Simple Pong 5: Scoring and Speed

Time required: 30 minutes

Contents

Comment each line of code as shown in the tutorials and other code examples.

Follow all directions carefully and accurately. Think of the directions as a minimum requirement.

Simple Pong Project Sequence

To give you an idea of what this project entails, here is the project sequence. If you are being creative with the project, you might want to wait until you get to that stage of the project.

1. Moving Ball
2. Bouncing Ball
3. Keyboard Input
4. Collision Detection
5. Scoring and Speed

6. Sound

Creative Contest

There are different ways an application can be programmed. The Creative Contest is to see who can program the most creative version of Pong, while keeping all the minimum functionality in the tutorials. Don't do anything with sound yet, as we do that in the last iteration of this program.

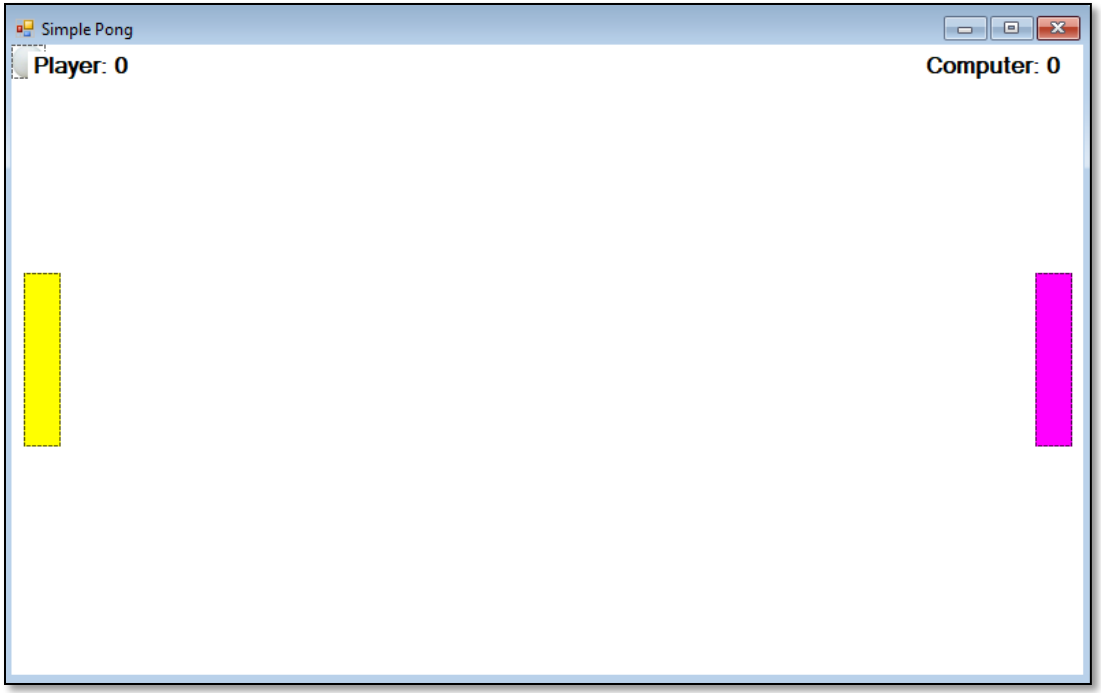
Grand Prize: Bragging rights!

Program Description

What fun is a game if we don't keep score? We also speed up the ball each time either player gets a point. That makes the game more challenging and fun!

The Code

1. Add 2 labels to the form. **lblComputerScore** on the right, and **lblPlayerScore** on the left. Change font and set the text properties.


Simple Pong
Player: 0
Computer: 0

1. Add the following form level variable to the existing ones.

```
public partial class Form1 : Form
{
    const int WIN = 10;              // How many games it takes to win
    const int INCREASE_SPEED = 1;    // Increase game speed factor
    int playerScore = 0;             // Track player score
    int computerScore = 0;           // Track computer score
    const int DISTANCE_FROM_EDGE = 10;  // Distance of computer paddle from edge of playing field
    int ComputerPaddleSpeed = 5;        // Set the Computer paddle speed in pixels
    bool GoUp;      // Boolean used to store keyboard up arrow status
    bool GoDown;    // Boolean used to store keyboard down arrow status
    const int PLAYER_PADDLE_SPEED = 8;  // Set the Player paddle speed in pixels
    const int TIME_DELAY = 10;  // How "fast" the game runs
    const int SPEED_X = 2;      // X movement speed constant
    const int SPEED_Y = 2;      // Y movement speed constant
    int MoveX = SPEED_X;        // Set horizontal movement/speed of the ball in pixels, based on SPEED_X
    int MoveY = SPEED_Y;        // Set vertical movement/speed of the ball in pixels, based on SPEED_Y
```

2. Add the following **KeepScore** method to keep score and speed up the game.

```
// Logic to keep score and increase the speed of the game
private void keepScore()
{
    // If the Player missed the ball, computer wins
    if (Ball.Left < 0)
    {
        Ball.Left = ClientSize.Width / 2;    // Move ball to middle of the screen
        MoveX = +MoveX;                      // Reverse the ball's direction
        MoveX = MoveX + INCREASE_SPEED;      // Increase speed
        if (MoveY < 0)
        {
            MoveY = MoveY - INCREASE_SPEED;  // Increase speed
        }
        else
        {
            MoveY = MoveY + INCREASE_SPEED;  // Increase speed
        }
        computerScore++;    // Increase computer score
        MoveX = SPEED_X;      // Reset speed for next round
        MoveY = SPEED_Y;
    }
```

```
    // If computer missed the ball, player wins
    if (Ball.Left + Ball.Width > ClientSize.Width)
    {
        Ball.Left = ClientSize.Width / 2;    // Move ball to middle of the screen
        MoveX = +MoveX;                      // Reverse the ball's direction
        MoveX = MoveX - INCREASE_SPEED;      // Increase speed
        if (MoveY < 0)
        {
            MoveY = MoveY - INCREASE_SPEED;  // Increase speed
        }
        else
        {
            MoveY = MoveY + INCREASE_SPEED;  // Increase speed
        }
        playerScore++;      // Increase player score
        MoveX = SPEED_X;      // Reset speed for next round
        MoveY = SPEED_Y;
    }
    // Update the score labels
    lblPlayerScore.Text = "Player: " + playerScore.ToString();
    lblComputerScore.Text = "Computer: " + computerScore.ToString();
    // End the game
    if (playerScore >= WIN)
    {
        MessageBox.Show("You Won!");
        playerScore = 0;
        computerScore = 0;
    }
    if (computerScore >= WIN)
    {
        MessageBox.Show("The Computer Won!");
        playerScore = 0;
        computerScore = 0;
    }
}
```

3. The **GameLoop** method should now look like this.

```
// Method with boolean (Created) that moves ball while this object/class is running (true)
public void GameLoop()
{
    // Infinite "Game Loop"
    while (this.Created)
    {
        moveBall();                         // Move the ball
        keepScore();                        // Keep score
        detectCollision();                  // Detect a collision
        movePaddle();                       // Move the paddles
        Refresh();                          // Repaint the form
        // Processes all events in the Thread queue so the program animation doesn't stop during refresh
        Application.DoEvents();
        // Pause the foreground or program thread for 10 ms
        System.Threading.Thread.Sleep(TIME_DELAY);
    }
}
```

4. Press F5 to debug and run the program.

As the round progresses the ball moves faster until someone wins. Score is kept. Can you beat the computer?