Волго-Вятский колледж информатики, финансов, права, управления.

Моделирование игры Flappy Bird

Пояснительная записка курсовой работы по дисциплине

“Основы алгоритмизации и программирования“

Выполнил студент группы ИС-21:

Лялин Иван Вячеславович

Преподаватель:

Блинова Светлана Дмитриевна

Киров, 2022

*Введение*

**Microsoft Visual Studio** — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментов. Данные продукты позволяют разрабатывать как консольные приложения, так и игры, и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms и т.д.

**Объектно‑ориентированное программирование (сокр. ООП)** — методология программирования, основанная на представлении программы в виде совокупности взаимодействующих объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

1. Цель разработки

Цель: воссоздать популярную игру «Flappy Bird» которая будет предназначена для одного игрока.

1. Постановка задачи

Нужно воссоздать игру, с физикой и оригинальным набором графических изображений.

Структура проекта

При загрузке приложения появляется окно (рисунок 1). Игрок сразу приступает к процессу игры. При поражении его счёт обнуляется и выпадает окно с выбором дальнейших действий (рисунок 2).

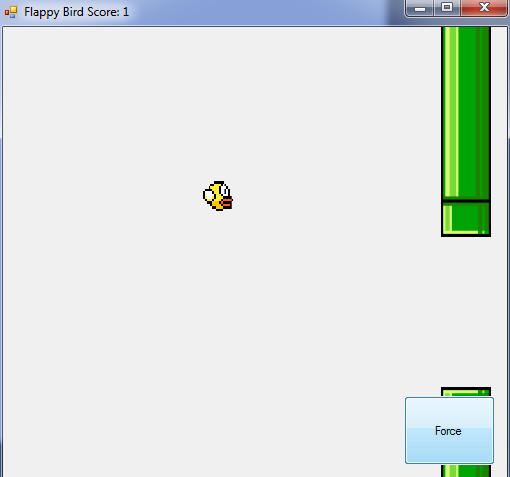


Рисунок 1.

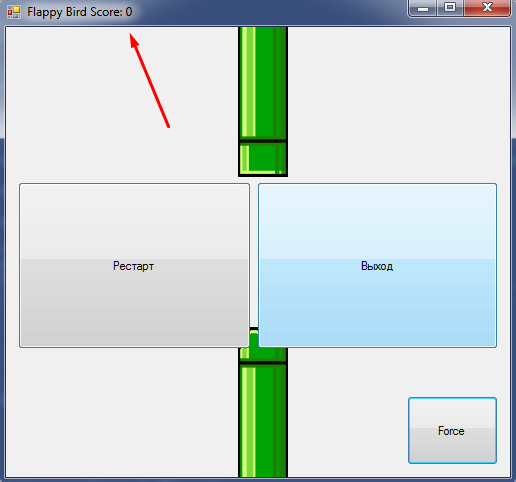


Рисунок 2.

1. Описание всех используемых объектов приложения и их свойств

На Рисунке 1 видно, что очки игрока расположены сверху на окне.

Ниже расположена кнопка “Force” которая отвечает за полёт птицы.

При проигрыше игра останавливается и даётся выбор:

начать всё с начала;

выйти из игры;

*Заключение*

В ходе работы над курсовым проектом была разработана игра FlappyBird. Были использованы средства среды визуального программирования Visual Studio 2022, языка C#. Запрограммированы событийные процедуры, реализующие счёт игры, расстановку препятствий и полёт птички.

Приложение А

Код программы

А.1 Код класса Form1

**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**ComponentModel**;**

**using** System**.**Data**;**

**using** System**.**Drawing**;**

**using** System**.**Linq**;**

**using** System**.**Text**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Windows**.**Forms**;**

**namespace** FlappyBird

**{**

**public** **partial** class Form1 **:** Form

**{**

Player bird**;** // объявляем переменные птица и труб

TheWall wall1**;**

TheWall wall2**;**

int MoveV **=** 2**;** // переменная скорости движения стен

float gravity**;** // отвечает за изменение позиции птички

**public** Form1**()**

**{**

InitializeComponent**();**

timer1**.**Interval **=** 10**;** // интервал обновления таймера

timer1**.**Tick **+=** **new** EventHandler**(**update**);** // функция которая взывается на обработке таймера

Init**();**

Invalidate**();** // нужен для отрисовки птички

**}**

**public** void Init**()**

**{**

bird **=** **new** Player**(**200**,** 200**);** // точка спавна птицы

wall1 **=** **new** TheWall**(**500**,** **-**100**,true);** //точки спавна трубы

wall2 **=** **new** TheWall**(**500**,** 300**);**

button2**.**Visible **=** **false;**

gravity **=** 0**;** // начальная гравитация птички

**this.**Text **=** "Flappy Bird Score: 0"**;**

**if** **(**bird**.**score **==** 0**)** // обнуление скорости при смерти

**{**

MoveV **=** 2**;**

**}**

timer1**.**Start**();** // запуск таймера

**}**

**private** void update**(object** sender**,** EventArgs e**)**

**{**

**if** **(**bird**.**y **>** 600**)** // если птичка умирает обнуление

**{**

bird**.**isAlive **=** **false;**

timer1**.**Stop**();**

button2**.**Visible **=** **true;**

button3**.**Visible **=** **true;**

**}**

**if** **(**Collide**(**bird**,** wall1**)** **||** Collide**(**bird**,** wall2**))** //если птичка врезается в трубу обнуление

**{**

bird**.**isAlive **=** **false;**

timer1**.**Stop**();**

button2**.**Visible **=** **true;**

button3**.**Visible **=** **true;**

**}**

**if** **(**bird**.**gravityValue **!=** 0.01f**)** // если скорость падения отрицательная то она взлетает

bird**.**gravityValue **+=** 0.005f**;**

gravity **+=** bird**.**gravityValue**;** // падение птички

bird**.**y **+=** gravity**;**

**if** **(**bird**.**isAlive**)** **{** // если птица жива то трубы двигаются

MoveWalls**();**

**}**

Invalidate**();**

**}**

**private** bool Collide**(**Player bird**,**TheWall wall1**)** // функция проверки столкновения с трубами

**{**

PointF delta **=** **new** PointF**();** // точка в которой хранится разность позиции по x и y для провеки столкновения

delta**.**X **=** **(**bird**.**x **+** bird**.**size **/** 2**)** **-** **(**wall1**.**x **+** wall1**.**sizeX **/** 2**);**

delta**.**Y **=** **(**bird**.**y **+** bird**.**size **/** 2**)** **-** **(**wall1**.**y **+** wall1**.**sizeY **/** 2**);**

**if** **(**Math**.**Abs**(**delta**.**X**)** **<=** bird**.**size **/** 2 **+** wall1**.**sizeX **/** 2**)** // проверка столкновения по x

**{**

**if** **(**Math**.**Abs**(**delta**.**Y**)** **<=** bird**.**size **/** 2 **+** wall1**.**sizeY **/** 2**)** // проверка столкновения по y

**{**

**return** **true;** // проверка условий

**}**

**}**

**return** **false;**

**}**

**private** void CreateNewWall**()** // функция генерации новых труб

**{**

**if** **(**wall1**.**x **<** bird**.**x**-**300**)** // если позиция игрока больше чем позиция труб то создаются новая трубы

**{**

Random r **=** **new** Random**();** // рандомное появление труб

int y1**;**

y1 **=** r**.**Next**(-**200**,** 000**);**

wall1 **=** **new** TheWall**(**500**,**y1**,** **true);**

wall2 **=** **new** TheWall**(**500**,** y1**+**400**);**

**this.**Text **=** "Flappy Bird Score: "**+** **++**bird**.**score**;** // после появления новых труб прибавляются очки

**}**

**}**

**private** void MoveWalls**()** // движение труб

**{**

wall1**.**x **-=** MoveV**;** // скорость движения труб

wall2**.**x **-=** MoveV**;**

**if** **(**bird**.**score **==** 3**)** MoveV **=** 3**;** // повышение скорости

**if** **(**bird**.**score **==** 7**)** MoveV **=** 4**;**

**if** **(**bird**.**score **==** 11**)** MoveV **=** 5**;**

CreateNewWall**();**

**}**

**private** void OnPaint**(object** sender**,** PaintEventArgs e**)** //отрисовка птички

**{**

Graphics graphics **=** e**.**Graphics**;**

graphics**.**DrawImage**(**bird**.**birdImg**,** bird**.**x**,** bird**.**y**,** bird**.**size**,** bird**.**size**);** // отрисовка птицы

graphics**.**DrawImage**(**wall1**.**wallImg**,** wall1**.**x**,** wall1**.**y**,** wall1**.**sizeX**,** wall1**.**sizeY**);** // отрисовка труб

graphics**.**DrawImage**(**wall2**.**wallImg**,** wall2**.**x**,** wall2**.**y**,** wall2**.**sizeX**,** wall2**.**sizeY**);**

**}**

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

**{**

**if** **(**bird**.**isAlive**)**

**{**

gravity **=** 0**;**

bird**.**gravityValue **=** **-**0.125f**;** // сила взлёта птички

**}**

**}**

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

**{**

button3**.**Visible **=** **false;**

Init**();** // рестарт

**}**

**private** void button3\_Click**(object** sender**,** EventArgs e**)**

**{**

Application**.**Exit**();** // выход

**}**

**}**

**}**

А.2 Код класса Player

**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**Linq**;**

**using** System**.**Text**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Drawing**;** // необходима для подключения картинки

**namespace** FlappyBird

**{**

class Player // этот класс отвечает за птичку

**{**

**public** float x**;** // координаты

**public** float y**;**

**public** int size**;** // переменная размера птички

**public** int score**;** // переменная счёта

**public** float gravityValue**;** // переменная которая отвечает за гравитацию в данный момент

**public** Image birdImg**;** //переменная которая отвечает за картинку

**public** bool isAlive**;**

**public** Player**(**int x**,**int y**)** // конструктор

**{**

birdImg **=** **new** Bitmap**(**"H:\\Bird.png"**);** // сама картинка и её путь

**this.**x **=** x**;** // координаты которые мы получаем на входе

**this.**y **=** y**;**

size **=** 30**;** // размер птички

gravityValue **=** 0.1f**;** // скорость падения птички

isAlive **=** **true;**

score **=** 0**;**

**}**

**}**

**}**

А.3 Код класса TheWall

**using** System**;**

**using** System**.**Collections**.**Generic**;**

**using** System**.**Linq**;**

**using** System**.**Text**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Drawing**;** // необходима для подключения картинки

**namespace** FlappyBird

**{**

class TheWall // этот класс отвечвает за трубы

**{**

**public** int x**;** // координаты появления

**public** int y**;**

**public** int sizeX**;** // переменные размеров труб

**public** int sizeY**;**

**public** Image wallImg**;** //переменная которая отвечает за картинку

**public** TheWall**(**int x**,** int y**,**bool isRotatedImage**=false)** // конструктор

**{**

wallImg **=** **new** Bitmap**(**"H:\\Pipe.png"**);** // сама картинка и её путь

**this.**x **=** x**;** // координаты которые мы получаем на входе

**this.**y **=** y**;**

sizeX **=** 50**;** // ширина и длина труб

sizeY **=** 250**;**

**if(**isRotatedImage**)**

wallImg**.**RotateFlip**(**RotateFlipType**.**Rotate180FlipX**);** // поворот верхней трубы

**}**

**}**

**}**