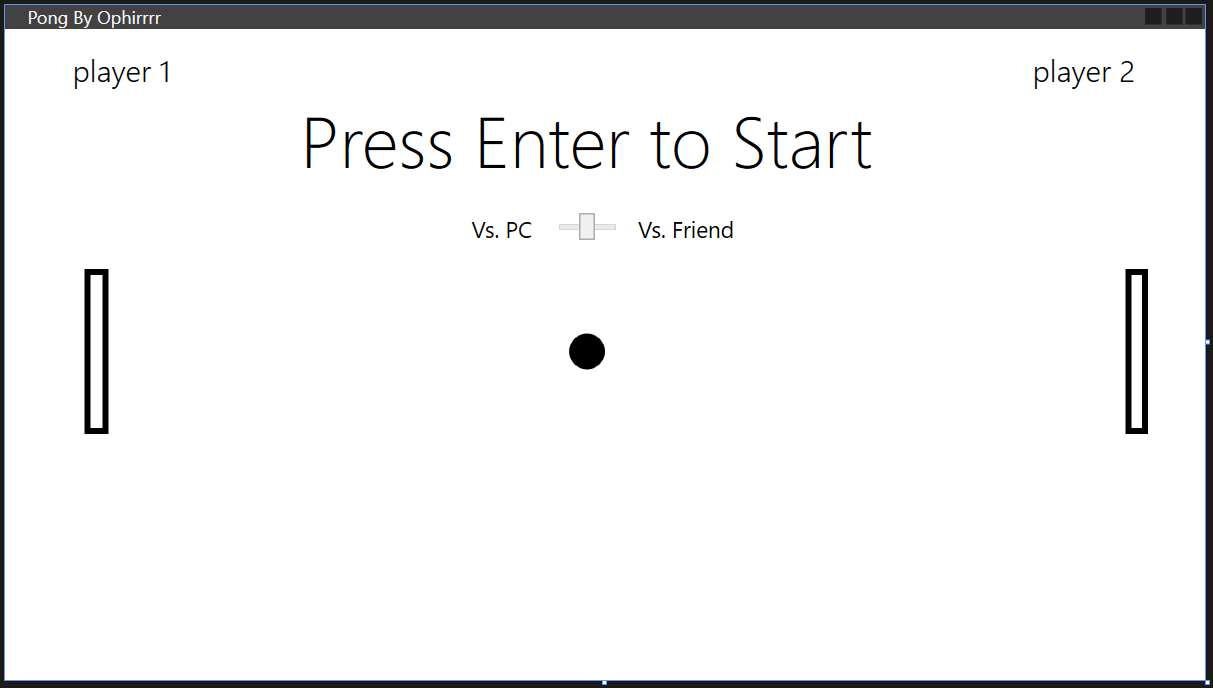
**PONG**

**מיני פרויקט מבוסס WPF – אופיר הופמן י3**

****

**Xaml Code:**

<Window

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PongWPF"

xmlns:Themes="clr-namespace:Microsoft.Windows.Themes;assembly=PresentationFramework.AeroLite" x:Name="window" x:Class="PongWPF.MainWindow"

mc:Ignorable="d"

Title="Pong By Ophirrrr" Height="450" Width="800" KeyDown="PlayerInput">

<Grid Focusable="True" KeyDown="PlayerInput">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="53\*"/>

<ColumnDefinition Width="747\*"/>

</Grid.ColumnDefinitions>

<Rectangle x:Name="right\_Tile" HorizontalAlignment="Left" Height="110" Margin="694,160,0,0" Stroke="Black" VerticalAlignment="Top" Width="15" StrokeThickness="4" Grid.Column="1"/>

<Rectangle x:Name="left\_Tile" HorizontalAlignment="Left" Height="110" Stroke="Black" VerticalAlignment="Top" Width="16" StrokeThickness="4" Margin="0,160,0,0" Grid.Column="1"/>

<Ellipse x:Name="ball" HorizontalAlignment="Left" Height="24" Stroke="Black" VerticalAlignment="Top" Width="24" StrokeThickness="18" Margin="323,203,0,0" Grid.Column="1"/>

<Label x:Name="WinLabel" Content="" HorizontalAlignment="Left" Height="76" Margin="192,47,0,0" VerticalAlignment="Top" Width="310" FontSize="48" FontWeight="UltraLight" Visibility="Hidden" Grid.Column="1"/>

<Label x:Name="startLabel" Content="Press Enter to Start" HorizontalAlignment="Left" Height="87" Margin="139,36,0,0" VerticalAlignment="Top" Width="392" FontSize="48" FontWeight="UltraLight" Grid.Column="1"/>

<Label x:Name="player1Lbl" Content="player 1" HorizontalAlignment="Left" VerticalAlignment="Top" Height="38" Width="82" FontSize="20" Margin="40,9,0,0" FontFamily="Yu Gothic UI Semilight" Grid.ColumnSpan="2"/>

<Label x:Name="player2Lbl" Content="player 2" HorizontalAlignment="Left" Margin="627,9,0,0" VerticalAlignment="Top" Height="38" Width="82" FontSize="20" FontFamily="Yu Gothic UI Semilight" Grid.Column="1"/>

<Rectangle x:Name="partition" Grid.Column="1" HorizontalAlignment="Left" Height="415" Margin="332,8,0,0" Stroke="Black" VerticalAlignment="Top" Width="6" StrokeThickness="4" Visibility="Hidden"/>

<Slider x:Name="slider" Grid.Column="1" HorizontalAlignment="Left" Height="24" Margin="311,123,0,0" VerticalAlignment="Top" Width="48" Value="5" ValueChanged="slider\_ValueChanged"/>

<Label x:Name="vsPC" Grid.Column="1" Content="Vs. PC" HorizontalAlignment="Left" Height="34" Margin="253,118,0,0" VerticalAlignment="Top" Width="53" FontSize="15"/>

<Label x:Name="vsFriend" Grid.Column="1" Content="Vs. Friend" HorizontalAlignment="Left" Height="34" Margin="364,118,0,0" VerticalAlignment="Top" Width="78" FontSize="15"/>

</Grid>

</Window>

**Xaml.Cs**

namespace PongWPF

{

enum Direction { UpRight, DownRight, DownLeft, UpLeft};

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

private bool againstPC;

private const int MinY = 5;

private const int MaxY = 300;

private const int MaxX = 795;

private const int MinX = 5;

private const int tile\_Speed = 20;

private const int ball\_speed = 3;

private Direction ball\_direction;

private int winner; // 1 - left, - right

private DispatcherTimer mainTimer;

Random rnd = new Random();

public MainWindow()

{

InitializeComponent();

// Timer Tick

mainTimer = new DispatcherTimer();

mainTimer.Interval = new TimeSpan(0, 0, 0, 0, 5);

mainTimer.Tick += new EventHandler(mainTimer\_Tick);

ball\_direction = (Direction)rnd.Next(4);

}

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

{

MoveBall();

if (againstPC)

MoveLeftTilePC();

CheckBallBorders();

CheckTileHit();

if (CheckWin())

{

mainTimer.Stop();

ball.Visibility = Visibility.Hidden;

partition.Visibility = Visibility.Hidden;

WinLabel.Content = "Player " + winner + " Won!";

WinLabel.Visibility = Visibility.Visible;

}

}

private bool CheckWin()

{

if (ball.Margin.Left < left\_Tile.Margin.Left)

{

winner = 2; // right player won

return true;

}

if (ball.Margin.Left > right\_Tile.Margin.Left + right\_Tile.Width)

{

winner = 1; // left player won

return true;

}

return false;

}

private void CheckTileHit()

{

double tileTop = left\_Tile.Margin.Top;

double tileBottom = left\_Tile.Margin.Top + left\_Tile.Height;

double tileX = left\_Tile.Margin.Left + left\_Tile.Width;

double ballTop = ball.Margin.Top;

double ballBottom = ball.Margin.Top + ball.Height;

double ballX = ball.Margin.Left;

// Check Left Tile hit

if (ballTop > tileTop && ballBottom < tileBottom && ballX < tileX+1)

{

ChangeBallDirection();

return;

}

// Check right Tile hit

tileTop = right\_Tile.Margin.Top;

tileBottom = right\_Tile.Margin.Top + right\_Tile.Height;

tileX = right\_Tile.Margin.Left;

if (ballTop > tileTop && ballBottom < tileBottom && ballX > tileX - 25)

{

ChangeBallDirection();

}

}

private void MoveBall()

{

if (ball\_direction == Direction.UpRight)

ball.Margin = new Thickness(ball.Margin.Left + ball\_speed, ball.Margin.Top - ball\_speed, 0, 0);

else if (ball\_direction == Direction.DownRight)

ball.Margin = new Thickness(ball.Margin.Left + ball\_speed, ball.Margin.Top + ball\_speed, 0, 0);

else if (ball\_direction == Direction.DownLeft)

ball.Margin = new Thickness(ball.Margin.Left - ball\_speed, ball.Margin.Top + ball\_speed, 0, 0);

else if (ball\_direction == Direction.UpLeft)

ball.Margin = new Thickness(ball.Margin.Left - ball\_speed, ball.Margin.Top - ball\_speed, 0, 0);

}

private void CheckBallBorders()

{

if (BallReachedBorder())

{

ChangeBallDirection();

}

}

private void ChangeBallDirection()

{

ball\_direction = (Direction)((int)(ball\_direction + 1) % 4);

}

private bool BallReachedBorder()

{

return ball.Margin.Top > 390 || ball.Margin.Top < 10;

}

private void MoveRightTile(System.Windows.Input.KeyEventArgs e)

{

// Right Tile Movement

if (e.Key == Key.Up && right\_Tile.Margin.Top > MinY)

right\_Tile.Margin = new Thickness(right\_Tile.Margin.Left, right\_Tile.Margin.Top - tile\_Speed, 0, 0);

else if (e.Key == Key.Down && right\_Tile.Margin.Top < MaxY)

right\_Tile.Margin = new Thickness(right\_Tile.Margin.Left, right\_Tile.Margin.Top + tile\_Speed, 0, 0);

}

private void MoveLeftTileUser(System.Windows.Input.KeyEventArgs e)

{

// Left Tile Movement

if (e.Key == Key.W && left\_Tile.Margin.Top > MinY)

left\_Tile.Margin = new Thickness(left\_Tile.Margin.Left, left\_Tile.Margin.Top - tile\_Speed, 0, 0);

else if (e.Key == Key.S && left\_Tile.Margin.Top < MaxY)

left\_Tile.Margin = new Thickness(left\_Tile.Margin.Left, left\_Tile.Margin.Top + tile\_Speed, 0, 0);

}

private void MoveLeftTilePC()

{

// Move left Tile accordingly to the ball Y position

double ballTop = ball.Margin.Top;

double ballBottom = ball.Margin.Top + ball.Height;

double tileTop = left\_Tile.Margin.Top;

double tileBottom = left\_Tile.Margin.Top + left\_Tile.Height;

if (ballTop < tileTop && ((int)ball\_direction == 3 || (int)ball\_direction == 2))

left\_Tile.Margin = new Thickness(left\_Tile.Margin.Left, left\_Tile.Margin.Top - tile\_Speed, 0, 0);

else if (ballBottom > tileBottom && ((int)ball\_direction == 3 || (int)ball\_direction == 2))

left\_Tile.Margin = new Thickness(left\_Tile.Margin.Left, left\_Tile.Margin.Top + tile\_Speed, 0, 0);

}

private void PlayerInput(object sender, System.Windows.Input.KeyEventArgs e)

{

MoveRightTile(e); // Check right tile movement

if (!againstPC) // Only if not against PC

MoveLeftTileUser(e); // Check left tile movement

// Start Game

if (e.Key == Key.Enter)

{

mainTimer.Start();

slider.Visibility = Visibility.Hidden;

vsFriend.Visibility = Visibility.Hidden;

vsPC.Visibility = Visibility.Hidden;

startLabel.Visibility = Visibility.Hidden;

partition.Visibility = Visibility.Visible;

}

// End Game & exit window

else if (e.Key == Key.Escape)

{

mainTimer.Stop();

window.Close();

}

}

public void Slider\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

{

// Check if player wants to play against PC ("AI") or Friend (W, S)

if (slider.Value == 10) // Against a friend

{

againstPC = false;

vsFriend.FontWeight = FontWeights.Bold;

vsPC.FontWeight = FontWeights.Normal;

}

else if (slider.Value == 0) // Against PC

{

againstPC = true;

vsPC.FontWeight = FontWeights.Bold;

vsFriend.FontWeight = FontWeights.Normal;

}

}

}

}