УДК 004

Создание простого красочного эффекта фейерверка на C#

Ульянов Егор Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается и описывается разработка простого красочного эффекта фейерверка на языке программирования C#. Практическим результатом является разработанная программа с демонстрацией эффекта.
Ключевые слова: анимации, эффект, C#, Visual Studio

Creating a simple colorful fireworks effect in C#

Ulianov Egor Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses and describes the development of a simple colorful fireworks effect in the C# programming language. The practical result is a developed program with a demonstration of the effect.
Keywords: animations, effect, C#, visual studio

Создание простого эффекта фейерверка на C# может быть интересно как начинающим, так и опытным программистам для изучения работа с графическими элементами, анимации и обработки событий мыши и клавиатуры. Это также может быть полезным для разработки игр и интерактивных приложений, которые содержат элементы визуализации и анимации.

Кроме того, это может быть полезной задачей для студентов, которые изучают программирование или компьютерную графику. Создание фейерверка может представлять собой хороший проект для обучения и оценки знаний студентов в области программирования, визуализации и анимации.

Целью данной статьи является разработка и демонстрация простого красочного эффекта фейверка в среде разработки «Visual Studio» на языке программирования C#.

В своей работе Н. Н. Додобоев, О. И. Кукарцева, Я. А. Тынченко рассмотрели вопросы появления различных языков программирования (в частности C#), определения особенностей этих языков, а также составления основных видов и классификаций языков программирования [1]. З. С. Магомадова рассмотрела языки программирования высокого уровня,

особенности, недостатки и сложности в изучении, а также описала несколько легких алгоритмов [2]. В своей работе В.Ж. Жамалова, Т.Т. Каримбаев, Ф.Р. Раймжанова, Э.С. Сатаров рассмотрели применение технологии WPF для создания тестирующей программы с мультимедийными компонентами занятий студентов физкультурников [3].

Создаем проект «Windows Forms App» и называем «Fireworks» (см. рисунок 1).

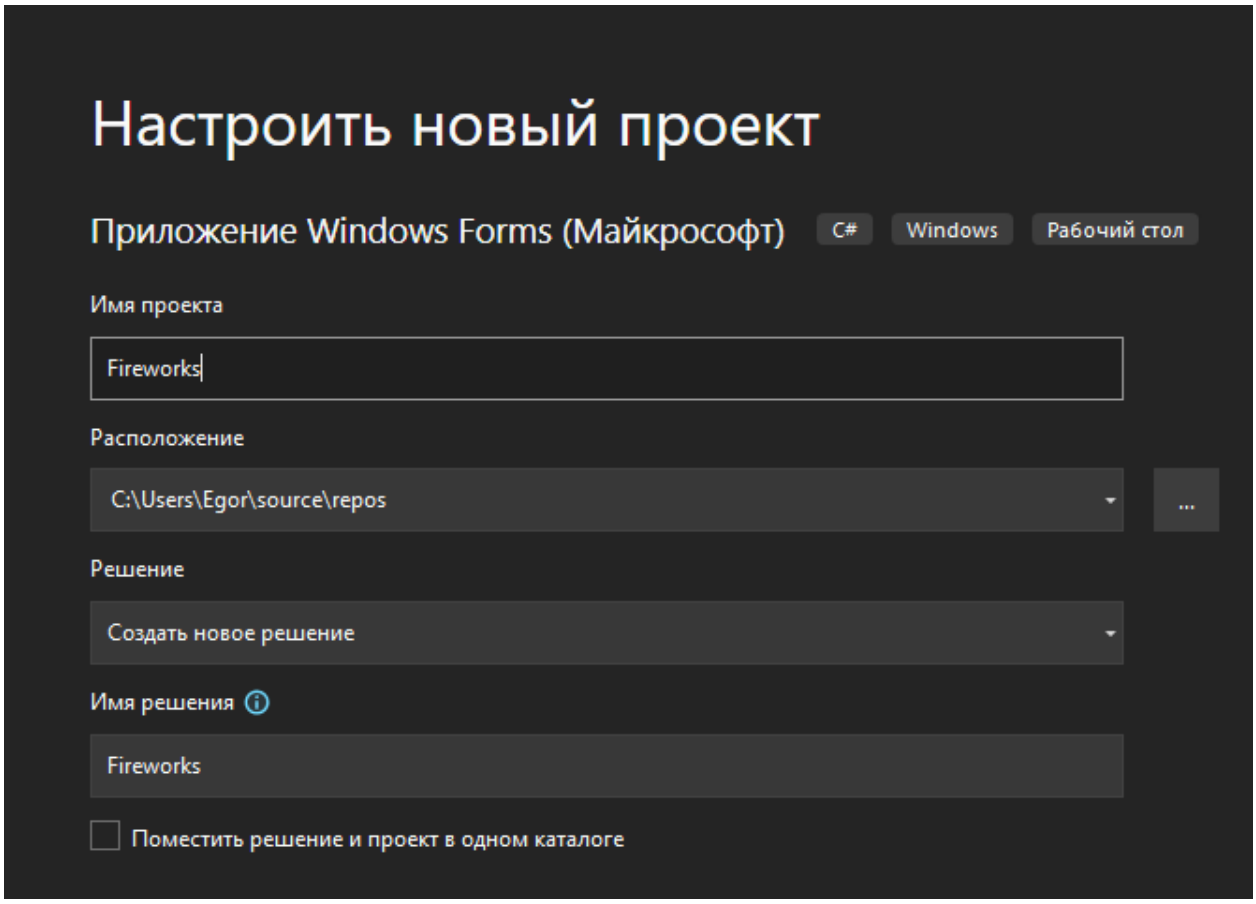

Рис. 1. Создание проекта

Далее необходимо найти асеты для заднего фона и кадров анимации фейверка (см. рисунок 2).

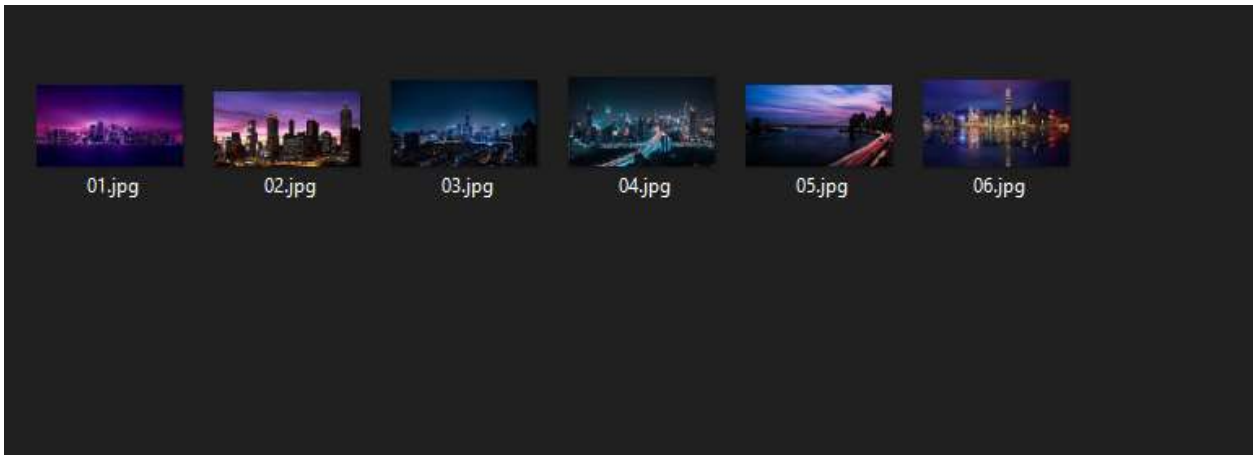

Рис. 2. Задние фоны

Рис. 3. Кадры анимации фейверка

Устанавливаем подходящий размер макета и добавляем на макет таймер, называем «animationTimer» (см. рисунок 4).

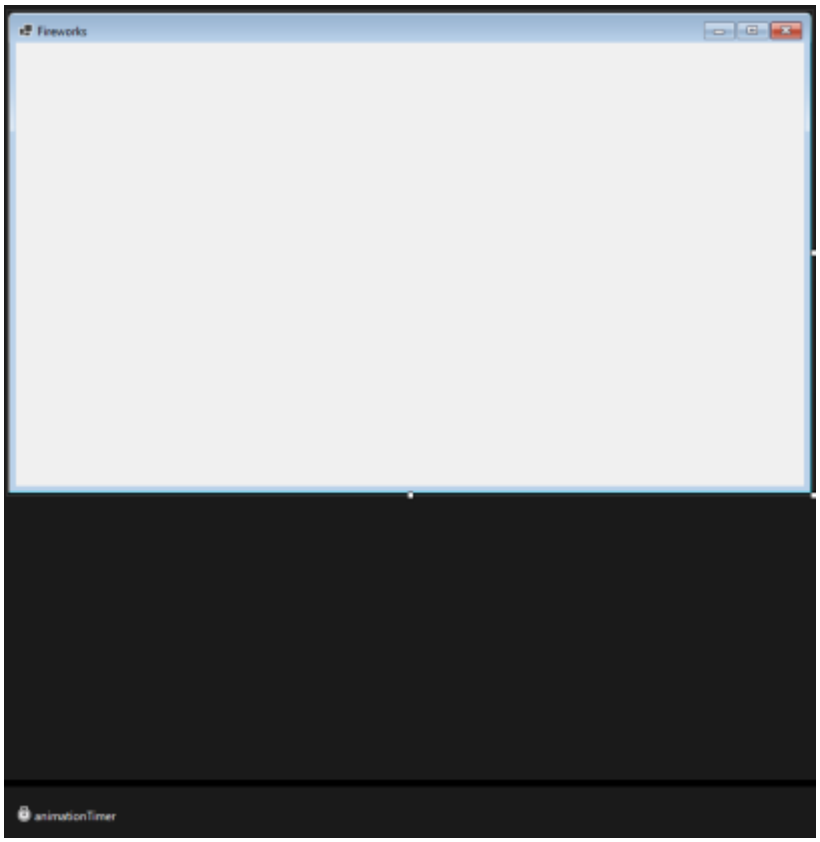

Рис. 4. Расположение элементов на макете

Добавляем класс с названием «Firework», который будет описывать анимацаию огненных фейерверков (см. рисунок 5).

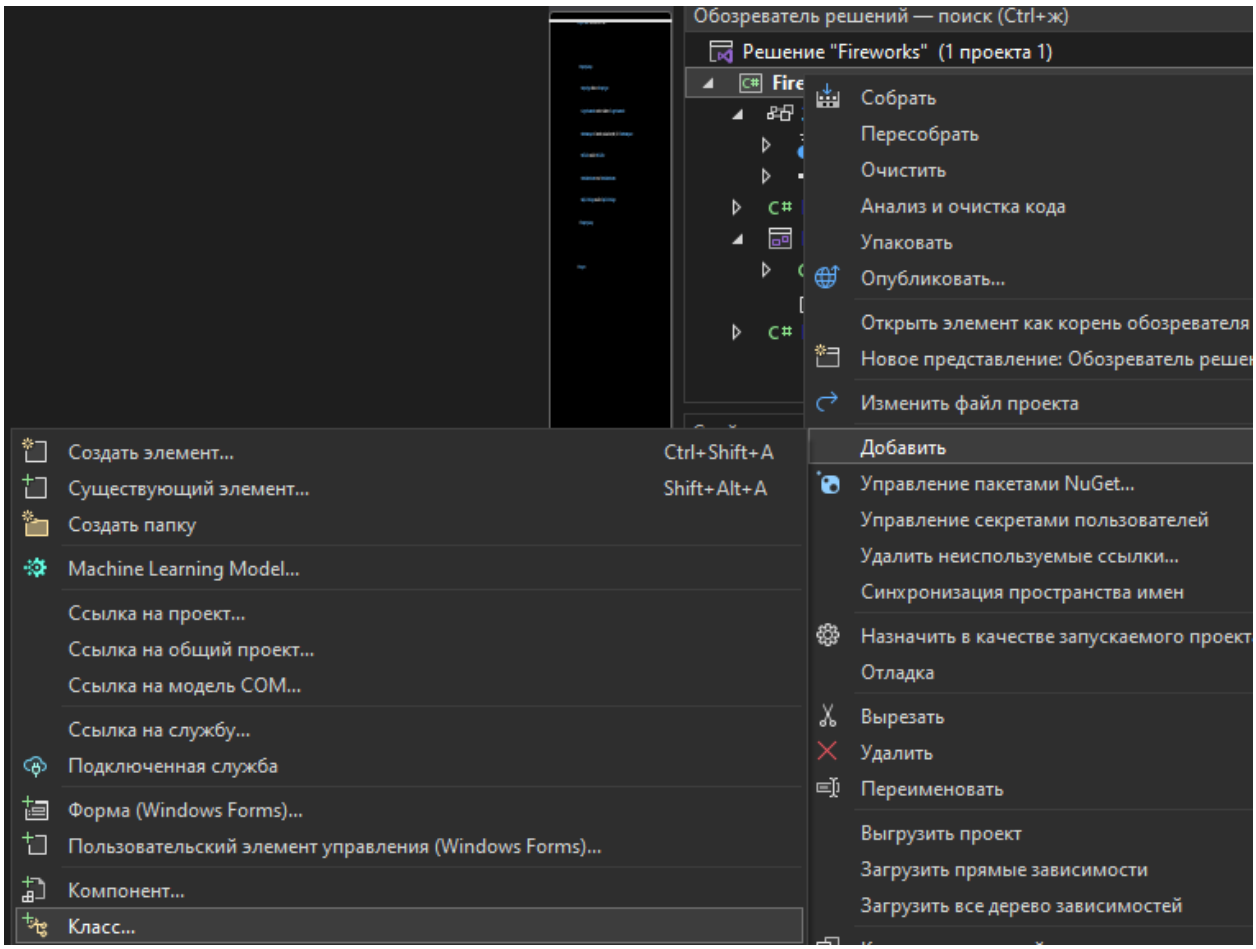

Рис. 5. Добавление класса

Переходим к написанию кода, открываем только что созданный класс. Там подключаем необходимые библиотеки, создаем несколько свойств и методов:

- «height», «width» - высота и ширина фейерверка;
- «frames» - количество кадров в анимации фейерверка;
- «current_frame» - текущий кадр в анимации;
- «position» - координаты фейерверка на экране;
- «image_location» - путь к файлам изображений для анимации фейерверка;
- «animationComplete» - флаг, который обозначает, завершена ли анимация;
- «firework» - объект класса «Image», который содержит текущий кадр анимации.

Конструктор класса инициализирует значения свойств «height», «width», «firework» и «frames», а также заполняет список «image_location» путями к файлам изображений.

Метод «AnimateFireWork()» отвечает за анимацию фейерверка. Если текущий кадр не является последним, метод увеличивает значение «current_frame» и изменяет значение свойства «firework» на объект «Image», который соответствует следующему кадру. Если текущий кадр является

последним, метод сбрасывает значение «current_frame» на 0, устанавливает значение «animationComplete» в «true», очищает список «image_location» и устанавливает значение «firework» в «null». см. рисунок 6.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Fireworks
{

    Ссылок: 1
    internal class Firework
    {
        public int height;
        public int width;
        public int frames;
        public int current_frame = 0;
        public Point position = new Point();
        public List<string> image_location = new List<string>();
        public bool animationComplete = false;
        public Image firework;

        Ссылок: 0
        public Firework()
        {
            image_location = Directory.GetFiles("images", "*.png").ToList();
            height = 200;
            width = 200;
            firework = Image.FromFile(image_location[0]);
            frames = image_location.Count;
        }


        Ссылок: 0
        public void AnimateFireWork()
        {

            if (current_frame < frames - 1)
            {
                current_frame++;
                firework = Image.FromFile(image_location[current_frame]);
            }
            else
            {
                current_frame = 0;
                animationComplete = true;
                firework = null;
                image_location.Clear();
```

Рис. 6. Заполнение класса «Firework»

Далее в исполняемый файл «Form1», который является основным классом приложения, добавляем триггер событий мыши и клавиатуры, отрисовку фейерверков и обновление анимации. Класс содержит несколько методов и свойств:

- «image_location» - путь к файлам изображений для фона и фейерверков;
- «fireworks_list» - список объектов класса Firework, которые представляют фейерверки на экране;
- «backgroundNumber» - индекс текущего изображения фона.

Конструктор класса инициализирует значения свойства «BackgroundImage» объекта «Form1» и устанавливает значение «BackgroundImageLayout».

Метод «SetUp()» загружает список файлов изображений из папки "background" и устанавливает первое изображение как фон макета.

Метод «KeyIsUp()» отслеживает нажатие клавиши на клавиатуре. Если нажата следующая клавиша, метод переключает фоновое изображение на следующее изображение в списке. Если последнее изображение в списке, метод переключает фон на первое изображение в списке.

Метод «FormMouseDown()» отслеживает нажатие кнопки мыши на форме. Если произошло нажатие, метод создает новый объект класса «Firework» и добавляет в список «fireworks_list». Координаты позиции созданного фейерверка рассчитываются на основе координаты нажатия мыши минус половина ширины и высоты создаваемого фейерверка.

Метод «FormPaintEvent()» отвечает за отрисовку фейерверков на экране. Фейерверки, которые завершили свою анимацию, не отображаются. Координаты позиции фейерверка и размеры изображения определяются свойствами объекта Firework.

Метод «AnimationTimerEvent()» отвечает за обновление анимации фейерверков. Метод проходит по списку «fireworks_list» и вызывает метод «AnimateFireWork()» для каждого объекта класса «Firework», который еще не завершил свою анимацию. Если анимация завершена, объект удаляется из списка «fireworks_list». После обновления анимации, метод вызывает метод «Invalidate()», который перерисовывает форму и ее элементы (см. рисунок 7-8).

```
namespace Fireworks
{

    Ссылок: 3
    public partial class Form1 : Form
    {

        List<string> image_location = new List<string>();
        List<Firework> fireworks_list = new List<Firework>();
        int backgroundNumber;


        Ссылок: 1
        public Form1()
        {
            InitializeComponent();
            SetUp();
        }

        Ссылок: 1
        private void SetUp()
        {
            image_location = Directory.GetFiles("background", "*.jpg").ToList();
            this.BackgroundImage = Image.FromFile(image_location[0]);
            this.BackgroundImageLayout = ImageLayout.Stretch;
        }


        Ссылок: 1
        private void KeyIsUp(object sender, KeyEventArgs e)
        {
            if (backgroundNumber < image_location.Count -1)
            {
                backgroundNumber++;
            }
            else
            {
                backgroundNumber = 0;
            }

            this.BackgroundImage = Image.FromFile(image_location[backgroundNumber]);

        }
```

Рис. 7. Класс «Form1»

```
private void FormMouseDown(object sender, MouseEventArgs e)
{
    Point mousePosition = new Point();
    mousePosition.X = e.X;
    mousePosition.Y = e.Y;

    Firework newFirework = new Firework();
    newFirework.position.X = mousePosition.X - (newFirework.width/2);
    newFirework.position.Y = mousePosition.Y - (newFirework.height/2);
    fireworks_list.Add(newFirework);
}

private void FormPaintEvent(object sender, PaintEventArgs e)
{
    foreach (Firework newFirework  in fireworks_list.ToList())
    {
        if (newFirework.animationComplete == false)
        {
            e.Graphics.DrawImage(newFirework.firework, newFirework.position.X, newFirework.position.Y, newFirework.width, newFirework.height);
        }
    }
}

private void AnimationTimerEvent(object sender, EventArgs e)
{
    if (fireworks_list != null)
    {
        foreach (Firework firework in fireworks_list.ToList())
        {

            if (firework.animationComplete == false)
            {
                firework.AnimateFirWork();
            }
            else
            {
                fireworks_list.Remove(firework);
            }
        }
    }


    this.Invalidate();
```

Рис. 8. Продолжение класса «Form1»

Теперь запускаем проект и проверяем все подготовленные функции (см. рисунок 9-13).

Рис. 9. Демонстрация фона

Рис. 10. Демонстрация переключения фона

Рис. 11. Демонстрация переключения фона

Рис. 12. Демонстрация работы анимации фейверка

Рис. 13. Демонстрация работы анимации фейверка

Таким образом, была разработан и продемонстрирован простой красочный эффект фейверка в среде разработки «Visual Studio» на языке программирования C#.

Библиографический список

1. Додобоев Н. Н., Кукарцева О. И., Тынченко Я. А. Современные языки программирования // Современные технологии: актуальные вопросы, достижения и инновации. 2014. №5. С. 81-85.
2. Магомадова З. С. Языки программирования высокого уровня // Разработка и применение наукоёмких технологий в эпоху глобальных трансформаций. 2020. №8. С. 94-96.
3. Жамалова В.Ж., Каримбаев Т.Т., Раймжанова Ф.Р., Сатаров Э.С. Программа тестирования с мультимедийными компонентами на основе WPF //Наука и инновационные технологии. 2020. С. 55-60.