**Lucky Roshambo** shows how to create a **Rock-Paper-Scissors** game or **Roshambo** as it is known in many parts of North America and is a good way of showing a game you’ve probably played in real-life converted into a game you can play on a Computer

## Step 1

|  |  |
| --- | --- |
|  | Follow **Setup and Start** on how to Install and/or Get Started with **Visual Studio 2019** if not already or in **Windows 10** choose **Start**, find and select **Visual Studio 2019** then from the **Get started** screen select **Create a new project** |
| A screenshot of a cell phone  Description automatically generated | Then choose **Blank App (Universal Windows)** and select **Next** and then in **Configure your new project** enter the **Project name** as **LuckyRoshambo** and select **Create** |
| A screenshot of a social media post  Description automatically generated | Finally, in **New Universal Windows Platform Project** pick the **Target version** and **Minimum version** to be at least **Windows 10, version 1903 (10.0; Build 18362)** and then select **OK** |

**Target Version** will control the most recent features of **Windows 10** your application can use. To make sure you always have the most recent version, check for any **Notifications** or **Updates** in **Visual Studio 2019**

## Step 2

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | Choose **Project** then **Add New Item...** from the **Menu** in **Visual Studio 2019** |

## Step 3

|  |  |
| --- | --- |
| A close up of a logo  Description automatically generated | Then choose **Code File** from **Add New Item** in **Visual Studio 2019**, enter the **Name** as **Library.cs** and select **Add** |

## Step 4

In the **Code** View of **Library.cs** will be displayed and in this the following should be entered:

|  |
| --- |
| using System;  using Windows.UI.Xaml;  using Windows.UI.Xaml.Controls;  using Windows.UI.Xaml.Media;  public class Library  {  private const int size = 3;  private const int lost = 0;  private const int win = 1;  private const int draw = 2;  private readonly int[,] match =  new int[size, size]  {  { draw, lost, win },  { win, draw, lost },  { lost, win, draw }  };  private readonly string[] options = new string[]  {  "\U0000270A", "\U0000270B", "\U0000270C"  };  private readonly string[] values = new string[]  {  "You Lost!", "You Win!", "You Draw!"  };  private Random \_random = new Random((int)DateTime.Now.Ticks);  } |

There are using statements to include necessary functionality. match is a int[,] is a **two-dimensional array** of values that will represent the possible combinations of the game and Random is used to create the numbers for the game

Then below the **private Random \_random = new Random((int)DateTime.UtcNow.Ticks);** line the following **method** should be entered:

|  |
| --- |
| private Viewbox Option(int value)  {  TextBlock textblock = new TextBlock()  {  Text = options[value],  IsColorFontEnabled = true,  Margin = new Thickness(2),  TextLineBounds = TextLineBounds.Tight,  FontFamily = new FontFamily("Segoe UI Emoji"),  HorizontalTextAlignment = TextAlignment.Center  };  return new Viewbox()  {  Child = textblock  };  } |

Option **method** is used to create a TextBlock to represent an option in the game

Next below the **private Viewbox Option(...) { ... }** **method** the following **method** should be entered:

|  |
| --- |
| private async void Show(string title, int option)  {  ContentDialog dialog = new ContentDialog()  {  Title = title,  PrimaryButtonText = "OK",  Content = new Viewbox()  {  Width = 150,  Height = 150,  Child = Option(option)  }  };  await dialog.ShowAsync();  } |

Show **method** is used to display a ContentDialog containing an Option within a Viewbox as the Content

After the **private async void Show(...) { ... }** **method** the following **methods** should be entered:

|  |
| --- |
| private void Choose(int player)  {  int computer = \_random.Next(0, size - 1);  int result = match[player, computer];  Show($"Computer Picked - {values[result]}", computer);  }  private void Get(StackPanel panel, int option)  {  Button button = new Button()  {  Width = 200,  Height = 200,  Tag = option,  Content = Option(option),  Margin = new Thickness(5)  };  button.Click += (object sender, RoutedEventArgs e) =>  {  button = (Button)sender;  Choose((int)button.Tag);  };  panel.Children.Add(button);  } |

Choose **method** is used to play the game by selecting a randomised choice and comparing this against the player's one and Add is used to create a Button containing an Option as the Content

Then after the **private void Get(...) { ... }** **method** the following **method** should be entered:

|  |
| --- |
| private void Layout(Grid grid)  {  grid.Children.Clear();  StackPanel panel = new StackPanel()  {  Orientation = Orientation.Horizontal,  HorizontalAlignment = HorizontalAlignment.Center  };  for (int i = 0; i < size; i++)  {  Get(panel, i);  }  grid.Children.Add(panel);  } |

Layout **method** is used to setup a StackPanel and using the Add **method**

Finally after **private void Layout(...) { ...}** **method** the following public **method** should be entered:

|  |
| --- |
| public void New(Grid grid)  {  Layout(grid);  } |

New **method** will setup the layout of the Grid using the Layout **method** and start playing the game

## Step 5

|  |  |
| --- | --- |
|  | In the **Solution Explorer** of **Visual Studio 2019** select **MainPage.xaml** |

## Step 6

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | Choose **View** then **Designer** from the **Menu** in **Visual Studio 2019** |

## Step 7

In the **Design** View and **XAML** View of **Visual Studio 2019** will be displayed, and in this between the **Grid** and **/Grid** elements enter the following **XAML**:

|  |
| --- |
| <Grid Margin="50" Name="Display"  VerticalAlignment="Center"  HorizontalAlignment="Center"/>  <CommandBar VerticalAlignment="Bottom">  <AppBarButton Icon="Page2" Label="New" Click="New\_Click"/>  </CommandBar> |

The first block of **XAML** the main user interface features a **Grid** with two **Grid** Controls within to represent the game. The second block of **XAML** is the **CommandBar** which contains **New** to start the game

## Step 8

|  |  |
| --- | --- |
|  | Choose **View** then **Code** from the **Menu** in **Visual Studio 2019** |

## Step 9

Once in the **Code** View, below the end of **public MainPage() { ... }** the following Code should be entered:

|  |
| --- |
| Library library = new Library();  private void New\_Click(object sender, RoutedEventArgs e)  {  library.New(Display);  } |

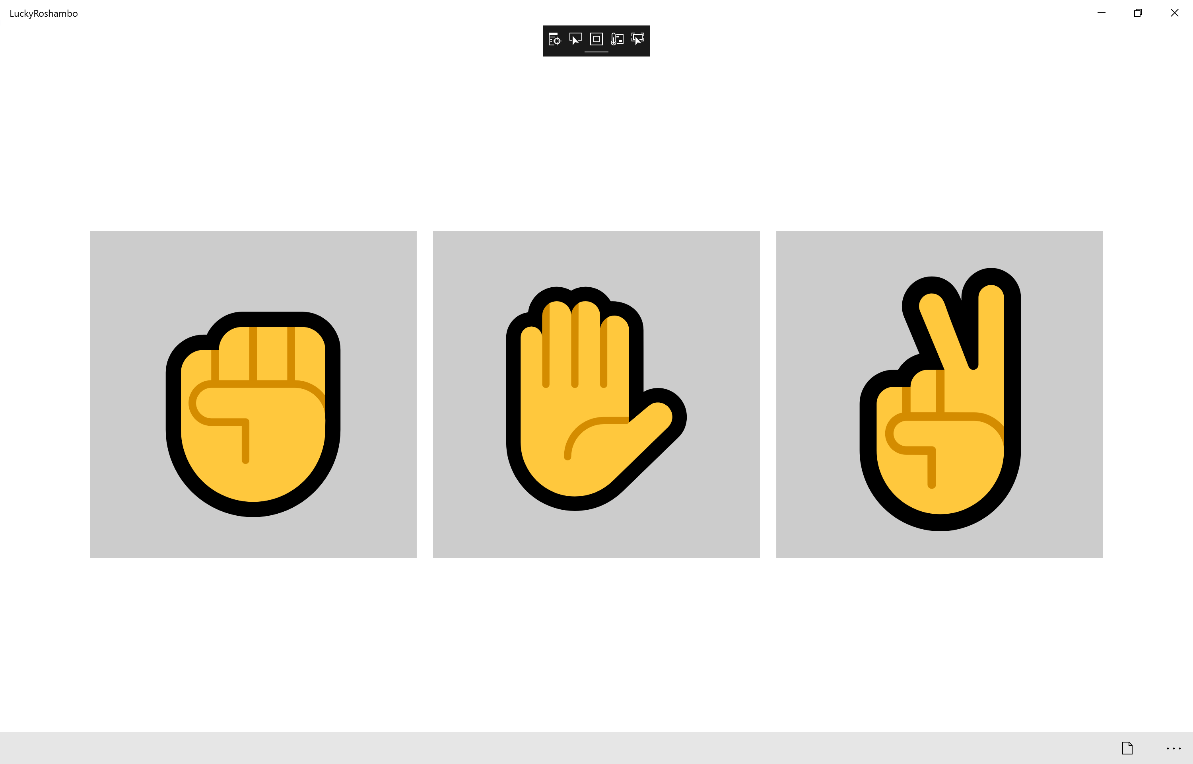
Below the **MainPage** method an instance of the Library **class** is created. In the New\_Click(...) **Event** handler will setup the game with the New **method** in the Library **class**

## Step 10

|  |  |
| --- | --- |
|  | That completes the **Universal Windows Platform** Application, in **Visual Studio 2019** select **Local Machine** to run the Application |

## Step 11

Once the Application is running you can then select **New** to start the game then press on **Rock** – the first button, **Paper** – the second button or **Scissors** – the third button, then you can see what the **Computer** selects to see if you **Win**, **Lose** or **Draw**



## Step 12

|  |  |
| --- | --- |
| A picture containing object  Description automatically generated | To Exit the Application, select the **Close** button in the top right of the Application |