УДК 004.032.26

Создание приложения с эффектом цветного дождя с использованием языка программирования C#

Ульянов Егор Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Целью данной статьи является, применение языка программирования C# и среды разработки Visual Studio 2019, для создания программы с эффектом цветного дождя. Практическим результатом является разработанное приложение.
Ключевые слова: Visual Studio, система частиц, язык программирования C#

Creating an application with the effect of colored rain using the C# programming language

Ulianov Egor Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article is to use the C# programming language and the Visual Studio 2019 development environment to create a program with the effect of colored rain. The practical result is a developed application.
Keywords: Visual Studio, particle system, C programming language#

На сегодняшний день для любого пользователя различных сервисов, таких как сайты, мобильные приложения, игры и так далее очень важно видеть красивые и продуманный интерфейс.

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 статье Ф.В. Патюченко, И.С. Слащев, А.В. Клименко, Л.А. Трегубенко были рассмотрены два подхода для создания программ на базе windows, обоснование выбора одного из них [3]. Д.З. Хасаева, А.Ю. Демин в своей работе проанализировали графическую библиотека «3Dbodies» и привели пример ее использования [4].

Цель исследования – применяя возможности среды разработки Visual Studio и языка программирования C#, создать программу с эффектом цветного дождя.

Для создания программы будем использованно программное обеспечение Visual Studio, а также язык программирования C#.

Для проекта использовалась Windows Presentation Foundation (WPF) [5], платформа пользовательского интерфейса для создания клиентских приложений для настольных систем. Платформа разработки WPF поддерживает широкий набор компонентов для разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязки данных, документы и безопасность. Эта платформа является частью платформы .NET.

Начнём разработку приложения, с создания ресурсов (рис. 1-3).

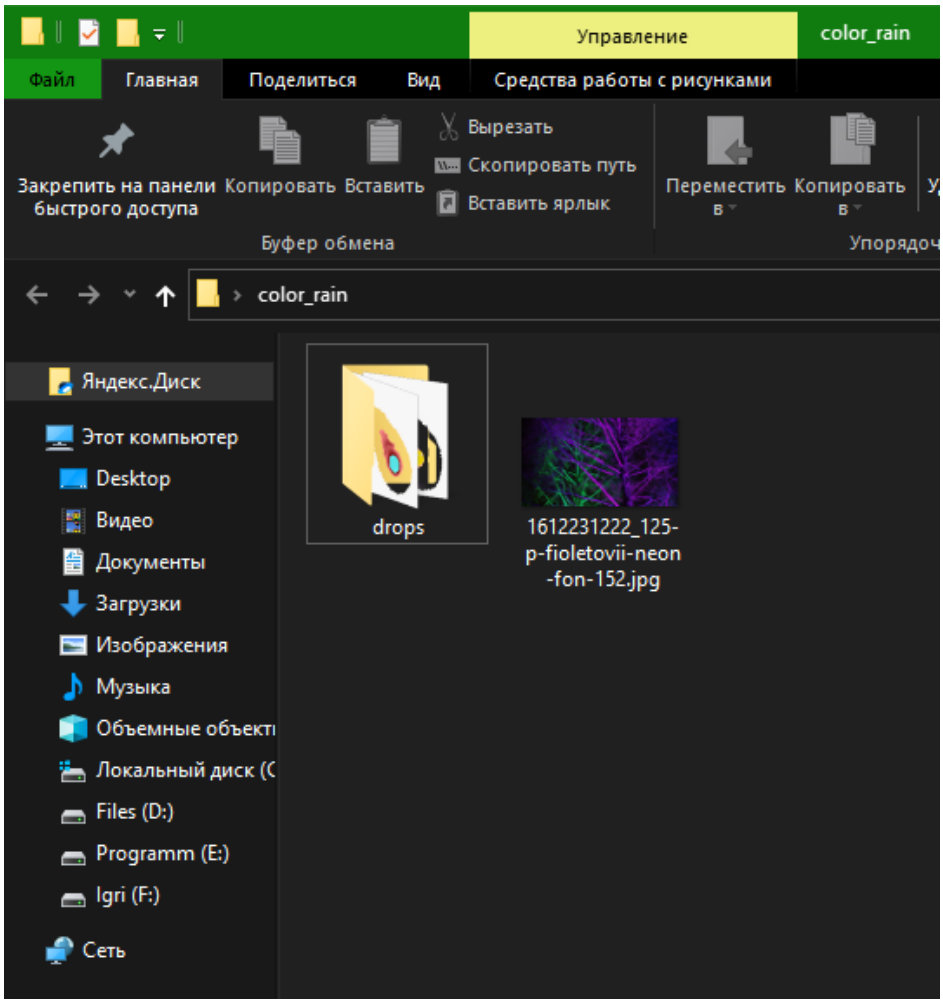

Рисунок 1. Подготовка «ассетов»

Рисунок 2. Подготовка фона

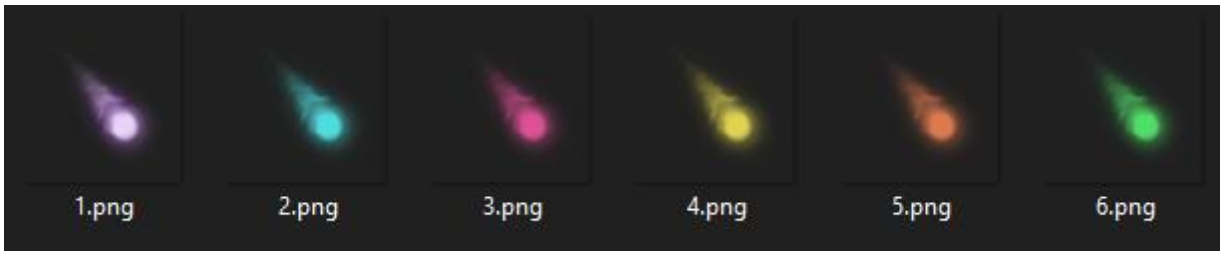

Рисунок 3. Подготовка «капель»

Переходим к созданию проекта, начинаем с называния (рис. 4).

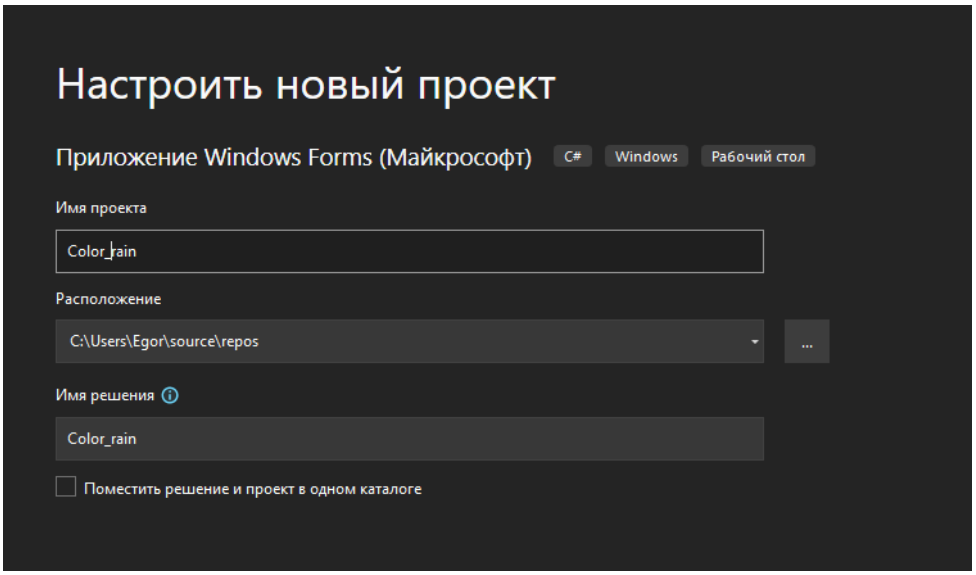

Рисунок 4. Создание проекта

Сразу добавим подготовленные ресурсы приложения, для этого жмем правой кнопкой мыши по проекту и откроем папку в проводнике (рис.5)

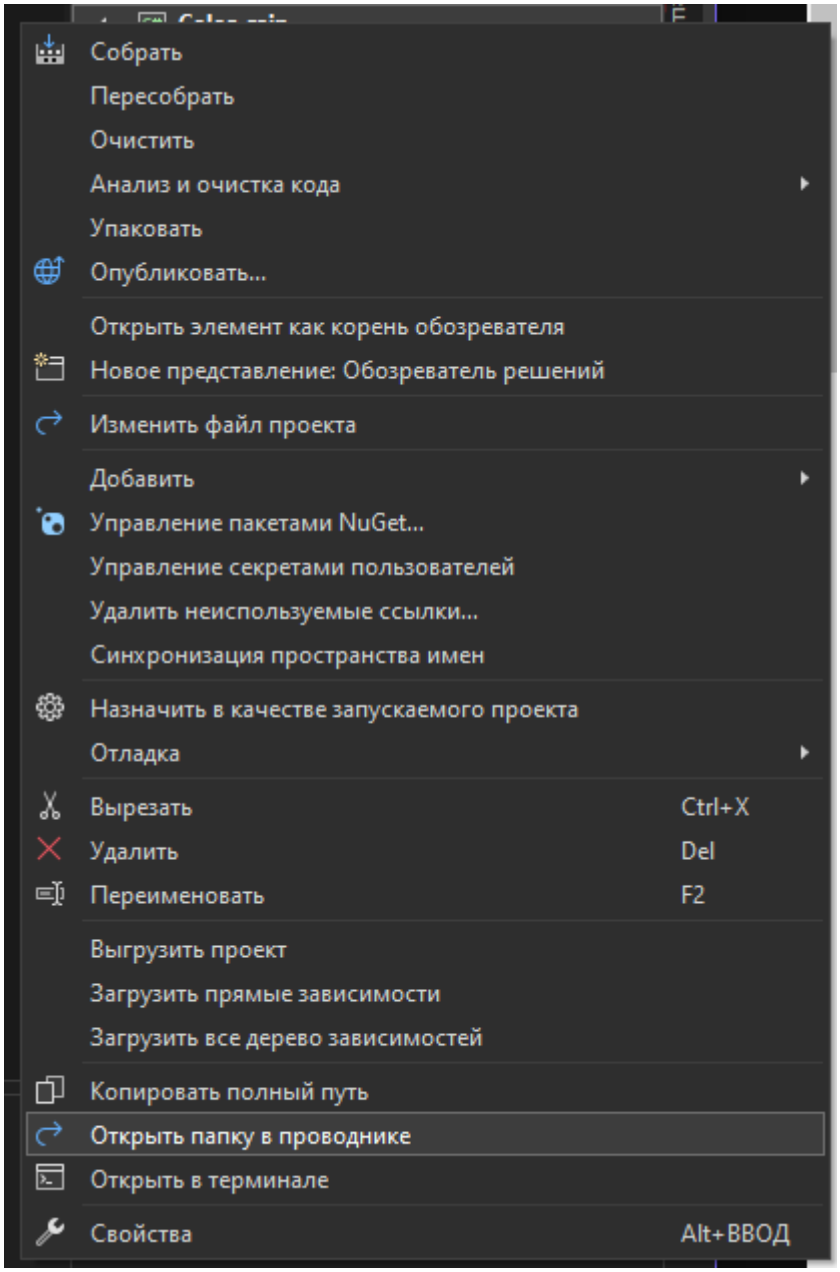

Рисунок 5. Переход в директорию проекта

Теперь переместим в проект подготовленные ресурсы приложения (рис.6).

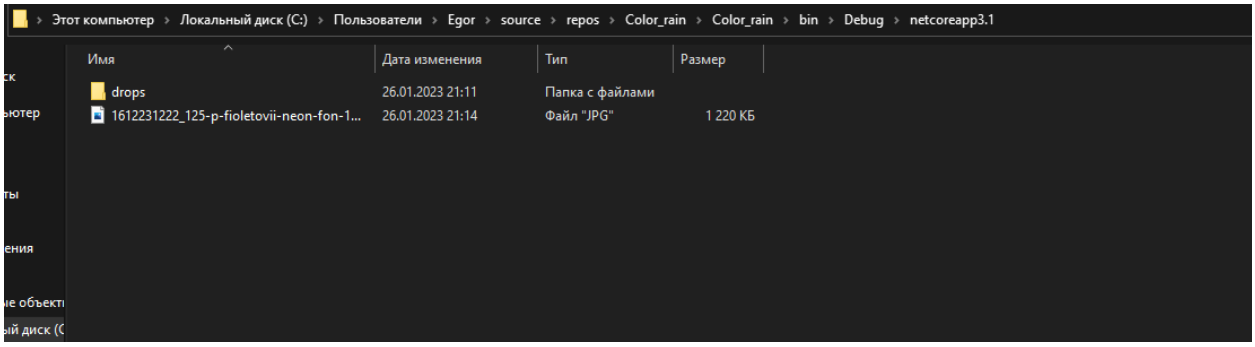

Рисунок 6. Перенос «ассетов»

Далее необходимо в свойствах формы, в пункте «backcgroundImage» импортировать фон, и для правильного отображения выбрать режим «Stretch» (рис.7).

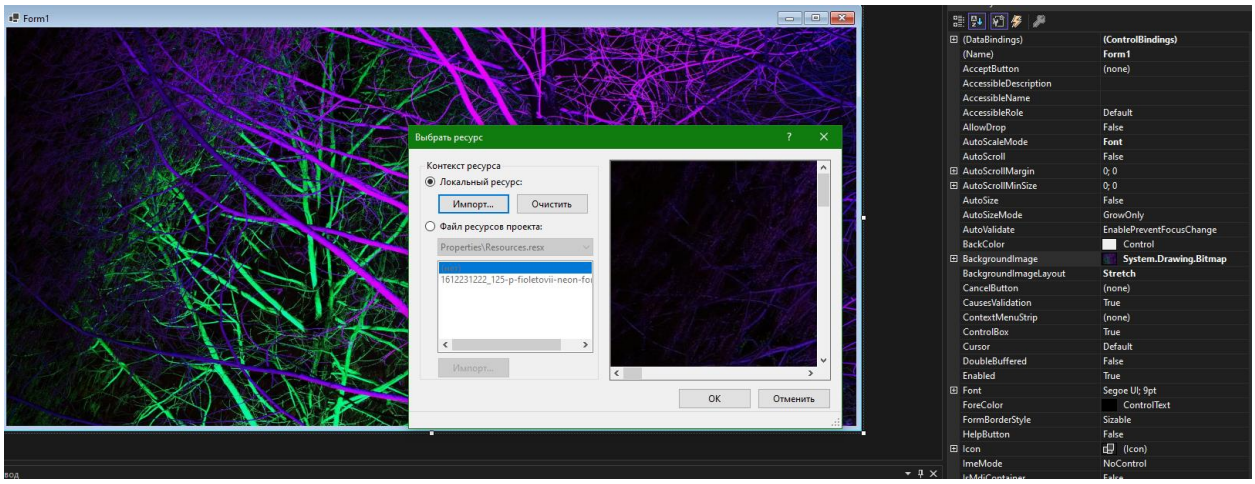

Рисунок 7. Импорт фона

Добавляем таймер в проект и называем «particleEffectTimer», в свойствах включаем таймер и меняем интервал на 20 (рис.8-9).

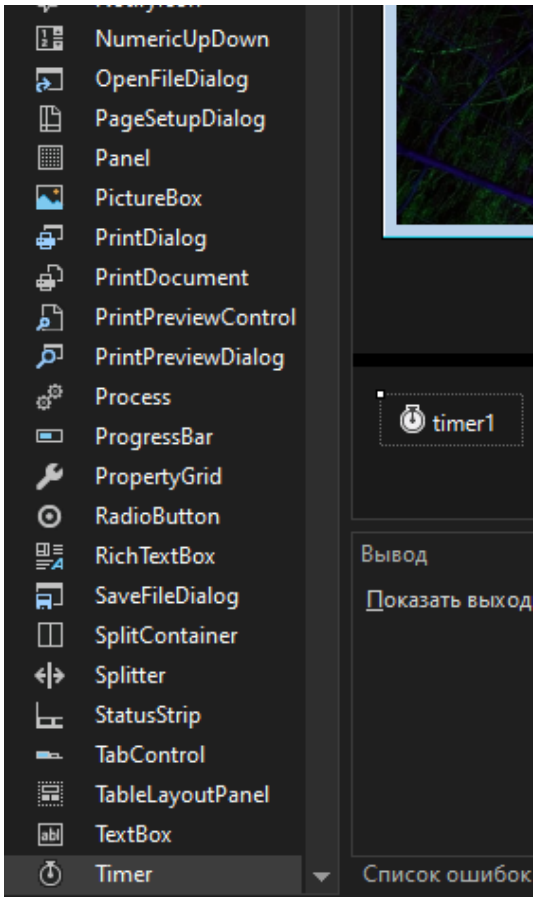

Рисунок 8. Добавление таймера в проект

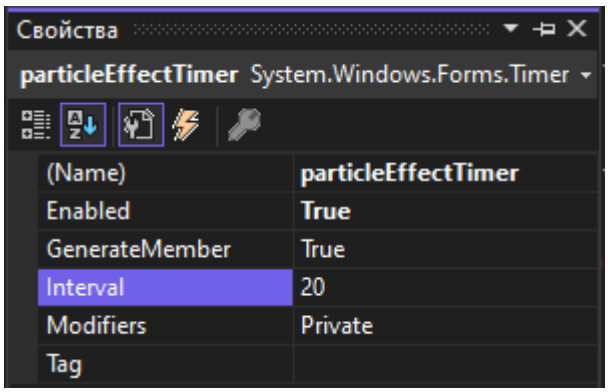

Рисунок 9. Свойства таймера

Переходим к написанию кода, для начала в событиях таймера создаем функцию «ParticleTimerEvent», а в свойствах формы функцию «FormPaintEvent» (рис.10-12).

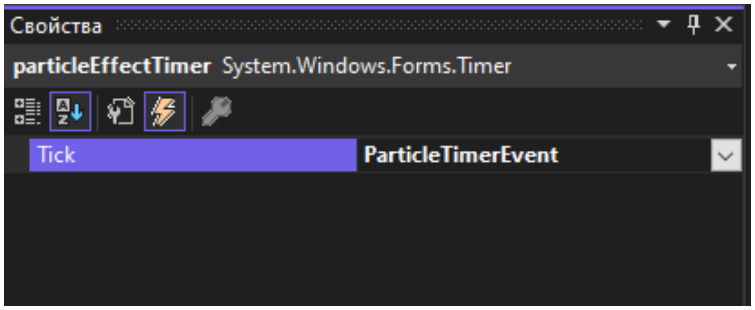

Рисунок 10. Создание функции «ParticleTimerEvent»

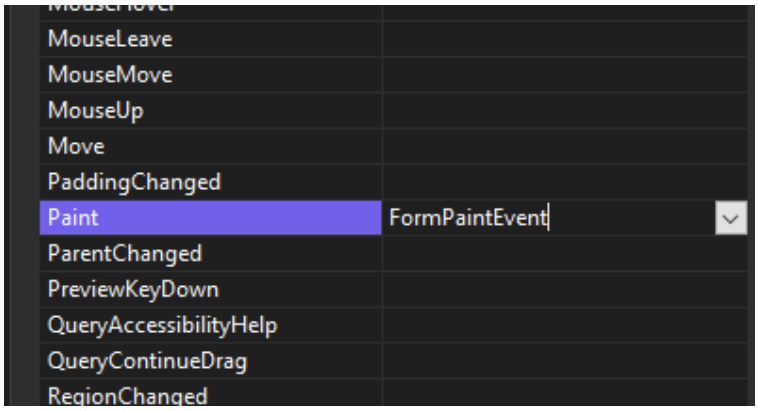

Рисунок 11. Создание функции «FormPaintEvent»

```
Ссылок: 3
public partial class Form1 : Form
{
    Ссылок: 1
    public Form1()
    {
        InitializeComponent();
    }

    Ссылок: 1
    private void ParticleTimerEvent(object sender, EventArgs e)
    {

    }

    Ссылок: 1
    private void FormPaintEvent(object sender, PaintEventArgs e)
    {

    }
}
}
```

Рисунок 12. Итоговые вид кода

Далее необходимо создать класс «Particles» (рис.13-14).

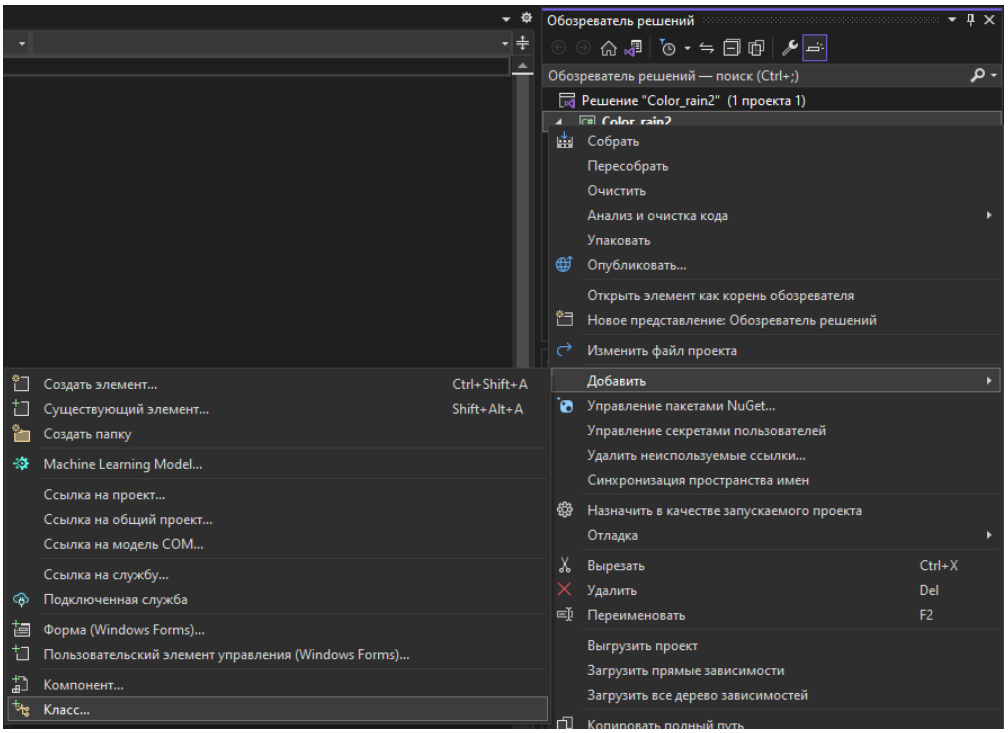

Рисунок 13. Создание класса

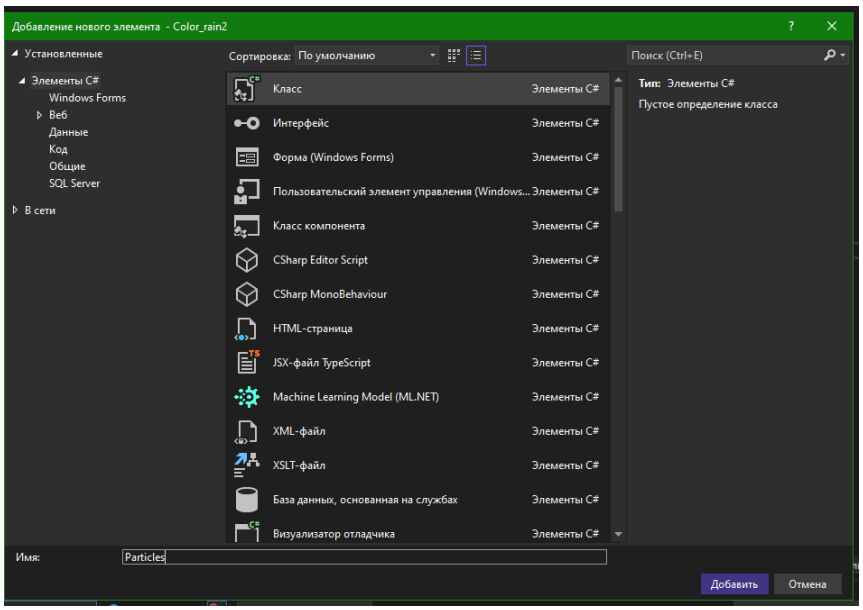

Рисунок 14. Создание класса

Создаем публичные переменные типа «int», которые будут хранить параметры капель дождя: ширину, высоту, скорость, размер, а также позиции по оси «X» и «Y». Также для всех параметров дождинок добавим эффект случайности, при помощи класса «Random» (рис.15).

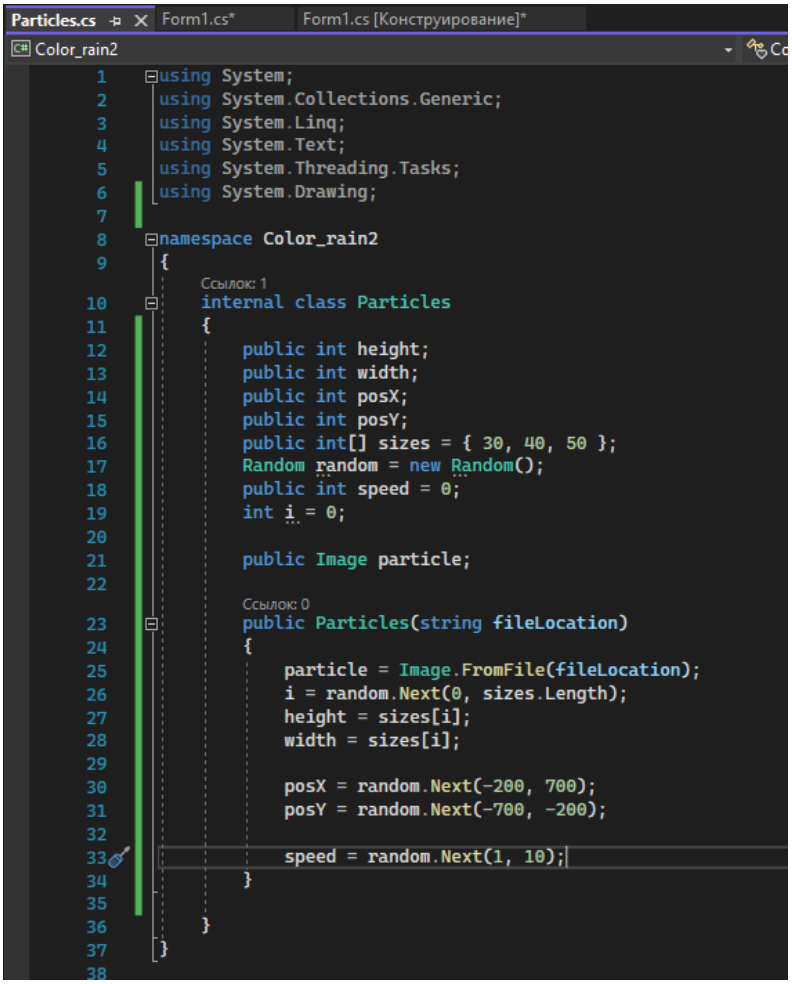

Рисунок 15. Настройка частиц (капель дождя)

Переходим к написанию логики формы, сначала сохраним файлы с каплями в список и пишем метод генерации на форме, для этого используем цикл «for» используя случайность (рис.16).

```
Particles.cs | Form1.cs | Form1.cs [Конструирование]
Color_rain2 | Color_rain2.Form1
1   ace Color_rain2
2
    лок: 3
3   lic partial class Form1 : Form
4
5       List<string>  files  =  new List<string>();
6       List <Particles> particles =  new List<Particles>();
7       Random random = new Random();
8
        Ссылок: 1
9       public Form1()
10      {
11          InitializeComponent();
12          LoadImages();
13      }
14
        Ссылок: 1
15      private void LoadImages()
16      {
17          files = Directory.GetFiles("drops", "*.png").ToList();
18
19          for (int i = 0; i < 400; i++)
20          {
21              Particles new_particle = new Particles(files[random.Next(0, files.Count)]);
22              particles.Add(new_particle);
23          }
24      }
25
26
```

Рисунок 16. Формирования списка капель и метод генерации

С помощью «foreach» в методе «ParticleTimerEvent» задаем случайные параметры каплям: скорость, позиция по осям «X» и «Y». В методе «FormPaintEvent» с помощью того же цикла рисуем капли (рис.17).

```
27      private void ParticleTimerEvent(object sender, EventArgs e)
28      {
29          foreach (Particles tempImage in particles.ToList())
30          {
31              tempImage.posY += tempImage.speed;
32              tempImage.posX += 8;
33              if (tempImage.posY > this.ClientSize.Height + 100)
34              {
35                  tempImage.speed = random.Next(5, 15);
36
37                  tempImage.posY = -100;
38
39                  tempImage.posX = random.Next(-200, 600);
40              }
41
42          }
43
44          this.Invalidate();
45      }
46
        Ссылок: 1
47      private void FormPaintEvent(object sender, PaintEventArgs e)
48      {
49          Graphics Canvas = e.Graphics;
50
51          foreach (Particles tempImage in particles.ToList())
52          {
53              Canvas.DrawImage(tempImage.particle, tempImage.posX, tempImage.posY, tempImage.width, tempImage.height);
54          }
55      }
56
57
        void Graphics.DrawImage(Image image, int x, int y, int width, int height) (+ 29 перегрузки)
```

Рисунок 17. Метод «ParticleTimerEvent» и «FormPaintEvent

Переходим к проверке работы эффект цветного дождя (рис.18-19).

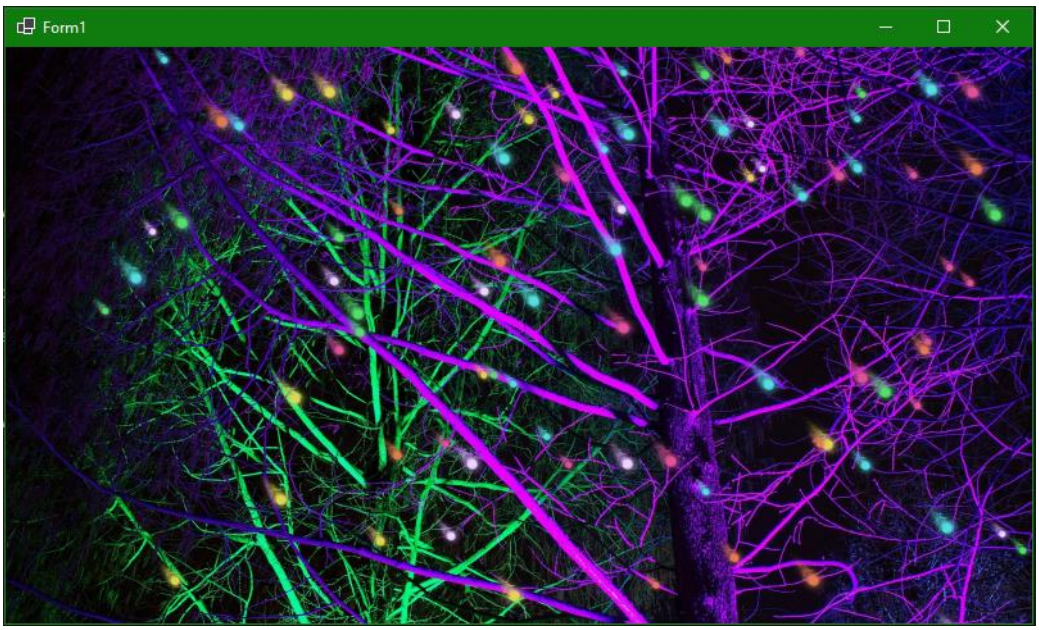

Рисунок 18. Работа эффекта цветного дождя

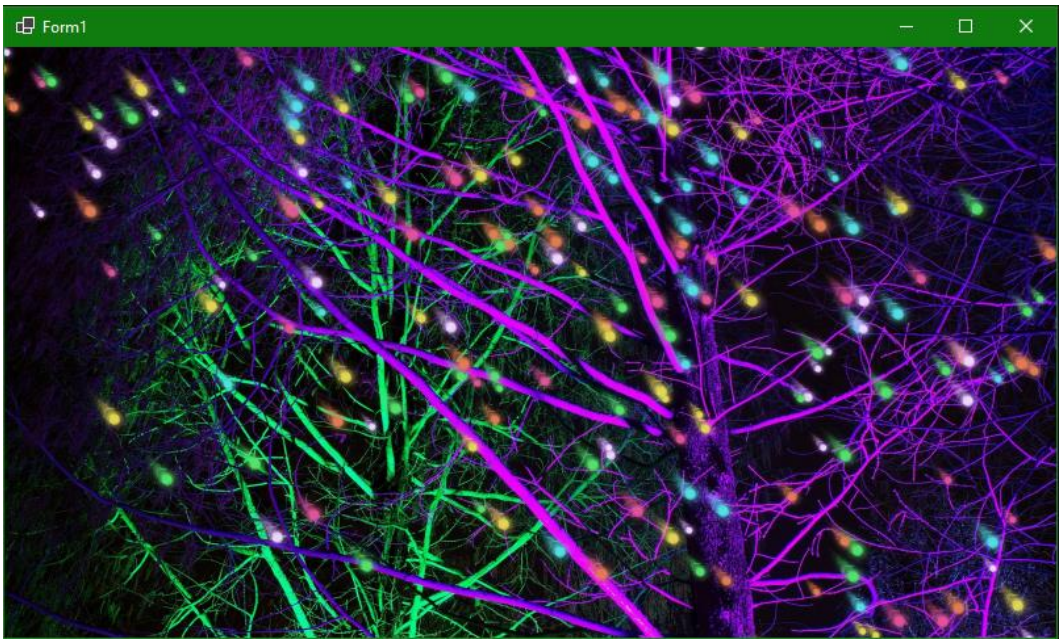

Рисунок 19. Работа эффекта цветного дождя

Были проанализированы существующие аналоги и методы разработки, а также выбрана среда разработки. Для реализации поставленной задачи отлично подошла разработка с помощью Visual Studio и языка программирования C#. Такой выбор заметно упростил разработку приложения, так как в интернете имеется достаточное количество документации. Во время создания приложения был полученный ценный опыт работы с этим средством разработки. В итоге было разработано приложения с эффектом цветного дождя.

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. Патюченко Ф.В., Слащев И.С., Клименко А.В., Трегубенко Л.А. Windows form или windows presentation foundation // Modern science. 2019. №7-2. С. 318-320.
4. Хасаева Д.З., Демин А.Ю. Разработка графической библиотеки для визуализации объектов робототехники на основе технологии Windows presentation foundation // XXVII Международная инновационно-ориентированная конференция молодых ученых и студентов. 2015. С. 536-539.
5. WFP URL: https://docs.microsoft.com/ru-ru/visualstudio/designers/getting-started-with-wpf?view=vs-2022