t had much time to test them as I’d like but 3 day ago I found the time to test what I wanted so I am more confidant now. Also regex didn’t pose as much a problem as I tough due to all the test we did in javascript, while it is not the same language, Regex and RegExp are similar enough for me to practice both at the same time. The true easiest part was probably all things related to forms. I tend to forget about them because it is so easy and intuitive (except Environment.BreakLine) that I tend to forget I did it because I don’t consider that it requires logic (well it does require some but not enough to entertain me). While the true hardest would be this word document! I always feel like I misunderstand the questions!