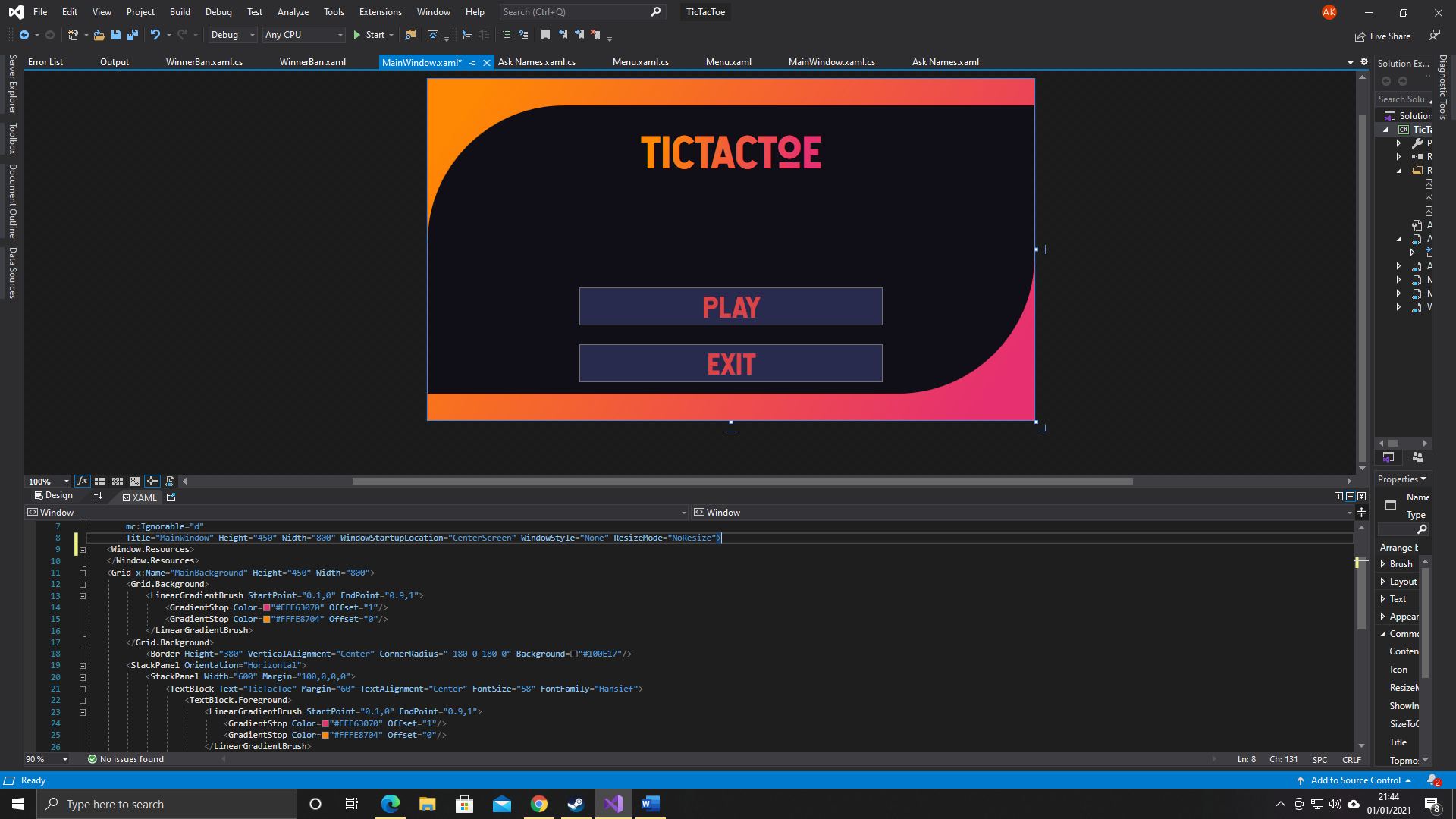
**Code behind the Main Window**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.Windows.Forms;

namespace TicTacToe

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

Ask\_Names askNames;

public MainWindow()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

//Opening a new AskNames form

if (askNames == null)

{

askNames = new Ask\_Names();

}

askNames.Show();

Hide();

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

//Closing the program

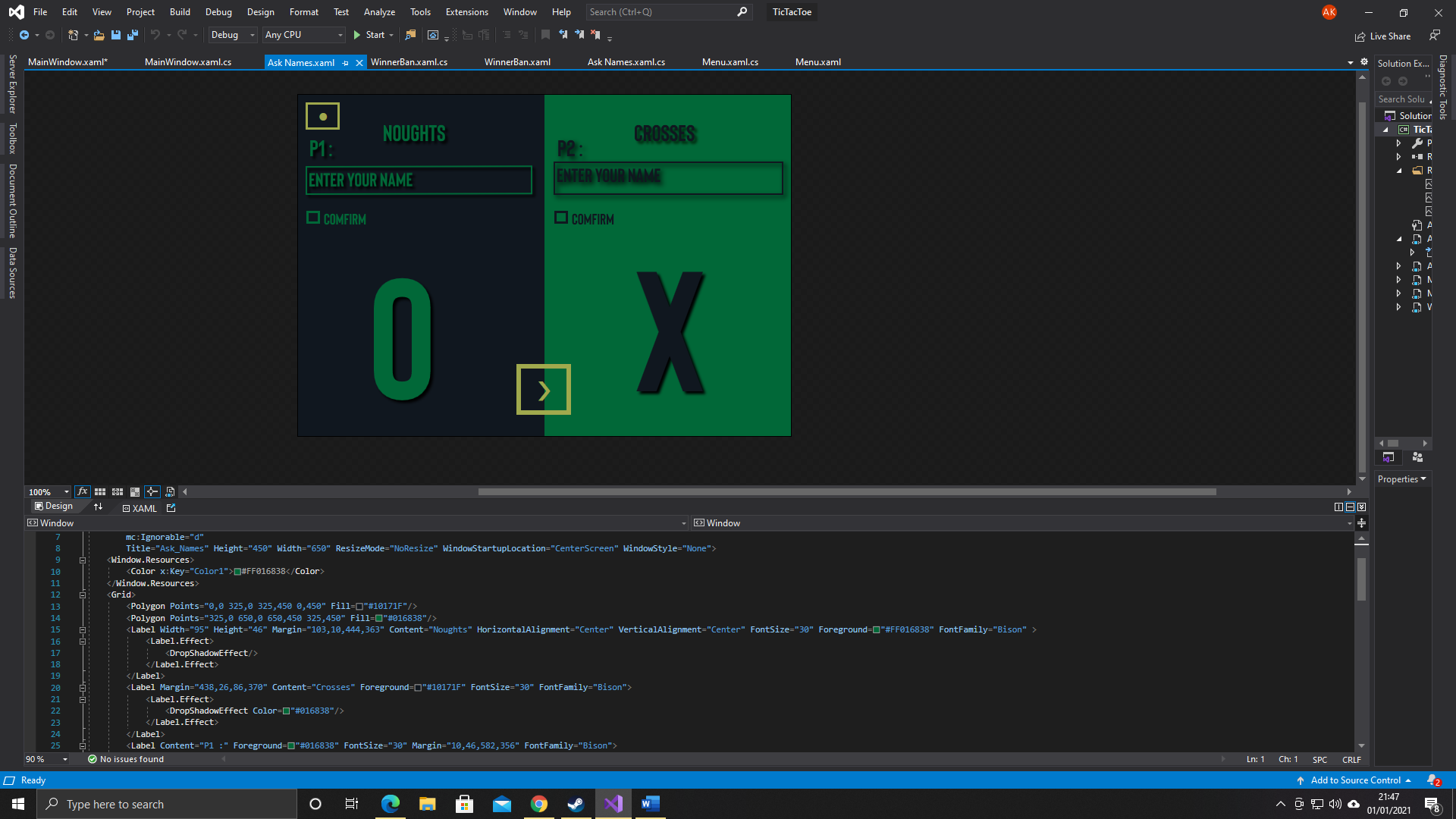
Close();

}

}

}

**Code behind the Ask Names Window**



using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace TicTacToe

{

/// <summary>

/// Interaction logic for Ask\_Names.xaml

/// </summary>

public partial class Ask\_Names : Window

{

//Variables made in order to recall other forms

MainWindow main;

Menu Menu;

public Ask\_Names()

{

main = new MainWindow();

InitializeComponent();

}

private void btnPlay\_Click(object sender, RoutedEventArgs e)

{

//Checks if the TextBoxes and CheckBoxes are all perfectly set

if (ChBoxComfirm1.IsChecked == true && ChBoxComfirm2.IsChecked == true && txtEnterName1.Text != "Enter Your Name" && txtEnterName2.Text != "Enter Your Name" && txtEnterName1.Text != "" && txtEnterName2.Text != "")

{

//Open the Menu form

if (Menu == null)

{

Menu = new Menu();

}

Menu.Show();

Hide();

//Applies the names input onto the other form's label and respective scores

Menu.lblName1.Text = txtEnterName1.Text;

Menu.lblName2.Text = txtEnterName2.Text;

Menu.ScoreContainer1.Content = "0";

Menu.ScoreContainer2.Content = "0";

}

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

//Simply opens the main form

main.Show();

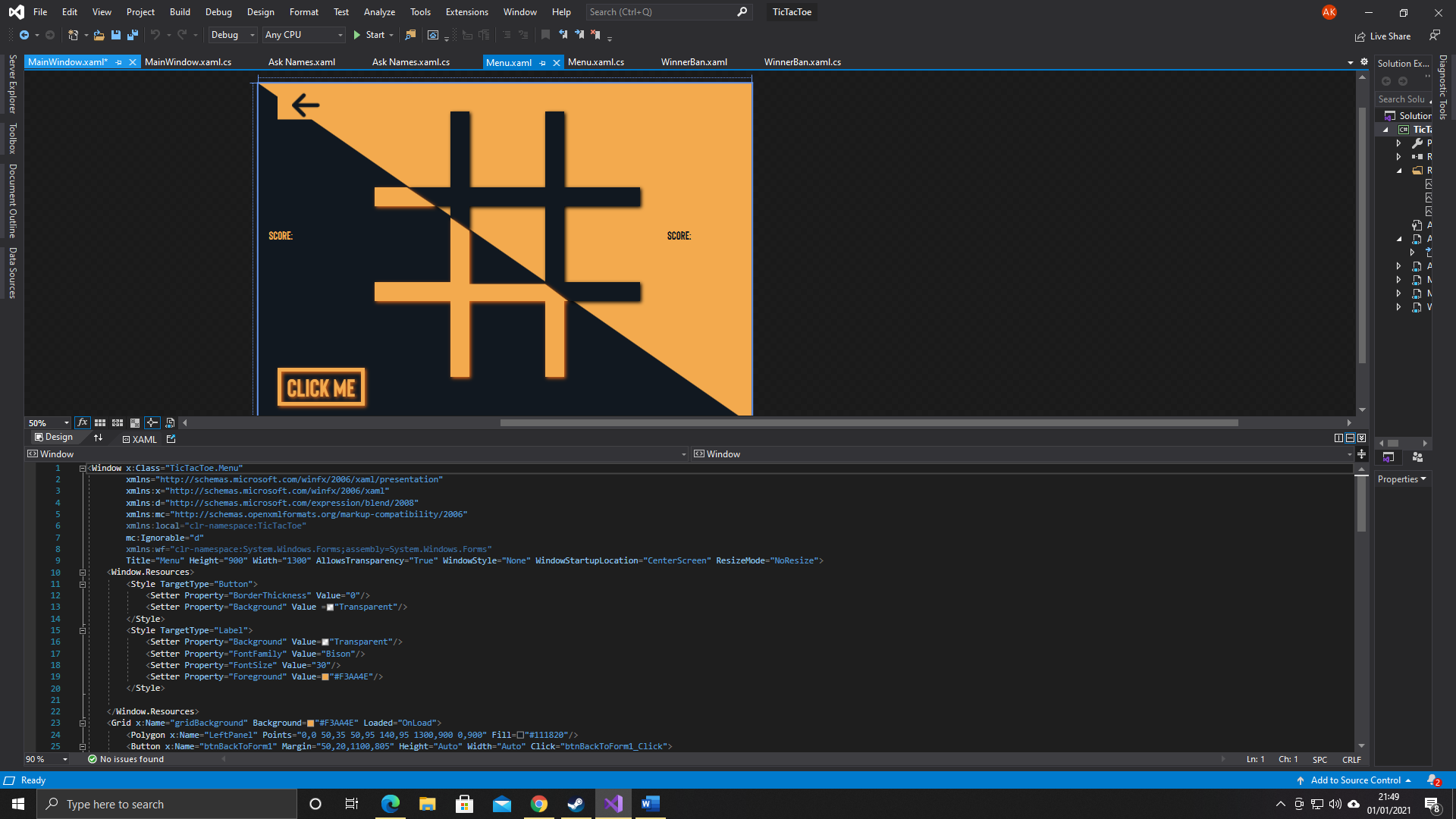
Hide();

}

}

}

**Code behind the Menu Window**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Windows.Media.Effects;

using System.Windows.Threading;

using System.Windows.Forms;

namespace TicTacToe

{

/// <summary>

/// Interaction logic for Menu.xaml

/// </summary>

public partial class Menu : Window

{

//Forms to open when necessary

WinnerBan Winnerban;

Ask\_Names AskNames;

//The array to store the numbers according to the buttons being clicked on the TicTacToe grid

int[,] TicTacToe = new int[3,3];

//A DropShadowEffect to apply the NeonEffect on some controls and the Noughts and Crosses

DropShadowEffect NeonBeLike = new DropShadowEffect();

Color MyDropShadowEffectColor = new Color();

public Menu()

{

AskNames = new Ask\_Names();

InitializeComponent();

}

void OnLoad(object sender, RoutedEventArgs e)

{

//Filling the array with Zeros

for (int rows = 0; rows <= 2; rows++)

{

for (int columns = 0; columns <= 2; columns++)

{

TicTacToe[rows, columns] = 0;

}

}

StartingMove();

}

private void StartingMove()

{

//Defining the properties of the DropShadowEffect

NeonBeLike.ShadowDepth = 0;

NeonBeLike.Direction = 0;

NeonBeLike.BlurRadius = 30;

NeonBeLike.Color = Color.FromRgb(255, 255, 255);

//This piece of code randomly decides who makes the first move

Random Randomicy = new Random();

bool FirstMove;

FirstMove = Convert.ToBoolean(Randomicy.Next(0, 2));

if (FirstMove == true)

{

btnIndicator.Foreground = Brushes.White;

btnIndicator.Effect = NeonBeLike;

}

else if (FirstMove == false)

{

btnIndicator2.Foreground = Brushes.White;

btnIndicator2.Effect = NeonBeLike;

}

}

//Event handler of each button on the TicTacToe grid is associated with this subroutine to make them all work in the same way

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

{

//Create an ellispe and defining its properties

Ellipse Nought = new Ellipse();

Nought.Width = 150;

Nought.Height = 175;

Nought.StrokeThickness = 25;

Nought.Stroke = System.Windows.Media.Brushes.White;

Nought.Fill = System.Windows.Media.Brushes.Transparent;

Nought.Effect = NeonBeLike;

//Create the lines, which will be poligons to give a much more 2D shape-like figure

Polygon Cross = new Polygon();

//Create the set of points for the first polygon

PointCollection PointSet = new PointCollection();

PointSet.Add(new Point(40, 10));

PointSet.Add(new Point(65, 10));

PointSet.Add(new Point(100, 80));

PointSet.Add(new Point(135, 10));

PointSet.Add(new Point(160, 10));

PointSet.Add(new Point(115, 100));

PointSet.Add(new Point(160, 190));

PointSet.Add(new Point(135, 190));

PointSet.Add(new Point(100, 120));

PointSet.Add(new Point(65, 190));

PointSet.Add(new Point(40, 190));

PointSet.Add(new Point(85, 100));

//Assign the Sets of Points to the respective polygons

Cross.Points = PointSet;

//Add details and shape the polygons to have the best look

Cross.StrokeThickness = 0;

Cross.Fill = Brushes.White;

Cross.Width = 200;

Cross.Height = 200;

Cross.Effect = NeonBeLike;

//A DropShadowEffect will help the players understand who is making the current move

//This code basically draws an ellispe or a cross depending on who is making the current move

System.Windows.Controls.Button btn = sender as System.Windows.Controls.Button;

if (btnIndicator.Effect != null)

{

btn.Content = Nought;

btnIndicator.Effect = null;

btnIndicator2.Foreground = Brushes.White;

btnIndicator.Foreground = Brushes.Transparent;

btnIndicator2.Effect = NeonBeLike;

}

else if (btnIndicator2.Effect != null)

{

btn.Content = Cross;

btnIndicator2.Effect = null;

btnIndicator.Foreground = Brushes.White;

btnIndicator2.Foreground = Brushes.Transparent;

btnIndicator.Effect = NeonBeLike;

}

//This states which button has been clicked based on the positions of the buttons

if (btn.Margin.Left == 0 && btn.Margin.Top == 0 && btnIndicator.Effect != null)

{

TicTacToe[0, 0] = 1;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 0 && btnIndicator2.Effect != null)

{

TicTacToe[0, 0] = 10;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 0 && btnIndicator.Effect != null)

{

TicTacToe[0, 1] = 1;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 0 && btnIndicator2.Effect != null)

{

TicTacToe[0, 1] = 10;

WinninSub();

}

else if (btn.Margin.Left == 500 && btn.Margin.Top == 0 && btnIndicator.Effect != null)

{

TicTacToe[0, 2] = 1;

WinninSub();

}

else if (btn.Margin.Left == 500 && btn.Margin.Top == 0 && btnIndicator2.Effect != null)

{

TicTacToe[0, 2] = 10;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 250 && btnIndicator.Effect != null)

{

TicTacToe[1, 0] = 1;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 250 && btnIndicator2.Effect != null)

{

TicTacToe[1, 0] = 10;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 250 && btnIndicator.Effect != null)

{

TicTacToe[1, 1] = 1;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 250 && btnIndicator2.Effect != null)

{

TicTacToe[1, 1] = 10;

WinninSub();

}

else if (btn.Margin.Left == 500 && btn.Margin.Top == 250 && btnIndicator.Effect != null)

{

TicTacToe[1, 2] = 1;

WinninSub();

}

else if (btn.Margin.Left == 500 && btn.Margin.Top == 250 && btnIndicator2.Effect != null)

{

TicTacToe[1, 2] = 10;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 500 && btnIndicator.Effect != null)

{

TicTacToe[2, 0] = 1;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 500 && btnIndicator2.Effect != null)

{

TicTacToe[2, 0] = 10;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 500 && btnIndicator.Effect != null)

{

TicTacToe[2, 1] = 1;

WinninSub();

}

else if (btn.Margin.Left == 250 && btn.Margin.Top == 500 && btnIndicator2.Effect != null)

{

TicTacToe[2, 1] = 10;

WinninSub();

}

else if (btn.Margin.Left == 500 && btn.Margin.Top == 500 && btnIndicator.Effect != null)

{

TicTacToe[2, 2] = 1;

WinninSub();

}

else if (btn.Margin.Left == 0 && btn.Margin.Top == 0 && btnIndicator2.Effect != null)

{

TicTacToe[2, 2] = 10;

WinninSub();

}

}

//Using a Timer for a flickering effect, a string to refer to side of which it should flicker, an integer value to understand who won or if it is a draw

private DispatcherTimer Timer;

private string FlickeringLine;

public int WinningPlayer;

private void WinninSub()

{

//This is the mechanism code that activates as soon as someone wins or both players draw

int Row0 = TicTacToe[0, 0] + TicTacToe[0, 1] + TicTacToe[0, 2];

int Row1 = TicTacToe[1, 0] + TicTacToe[1, 1] + TicTacToe[1, 2];

int Row2 = TicTacToe[2, 0] + TicTacToe[2, 1] + TicTacToe[2, 2];

int Column0 = TicTacToe[0, 0] + TicTacToe[1, 0] + TicTacToe[2, 0];

int Column1 = TicTacToe[0, 1] + TicTacToe[1, 1] + TicTacToe[2, 1];

int Column2 = TicTacToe[0, 2] + TicTacToe[1, 2] + TicTacToe[2, 2];

int Diagonal1 = TicTacToe[0, 0] + TicTacToe[1, 1] + TicTacToe[2, 2];

int Diagonal2 = TicTacToe[0, 2] + TicTacToe[1, 1] + TicTacToe[2, 0];

Timer = new DispatcherTimer(); //Setting an instance of the timer

Timer.Interval = TimeSpan.FromMilliseconds(100); //Setting the interval in every 100 milliseconds

Timer.Tick += Timer\_Tick; //Assigning the timer's tick subroutine

//These If statements are used to understand who effectively won or if there is a draw

if (Row0 == 30 || Row1 == 30 || Row2 == 30 || Column0 == 30 || Column1 == 30 || Column2 == 30 || Diagonal1 == 30 || Diagonal2 == 30)

{

WinningPlayer = 0; //Player1 wins

}

else if (Row0 == 3 || Row1 == 3 || Row2 == 3 || Column0 == 3 || Column1 == 3 || Column2 == 3 || Diagonal1 == 3 || Diagonal2 == 3)

{

WinningPlayer = 1; //Player2 wins

}

else

{

WinningPlayer = 2; //Draw

}

//These series of Switch statements basically state which Row, Column, Diagonal is the winning line and starts the timer

switch (Row0)

{

case 30:

case 3:

FlickeringLine = "Row0";

Timer.Start();

break;

}

switch (Row1)

{

case 30:

case 3:

FlickeringLine = "Row1";

Timer.Start();

break;

}

switch (Row2)

{

case 30:

case 3:

FlickeringLine = "Row2";

Timer.Start();

break;

}

switch (Column0)

{

case 30:

case 3:

FlickeringLine = "Column0";

Timer.Start();

break;

}

switch (Column1)

{

case 30:

case 3:

FlickeringLine = "Column1";

Timer.Start();

break;

}

switch (Column2)

{

case 30:

case 3:

FlickeringLine = "Column2";

Timer.Start();

break;

}

switch (Diagonal1)

{

case 30:

case 3:

FlickeringLine = "Diagonal1";

Timer.Start();

break;

}

switch (Diagonal2)

{

case 30:

case 3:

FlickeringLine = "Diagonal2";

Timer.Start();

break;

}

//This is the case it is a draw, in the case every button in the grid has a nought or a cross and in the case of which every row, column, diagonal is not equal to 3 or 30, it will defenitely be a draw;

if (R0C0.Content != null && R0C1.Content != null && R0C2.Content != null && R1C0.Content != null && R1C1.Content != null && R1C2.Content != null && R2C0.Content != null && R2C1.Content != null && R2C2.Content != null)

{

if (Row0 != 30 && Row0 != 3 && Row1 != 30 && Row1 != 3 && Row0 != 30 && Row0 != 3 && Column0 != 30 && Column0 != 3 && Column1 != 30 && Column1 != 3 && Column2 != 30 && Column2 != 3 && Diagonal1 != 30 && Diagonal1 != 3 && Diagonal2 != 30 && Diagonal2 != 3)

{

Winnerban = new WinnerBan(WinningPlayer, lblName1.Text, lblName2.Text, Convert.ToInt32(ScoreContainer1.Content), Convert.ToInt32(ScoreContainer2.Content));

Winnerban.Show();

Hide();

}

}

}

private int Increment = 0;

private void Timer\_Tick(object sender, EventArgs e)

{

Increment++;

//Declaring a SolidColourBrush for colour of the Flickering Effect

SolidColorBrush FlickeringColour = new SolidColorBrush(Color.FromRgb(247, 224, 20));

//Based on the string FlickeringLine different buttons will flicker

switch (FlickeringLine)

{

case "Row0":

switch (Increment % 2)

{

case 0:

R0C0.Background = FlickeringColour;

R0C1.Background = FlickeringColour;

R0C2.Background = FlickeringColour;

break;

default:

R0C0.Background = Brushes.Transparent;

R0C1.Background = Brushes.Transparent;

R0C2.Background = Brushes.Transparent;

break;

}

break;

case "Row1":

switch (Increment % 2)

{

case 0:

R1C0.Background = FlickeringColour;

R1C1.Background = FlickeringColour;

R1C2.Background = FlickeringColour;

break;

default:

R1C0.Background = Brushes.Transparent;

R1C1.Background = Brushes.Transparent;

R1C2.Background = Brushes.Transparent;

break;

}

break;

case "Row2":

switch (Increment % 2)

{

case 0:

R2C0.Background = FlickeringColour;

R2C1.Background = FlickeringColour;

R2C2.Background = FlickeringColour;

break;

default:

R2C0.Background = Brushes.Transparent;

R2C1.Background = Brushes.Transparent;

R2C2.Background = Brushes.Transparent;

break;

}

break;

case "Column0":

switch (Increment % 2)

{

case 0:

R0C0.Background = FlickeringColour;

R1C0.Background = FlickeringColour;

R2C0.Background = FlickeringColour;

break;

default:

R0C0.Background = Brushes.Transparent;

R1C0.Background = Brushes.Transparent;

R2C0.Background = Brushes.Transparent;

break;

}

break;

case "Column1":

switch (Increment % 2)

{

case 0:

R0C1.Background = FlickeringColour;

R1C1.Background = FlickeringColour;

R2C1.Background = FlickeringColour;

break;

default:

R0C1.Background = Brushes.Transparent;

R1C1.Background = Brushes.Transparent;

R2C1.Background = Brushes.Transparent;

break;

}

break;

case "Column2":

switch (Increment % 2)

{

case 0:

R0C2.Background = FlickeringColour;

R1C2.Background = FlickeringColour;

R2C2.Background = FlickeringColour;

break;

default:

R0C2.Background = Brushes.Transparent;

R1C2.Background = Brushes.Transparent;

R2C2.Background = Brushes.Transparent;

break;

}

break;

case "Diagonal1":

switch (Increment % 2)

{

case 0:

R0C0.Background = FlickeringColour;

R1C1.Background = FlickeringColour;

R2C2.Background = FlickeringColour;

break;

default:

R0C0.Background = Brushes.Transparent;

R1C1.Background = Brushes.Transparent;

R2C2.Background = Brushes.Transparent;

break;

}

break;

case "Diagonal2":

switch (Increment % 2)

{

case 0:

R0C2.Background = FlickeringColour;

R1C1.Background = FlickeringColour;

R2C0.Background = FlickeringColour;

break;

default:

R0C2.Background = Brushes.Transparent;

R1C1.Background = Brushes.Transparent;

R2C0.Background = Brushes.Transparent;

break;

}

break;

}

//End of the effect

switch(Increment)

{

case 10:

//Stopping the timer, opening the WinningBan form

Timer.Stop();

Winnerban = new WinnerBan(WinningPlayer, lblName1.Text, lblName2.Text, Convert.ToInt32(ScoreContainer1.Content) , Convert.ToInt32(ScoreContainer2.Content));

Winnerban.Show();

Hide();

break;

}

}

private void btnClickMe\_Click(object sender, RoutedEventArgs e)

{

//Calling the random instance to have a random colour each time

Random Randomicy = new Random();

int Counter = Randomicy.Next(0, 21);

//Creating a LinearGradientBrush and defining its properties

LinearGradientBrush EpicNess = new LinearGradientBrush();

EpicNess.StartPoint = new System.Windows.Point(0, 0);

EpicNess.EndPoint = new System.Windows.Point(1, 1);

//Creating a DropShadowEffect for some controls

DropShadowEffect EpicEffect = new DropShadowEffect();

EpicEffect.BlurRadius = 15;

EpicEffect.ShadowDepth = 5;

EpicEffect.Opacity = 100;

EpicEffect.Direction = 315;

//For each colour, GradientsStops are added with their respective colours, the Lineargradientbrush and the DropShadowEffect are applied to some controls

switch (Counter)

{

case 0:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(240, 235, 54), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(115, 194, 105), 0.5));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(30, 145, 165), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(240, 235, 54);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(240, 235, 54);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 1:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 111, 105), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 204, 92), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(255, 111, 105);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(255, 111, 105);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 2:

gridBackground.Background = new SolidColorBrush(System.Windows.Media.Color.FromRgb(243, 170, 78));

YellowColumn.Fill = new SolidColorBrush(System.Windows.Media.Color.FromRgb(243, 170, 78));

YellowColumns.Fill = new SolidColorBrush(System.Windows.Media.Color.FromRgb(243, 170, 78));

btnClickMe.Foreground = new SolidColorBrush(System.Windows.Media.Color.FromRgb(243, 170, 78));

btnClickMe.BorderBrush = new SolidColorBrush(System.Windows.Media.Color.FromRgb(243, 170, 78));

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(243, 170, 78);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(243, 170, 78);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 3:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(123, 233, 246), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(240, 131, 216), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(123, 233, 246);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(123, 233, 246);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 4:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(60, 165, 92), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(181, 172, 73), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(60, 165, 92);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(60, 165, 92);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 6:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(218, 34, 255), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(151, 51, 238), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(218, 34, 255);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(218, 34, 255);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 7:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(2, 170, 176), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(0, 205, 172), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(2, 170, 176);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(2, 170, 176);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 8:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(237, 229, 116), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(225, 245, 196), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(237, 229, 116);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(237, 229, 116);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 9:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(211, 16, 39), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(234, 56, 77), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(211, 16, 39);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(211, 16, 39);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 10:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(49, 71, 85), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(38, 160, 218), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(49, 71, 85);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(49, 71, 85);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 11:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(43, 88, 118), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(78, 67, 118), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(43, 88, 118);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(43, 88, 118);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 12:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(230, 92, 0), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(249, 212, 35), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(230, 92, 0);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(230, 92, 0);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 13:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(204, 43, 94), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(117, 58, 136), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(204, 43, 94);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(204, 43, 94);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 14:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(20, 136, 204), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(43, 50, 178), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(20, 136, 204);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(20, 136, 204);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 15:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 224, 0), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(121, 159, 12), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(255, 224, 0);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(255, 224, 0);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 16:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 226, 89), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 167, 81), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(255, 226, 89);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(255, 226, 89);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 17:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(84, 51, 255), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(32, 189, 255), 0.5));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(165, 254, 203), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(84, 51, 255);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(84, 51, 255);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 18:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(0, 82, 212), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(67, 100, 247), 0.5));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(111, 177, 252), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(0, 82, 212);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(0, 82, 212);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 19:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(51, 77, 80), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(203, 202, 165), 0.5));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(51, 77, 80);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(51, 77, 80);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

case 20:

EpicNess.GradientStops.Clear();

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(0, 65, 106), 0.0));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(121, 159, 12), 0.5));

EpicNess.GradientStops.Add(new GradientStop(System.Windows.Media.Color.FromRgb(255, 224, 0), 1.0));

gridBackground.Background = EpicNess;

YellowColumn.Fill = EpicNess;

YellowColumns.Fill = EpicNess;

btnClickMe.Foreground = EpicNess;

btnClickMe.BorderBrush = EpicNess;

lblName1.Foreground = EpicNess;

lblHighestScore1.Foreground = EpicNess;

EpicEffect.Color = System.Windows.Media.Color.FromRgb(0, 65, 106);

YellowColumn.Effect = EpicEffect;

YellowColumns.Effect = EpicEffect;

btnClickMe.Effect = EpicEffect;

MyDropShadowEffectColor = Color.FromRgb(0, 65, 106);

if (lblName1.Effect != null)

{

lblName1.Effect = NeonBeLike;

}

break;

}

}

private void btnBackToForm1\_Click(object sender, RoutedEventArgs e)

{

//Opening the AskNames form

AskNames.Show();

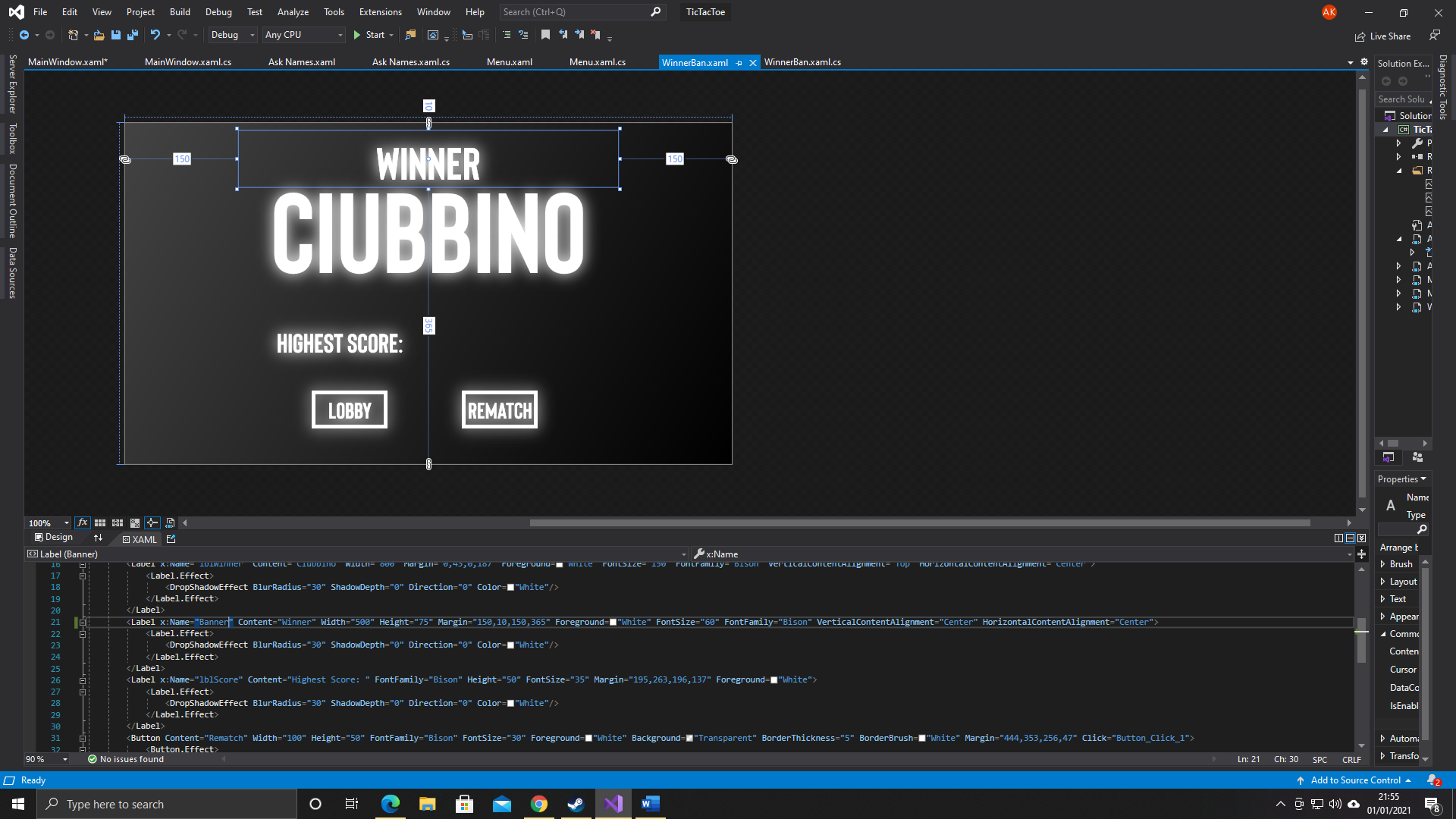
Hide();

}

}

}

**Code behind the WinnerBan Window**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace TicTacToe

{

/// <summary>

/// Interaction logic for WinnerBan.xaml

/// </summary>

public partial class WinnerBan : Window

{

Ask\_Names AskNames;

Menu menu;

int winningplayer;

string P1;

string P2;

int score1;

int score2;

public WinnerBan(int WinningPlayer, string Player1, string Player2, int Score1, int Score2)

{

P1 = Player1;

P2 = Player2;

winningplayer = WinningPlayer;

score1 = Score1;

score2 = Score2;

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

//Opening a new AskNames form

if (AskNames == null)

{

AskNames = new Ask\_Names();

}

AskNames.Show();

Hide();

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

if (Banner.Visibility == Visibility.Hidden)

{

Banner.Visibility = Visibility.Visible;

}

if (lblScore.Visibility == Visibility.Hidden)

{

lblScore.Visibility = Visibility.Visible;

}

//Opening a new menu form

if (menu == null)

{

menu = new Menu();

}

menu.Show();

Hide();

//Resetting the names and scores for the new Menu form

menu.lblName1.Text = P1;

menu.lblName2.Text = P2;

menu.ScoreContainer1.Content = score1.ToString();

menu.ScoreContainer2.Content = score2.ToString();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

if (winningplayer == 0)

{

lblWinner.Content = P1;

score1 += 1;

lblScore.Content += score1.ToString();

}

else if (winningplayer == 1)

{

lblWinner.Content = P2;

score2 += 1;

lblScore.Content += score2.ToString();

}

else if (winningplayer == 2)

{

Banner.Visibility = Visibility.Hidden;

lblWinner.Content = "Draw";

lblScore.Visibility = Visibility.Hidden;

}

}

}

}