МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ автономное ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

(ФГБОУ ВПО МПУ)



Кафедра СМАРТ-технологии

Направление: Киберфизические системы

Группа 231-328

Отчет по учебной(проектной) практике

Студент: Поволоцкий Никита Кириллович

Место прохождения практики: ФГУП НАМИ

Руководитель от предприятия: \_\_\_\_\_\_\_\_\_\_\_\_

Москва

2024

**Введение**

В ходе прохождения практики на предприятии ФГУП

«НАМИ» в центре программного обеспечения в период с «06» мая по «01» июня 2024 г. Были изучены методы создания компьютерных программ с помощью инструментов С# Winforms. В результате практики студент разработал программное обеспечение для интерактивной симуляции спортивной игры (Air hockey). Программа выполнена на языке программирования С# в среде разработки Visual Studio c помощью платформы .NET.

Целью настоящего проекта являлась разработка приложения для визуализации движения круглого объекта (шайбы) в плоскости экрана с расчетом упругой модели столкновений с круглыми препятствиями (битами) и стенками-границами зоны движения.

Характеристики приложения:

1) Размер поля движения: не менее 640х480 пикселей;

2) Форма поля: прямоугольный;

3) На левой и правой границе поля находятся отверстия заданного размера (ворота);

4) Модель движения: упругая, без трения, без изменения скорости шайбы;

Функциональность приложения:

1) Симуляция движения шайбы по полю;

2) Возможность настройки размеров движущегося объекта, бит и ворот в режиме остановки симуляции;

3) Возможность настройки скорости шайбы в режиме остановки симуляции;

4) Сброс игры в режиме остановки симуляции;

4) Установка (перемещение) препятствий должна выполняться нажатием кнопок клавиатуры (левая бита) и движением указателя мыши (правая бита);

4) Биты должны свободно перемещаться в пределах своей половины поля;

5) При проходе шайбы в створ ворот должно засчитываться игровое очко одному из участников и проводиться сброс шайбы с середины поля;

**Основная часть**

Для визуализации требуемого процесса и рендера соответствующей графики будет использоваться такой элемент формы, как picturebox, в свойствах я указал его размер согласно заданию (640;480). Также были добавлены такие элементы управления как кнопки, ползунки для управления элементами симуляции (рис. 1).

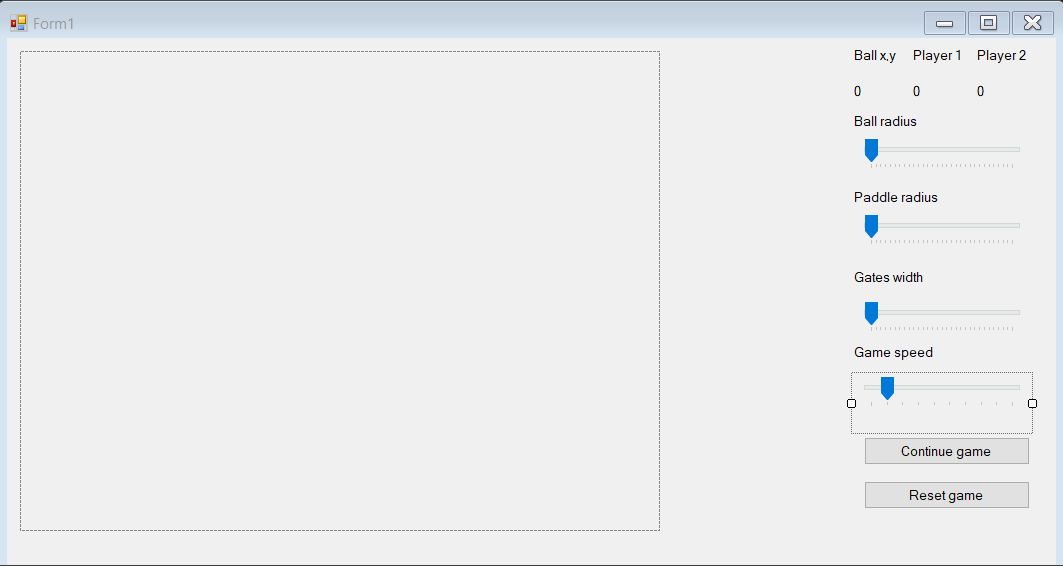


Рис. 1 – расположение элементов на форме

В ходе планирования и получения новой информации о доступных инструментах была создана базовая схема работы игры (рис. 2), в которой выделены два основных компонента: таймер и обработчики событий.

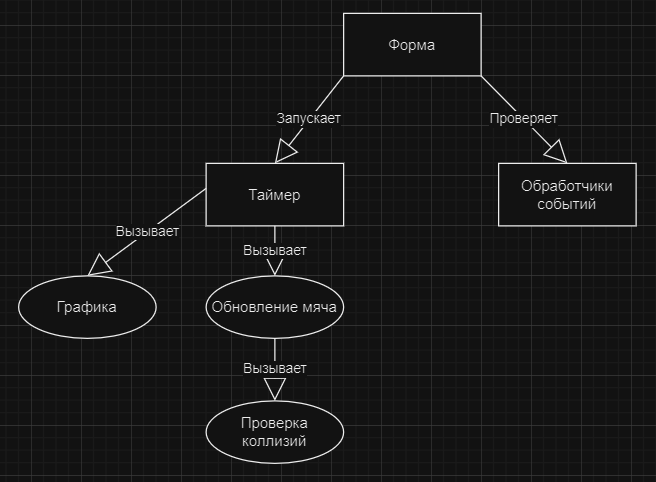
Таймер отвечает за генерацию множества событий, включая отрисовку графики. Обработчики событий, в свою очередь, занимаются обработкой нажатий клавиш и движений мыши по экрану.

Рис. 2

Строки подключения обработчика событий для движения мыши и нажатия клавиш указаны в приложении 2.

Проверка коллизии шайбы с битой происходит в соответствующей функции CheckCollision(прил. 1) методом rectangle.intersectsWith, принцип работы которого с эллипсами изображен на рис. 3.

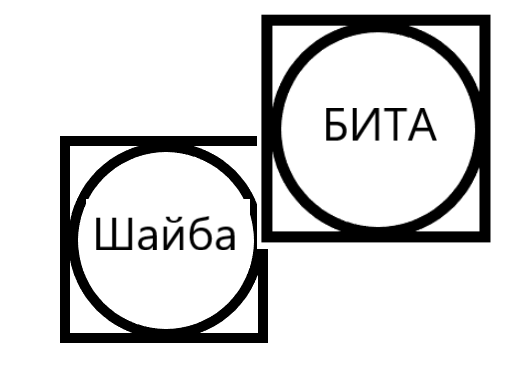


Рис. 3 – иллюстрация принципа проверки коллизии биты и шайбы

Приложение после компиляции показано на рис. 4.

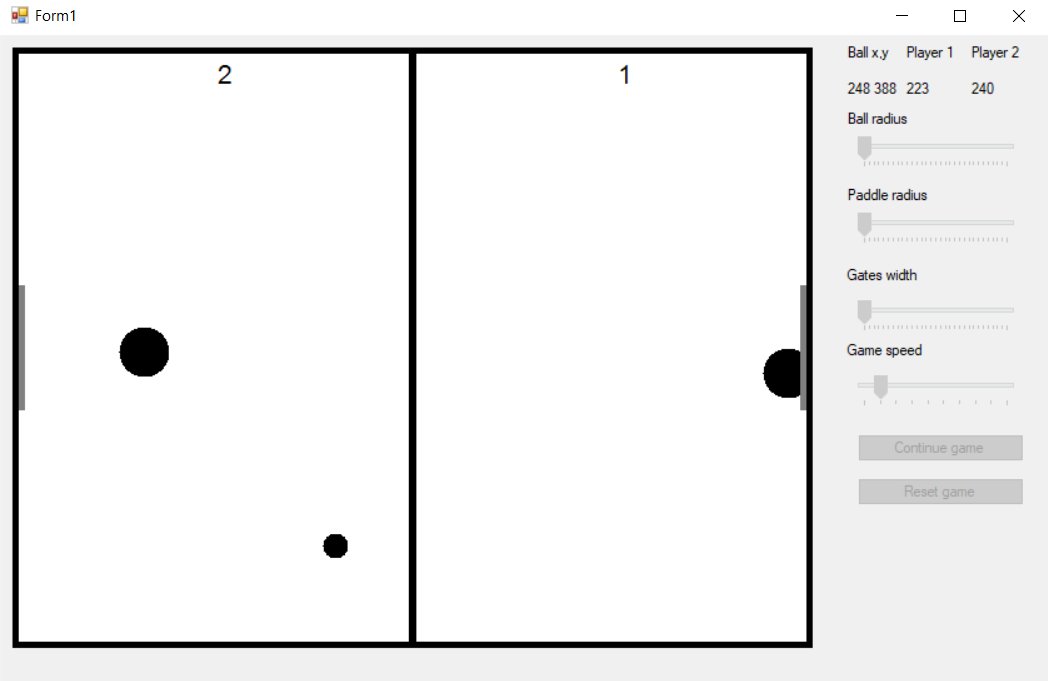


Рис. 4 – рабочее приложение.

Сброс шайбы производится в центре поля с произвольным направлением вектора движения(Приложение 2).

Также при нажатии на кнопку заново (Рис.1) проводится сброс накопленных очков (игра начинается сначала).

Результаты тестирования показали, что приложение работает корректно, а симуляция движения шайбы на поле является интерактивной и соответствует задаче.

Полный код созданных процедур указан в приложениях.

**Заключение**

В ходе прохождения практики на предприятии ФГУП «НАМИ» в центре программного обеспечения были изучены методы создания ПО с помощью инструментов Visual Studio и платформы .NET. В результате практики было разработано приложение для визуализации движения круглого объекта (шайбы) в плоскости экрана с расчетом упругой модели столкновений с круглыми препятствиями (битами) и стенками-границами зоны движения. Приложение функционирует корректно, что подтверждается результатами тестирования. Приложение может быть использовано для обучающих целей и для развлечения.

**Список литературы:**

1.Макконнелл, С. "Совершенный код", Издательство "БХВ-Петербург", 2019.

2.Шилдт, Г. "C# 4.0: Полное руководство", Издательство "Вильямс", 2017.

3. Programming C# 8.0 by Ian Griffiths, Published by O’Reilly Media, Inc., 2020

4.Таненбаум, Э. "Современные операционные системы", Издательство "Питер", 2018.

5.Скиена, С. "Алгоритмы. Руководство по разработке", Издательство "Вильямс", 2019.

**ПРИЛОЖЕНИЯ**

Приложение 1

private bool CheckCollision(int paddleY, int paddleX)

{

return new Rectangle(ballx, bally, ballRadius\*2, ballRadius\*2).IntersectsWith(new Rectangle(paddleX, paddleY, paddleWidth, paddleWidth));

}

Приложение 2

this.pictureBox1.MouseMove += new System.Windows.Forms.MouseEventHandler(this.pictureBox1\_MouseMove); //обработчик движения мышью

this.KeyDown += new System.Windows.Forms.KeyEventHandler(this.Form1KeyDown);// обработчик нажатия клавиш

Приложение 3

Код формы целиком:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Timers;

using System.Windows.Forms;

using Timer = System.Windows.Forms.Timer;

namespace Practice

{

public partial class Form1 : Form

{

Bitmap buffer;

Graphics graphics;

Timer timer = new Timer();

int FPS = 120;

int player1y, player1x, player2y, player2x;

int player1Score = 0, player2Score = 0;

int ballx, bally;

int ballRadius = 10;

int paddleWidth = 40;

int gateWidth = 10;

int gateHeight = 100;

int gatey;

int ballSpeedX = 2, ballSpeedY = 2;

Random random = new Random();

public Form1()

{

InitializeComponent();

pictureBox1.Width = 640;

pictureBox1.Height = 480;

buffer = new Bitmap(pictureBox1.Width, pictureBox1.Height);

graphics = Graphics.FromImage(buffer);

pictureBox1.Image = buffer;

timer.Interval = 1000 / FPS;

timer.Tick += TimerCallback;

timer.Enabled = true;

gatey = pictureBox1.Height / 2 - gateHeight / 2;

ResetGame();

}

private void ResetGame()

{

ballx = pictureBox1.Width / 2 - ballRadius;

bally = pictureBox1.Height / 2 - ballRadius;

ballSpeedX = random.Next(2) == 0 ? ballSpeedX : -ballSpeedX;

ballSpeedY = random.Next(2) == 0 ? ballSpeedY : -ballSpeedY;

player1y = pictureBox1.Height / 2;

player2y = pictureBox1.Height / 2;

player1x = gateWidth;

player2x = pictureBox1.Width - paddleWidth;

timer.Start();

}

private void TimerCallback(object sender, EventArgs e)

{

Ballxy.Text = ballx.ToString()+ " " + bally.ToString();

Player1.Text = player1y.ToString();

Player2.Text = player2y.ToString();

UpdateBall();

DrawGame();

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

if (timer.Enabled == false) ballRadius = trackBar1.Value;

}

private void trackBar2\_Scroll(object sender, EventArgs e)

{

if (timer.Enabled == false) paddleWidth = trackBar2.Value;

}

private void trackBar3\_Scroll(object sender, EventArgs e)

{

if (timer.Enabled == false) gateHeight = trackBar3.Value\*10;

}

private void button1\_Click(object sender, EventArgs e)

{

timer.Enabled = true;

trackBar1.Enabled = false;

trackBar2.Enabled = false;

trackBar3.Enabled = false;

trackBar4.Enabled = false;

button1.Enabled = false;

button2.Enabled = false;

}

private void button2\_Click(object sender, EventArgs e)

{

ResetGame();

}

private void trackBar4\_Scroll(object sender, EventArgs e)

{

ballSpeedX = trackBar4.Value;

ballSpeedY = trackBar4.Value;

}

private void UpdateBall()

{

ballx += ballSpeedX;

bally += ballSpeedY;

// Столкновение с верхней и нижней стенками

if (bally <= 0 || bally+ballRadius\*2 >= pictureBox1.Height)

ballSpeedY = -ballSpeedY;

// Столкновение с боковыми стенками

if (ballx <= 0 || ballx + ballRadius\*2 >= pictureBox1.Width)

{

ballSpeedX = -ballSpeedX;

}

// Столкновение с битами

if (CheckCollision(player1y, player1x) || CheckCollision(player2y, player2x))

ballSpeedX = -ballSpeedX;

// Проверка на гол

if (GateEnter() && ballx < pictureBox1.Width / 2)

{

player2Score++;

ResetGame();

}

if (GateEnter() && ballx > pictureBox1.Width / 2)

{

player1Score++;

ResetGame();

}

}

private bool CheckCollision(int paddleY, int paddleX)

{

return new Rectangle(ballx, bally, ballRadius\*2, ballRadius\*2).IntersectsWith(new Rectangle(paddleX, paddleY, paddleWidth, paddleWidth));

}

private bool GateEnter()

{

return ((ballx <= gateWidth || ballx + ballRadius >= pictureBox1.Width - gateWidth) && (bally >= gatey && bally+ballRadius\*2 <= gatey+gateHeight));

}

private void DrawGame()

{

graphics.Clear(Color.White);

//Рисуем линию раздления

graphics.DrawLine(new Pen(Color.Black, 6), pictureBox1.Width / 2, 0, pictureBox1.Width / 2, pictureBox1.Height);

// Рисуем биты

graphics.FillEllipse(Brushes.Black, new Rectangle(player1x, player1y, paddleWidth, paddleWidth));

graphics.FillEllipse(Brushes.Black, new Rectangle(player2x, player2y, paddleWidth, paddleWidth));

// Рисуем шайбу

graphics.FillEllipse(Brushes.Black, ballx, bally, ballRadius\*2, ballRadius\*2);

// Рисуем ворота

graphics.FillRectangle(Brushes.Gray, new Rectangle(0, gatey, gateWidth, gateHeight));

graphics.FillRectangle(Brushes.Gray, new Rectangle(pictureBox1.Width - gateWidth, gatey, gateWidth, gateHeight));

// Рисуем счет

graphics.DrawString(player1Score.ToString(), new Font("Arial", 16), Brushes.Black, new Point(pictureBox1.Width / 4, 10));

graphics.DrawString(player2Score.ToString(), new Font("Arial", 16), Brushes.Black, new Point(pictureBox1.Width \* 3 / 4, 10));

graphics.DrawRectangle(new Pen(Color.Black, 10), new Rectangle(0, 0, pictureBox1.Width, pictureBox1.Height));

pictureBox1.Invalidate();

}

private void Form1KeyDown(object sender, KeyEventArgs e)

{

int step = 10;

switch (e.KeyCode)

{

case Keys.Up:

if (player2y > 0) player2y -= step;

break;

case Keys.Down:

if (player2y < pictureBox1.Height) player2y += step;

break;

case Keys.Left:

if (player2x > pictureBox1.Width / 2) player2x -= step;

break;

case Keys.Right:

if (player2x < pictureBox1.Width) player2x += step;

break;

case Keys.Space:

timer.Enabled = !timer.Enabled;

if (timer.Enabled == false)

{

trackBar1.Enabled = true;

trackBar2.Enabled = true;

trackBar3.Enabled = true;

trackBar4.Enabled = true;

button1.Enabled = true;

button2.Enabled = true;

} else

{

trackBar1.Enabled = false;

trackBar2.Enabled = false;

trackBar3.Enabled = false;

trackBar4.Enabled = true;

button1.Enabled = false;

button2.Enabled = false;

}

break;

}

}

private void pictureBox1\_MouseMove(object sender, MouseEventArgs e)

{

if (e.X < pictureBox1.Width / 2)

{

player1y = e.Y;

player1x = e.X;

}

}

}

}