Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |

наименование института

|  |
| --- |
| Допускаю к защите  Руководитель:  Бахвалова З.А. |
| И.О. Фамилия |

Разработка системы управления частицами на языке C#

наименование темы

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| Технологии программирования |

|  |
| --- |
| 1.021.00.00 ПЗ |
| обозначение документа |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | ИСТб-20-3 |  |  |  | Пипия К.Т. |
|  | шифр группы |  | подпись |  | Фамилия И.О |
| Нормоконтроль |  |  |  |  | Бахвалова З.А. |
|  |  |  | подпись |  | Фамилия И.О |

|  |  |
| --- | --- |
| Курсовая работа защищена с оценкой |  |

Иркутск 2022 г.

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

|  |  |  |
| --- | --- | --- |
| По курсу | Технологии программирования | |
| Студенту | Пипия Кобе Тенгизиевичу | |
|  | (фамилия, инициалы) | |
| Тема работы: | Разработка системы управления частицами на языке C# | | |
| Исходные данные: | | Вариант 9 | |
| Рекомендуемая литература: | | | |
| 1. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 c  2. Система частиц, часть 1. [Электронный ресурс] // aqua.tealeaf.su : чаинка, 2021. URL: http://aqua.tealeaf.su/particle-system.html (дата обращения: 01.12.2021). | | | |
|  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Графическая часть на | 23 | | листах. | |
| Дата выдачи задания |  | 23 / 11 / 2021 г. | | |
| Задание получил студент | | | |  | |  | Пипия К.Т. |
|  | | | | подпись | |  | Фамилия И.О. |

|  |  |
| --- | --- |
| Дата представления работы руководителю | 27 / 12 / 2021 г. |
| Руководитель курсовой работы |  |  | Бахвалова З.А. |
|  | подпись | Фамилия И.О. |

# Содержание

[1. Введение 4](#_Toc92671442)

[2. Индивидуальный вариант 5](#_Toc92671443)

[3. Внешний вид главного окна 6](#_Toc92671444)

[4. Код класса Emitter 7](#_Toc92671445)

[5. Код класса IImpactPoint 10](#_Toc92671446)

[6. Код класса Particle 11](#_Toc92671447)

[7. Код класса Form1 11](#_Toc92671448)

[8. Код класса Program 14](#_Toc92671449)

[9. Описание работы интерфейса 14](#_Toc92671450)

[10. Заключение 16](#_Toc92671451)

[11. Список использованных литературных источнинков 17](#_Toc92671452)

# Введение

Для данной работы необходимо разработать графический интерфейс для управления системами частицами.

Система частиц используется в компьютерной графике для изображения нечетких объектов.

В системе частиц содержится определённое количество частиц, каждая из которых имеет свои свойства

При выполнении программы частицы подвергаются определённым изменениям в их состоянии. Для этого производятся расчёты.

Частицы генерируются эмиттером. В своё время эмиттер может быть представлен в виде различных фигур или точки. Это необходимо для моделирования различных эффектов.

Для выполнения работы используется язык C#. Он содержит весь необходимый набор функций необходимый для выполнения поставленной задачи.

# Индивидуальный вариант

Реализовать точки перекрашивания частиц, попадая в радиус действия которой частицы меняют свой цвет на цвет указанный у точки.

* Реализовать раскрашивание, например, снега падающего с неба
* Добавить trackbar, с помощью которого можно перемещать точки перекрашивания

# Внешний вид главного окна

Главное окно содержит в себе всё необходимое для корректной работы программы. А именно это область для отображения отрисовки частиц и область для размещения необходимых для управления деталями отображения элементами управления.

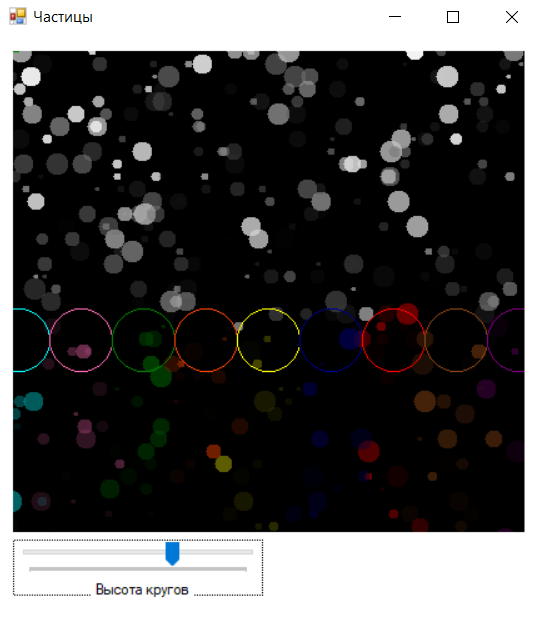


Рисунок 2.1

# Код класса Emitter

Данный класс необходим для установления значений для всех производимых им частиц.

public class Emitter

{

public int X; // координата X центра эмиттера

public int Y; // соответствующая координата Y

public int Direction = 0; // вектор направления в градусах

public int Spreading = 360; // разброс частиц относительно Direction

public int SpeedMin = 1; // начальная минимальная скорость движения частицы

public int SpeedMax = 10; // начальная максимальная скорость движения частицы

public int RadiusMin = 2; // минимальный радиус частицы

public int RadiusMax = 10; // максимальный радиус частицы

public int LifeMin = 20; // минимальное время жизни частицы

public int LifeMax = 100; // максимальное время жизни частицы

public int ParticlesCount = 500;

public Color ColorFrom = Color.White; // начальный цвет частицы

public Color ColorTo = Color.FromArgb(0, Color.Black); // конечный цвет частиц

List<ParticleColorful> particles = new List<ParticleColorful>();

public int MousePositionX;

public int MousePositionY;

public float GravitationX = 0;

public float GravitationY = 1;

public int ParticlesPerTick = 1;

public List<IImpactPoint> impactPoints = new List<IImpactPoint>();

public virtual ParticleColorful CreateParticle()

{

var particle = new ParticleColorful();

particle.FromColor = ColorFrom;

particle.ToColor = ColorTo;

return particle;

}

public virtual void ResetParticle(ParticleColorful particle)

{

particle.FromColor = Color.White;

//particle = new ParticleColorful();

particle.Life = Particle.rand.Next(LifeMin, LifeMax);

particle.X = X;

particle.Y = Y;

var direction = Direction

+ (double)Particle.rand.Next(Spreading)

- Spreading / 2;

var speed = Particle.rand.Next(SpeedMin, SpeedMax);

particle.SpeedX = (float)(Math.Cos(direction / 180 \* Math.PI) \* speed);

particle.SpeedY = -(float)(Math.Sin(direction / 180 \* Math.PI) \* speed);

particle.Radius = Particle.rand.Next(RadiusMin, RadiusMax);

}

public void UpdateIm()

{

foreach (var particle in particles)

{

foreach (var point in impactPoints)

{

point.ImpactParticle(particle);

}

}

}

public void UpdateState()

{

int particlesToCreate = ParticlesPerTick;

foreach (var particle in particles)

{

if (particle.Life <= 0)

{

if (particlesToCreate > 0)

{

particlesToCreate -= 1;

ResetParticle(particle);

}

}

else

{

particle.Life -= 1;

foreach (var point in impactPoints)

{

point.ImpactParticle(particle);

}

particle.SpeedX += GravitationX;

particle.SpeedY += GravitationY;

particle.X += particle.SpeedX;

particle.Y += particle.SpeedY;

}

}

while (particlesToCreate >= 1)

{

particlesToCreate -= 1;

var particle = CreateParticle();

ResetParticle(particle);

particles.Add(particle);

}

}

public void Render(Graphics g)

{

// утащили сюда отрисовку частиц

foreach (var particle in particles)

{

particle.Draw(g);

}

foreach (var point in impactPoints)

{

point.Render(g);

}

}

}

public class TopEmitter : Emitter

{

public int Width; // длина экрана

public override void ResetParticle(ParticleColorful particle)

{

base.ResetParticle(particle); // вызываем базовый сброс частицы

// параметры движения

particle.X = Particle.rand.Next(Width); // позиция X - произвольная точка от 0 до Width

particle.Y = 0; // ноль - это верх экрана

particle.SpeedY = 10;

particle.SpeedX = Particle.rand.Next(-2, 2); // разброс влево и вправо

}

}

# Код класса IImpactPoint

Этот класс необходим для создания точек, которые влияют на поведение состояния частиц.

public abstract class IImpactPoint

{

public float X;

public float Y;

public abstract void ImpactParticle(ParticleColorful particle);

public virtual void Render(Graphics g)

{

g.FillEllipse(

new SolidBrush(Color.Red),

X - 5,

Y - 5,

10,

10

);

}

}

public class ColorPoint : IImpactPoint

{

public Color color;

public override void Render(Graphics g)

{

// буду рисовать окружность с диаметром равным Power

g.DrawEllipse(

new Pen(color), // цвет точки(окружности)

X - 25,

Y - 25,

50,

50

);

}

public override void ImpactParticle(ParticleColorful particle)

{

float gX = X - particle.X; // расстояние между точкой и частицей по X

float gY = Y - particle.Y; // по Y

double r = Math.Sqrt(gX \* gX + gY \* gY); // считаем расстояние от центра точки до центра частицы

// если расстояние между их центрами не больше суммы их радиусов

if (r < particle.Radius + 25)

{

// меняем цвет частицы на цвет точки

particle.FromColor = color;

}

}

}

# Код класса Particle

Данный класс реализует генерацию каждой частицы и позволяет задавать для них свойства отображения и поведения на экране.

public class Particle

{

public int Radius; // радиус частицы

public float X; // X координата положения частицы в пространстве

public float Y; // Y координата положения частицы в пространстве

public float SpeedX; // скорость перемещения по оси X

public float SpeedY; // скорость перемещения по оси Y

public float Life;

// добавили генератор случайных чисел

public static Random rand = new Random();

// конструктор по умолчанию будет создавать рандомную частицу

public Particle()

{

// генерируем произвольное направление и скорость

var direction = (double)rand.Next(360);

var speed = 1 + rand.Next(10);

// рассчитываем вектор скорости

SpeedX = (float)(Math.Cos(direction / 180 \* Math.PI) \* speed);

SpeedY = -(float)(Math.Sin(direction / 180 \* Math.PI) \* speed);

Radius = 2 + rand.Next(10);

Life = 20 + rand.Next(100);

}

public virtual void Draw(Graphics g)

{

float k = Math.Min(1f, Life / 100); //коэффициент прозрачности по шкале от 0 до 1.0

int alpha = (int)(k \* 255);

// создали кисть для рисования

var color = Color.FromArgb(alpha, Color.Black);

var b = new SolidBrush(color);

// нарисовали залитый кружок радиусом Radius с центром в X, Y

g.FillEllipse(b, X - Radius, Y - Radius, Radius \* 2, Radius \* 2);

g.DrawLine(new Pen(Color.Black, 2), 0, 0, 5, 0);

b.Dispose();

}

}

# Код класса Form1

Класс, который позволяет настроить отображение информации на главном окне.

public partial class Form1 : Form

{

List<Emitter> emitters = new List<Emitter>();

Emitter emitter;

public Form1()

{

InitializeComponent();

picDisplay.Image = new Bitmap(picDisplay.Width, picDisplay.Height);

emitter = new TopEmitter

{

Direction = 0,

Spreading = 10,

SpeedMin = 10,

SpeedMax = 10,

ColorFrom = Color.White,

ColorTo = Color.FromArgb(0, Color.Black),

ParticlesPerTick = 10,

Width = picDisplay.Width,

GravitationY = 0f

};

// ниже добавляю цветные круги

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 - 200),

Y = picDisplay.Height / 2 - 25,

color = Color.Cyan

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 - 150),

Y = picDisplay.Height / 2 - 20,

color = Color.HotPink

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 - 100),

Y = picDisplay.Height / 2 - 15,

color = Color.Green

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 - 50),

Y = picDisplay.Height / 2 - 10,

color = Color.OrangeRed

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2),

Y = picDisplay.Height / 2 - 5,

color = Color.Yellow

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 + 50),

Y = picDisplay.Height / 2 - 10,

color = Color.DarkBlue

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 + 100),

Y = picDisplay.Height / 2 - 15,

color = Color.Red

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 + 150),

Y = picDisplay.Height / 2 - 20,

color = Color.SaddleBrown

});

emitter.impactPoints.Add(new ColorPoint

{

X = (float)(picDisplay.Width / 2 + 200),

Y = picDisplay.Height / 2 - 25,

color = Color.Purple

});

}

private void timer1\_Tick(object sender, EventArgs e)

{

emitter.UpdateState(); // обновляем эмиттер

using (var g = Graphics.FromImage(picDisplay.Image))

{

g.Clear(Color.Black);

emitter.Render(g); // рендерим через эмиттер

}

picDisplay.Invalidate();

}

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

{

foreach (var p in emitter.impactPoints)

{

// если эмиттер цветной круг

if (p is ColorPoint)

{

p.Y += trackBar2.Value - p.Y; // меняем положение

}

}

}

}

# Код класса Program

Класс с которого начинается работа программы, так как он содержит точку входа в программу.

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

# Описание работы интерфейса

Меняя значение trackback «Высотакругов**»** мы меняем высоту цветных кругов на форме:

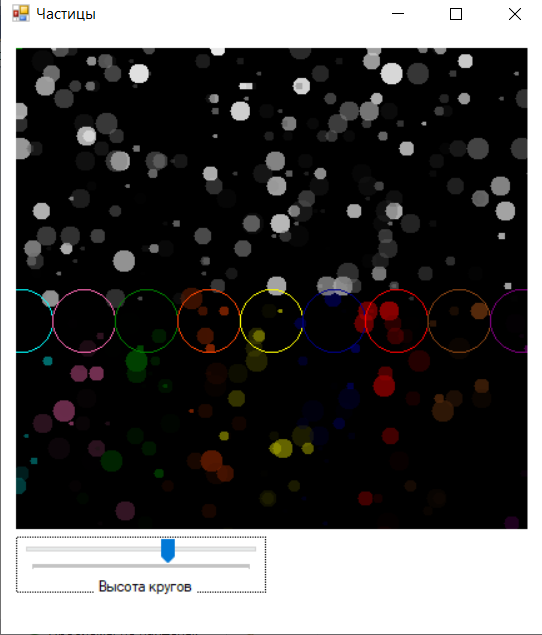


Рисунок 1 - Интерфейс программы

Рисунок 7.1

Частицы попадающие в область действия специальной точки меняют свой цвет в соответствии с цветом этой точки:

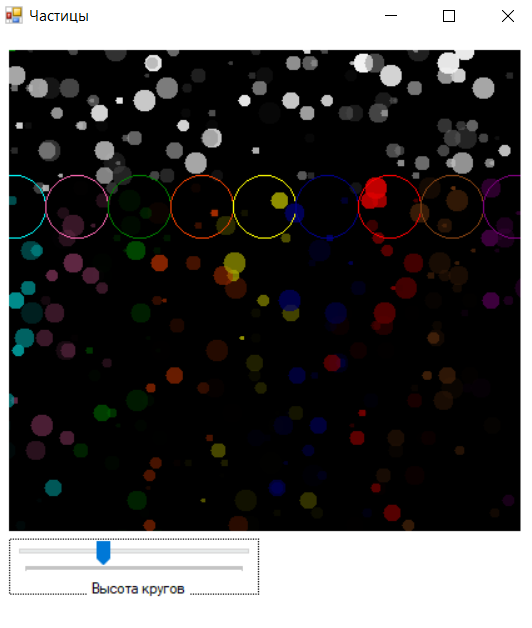


Рисунок 2 - Интерфейс программы с демонстрацией возможностей

Рисунок 7.2

# Заключение

В рамках курсовой работы было разработано приложение, позволяющее управлять системой частиц. В приложении реализованы следующие возможности:

* Ползунок «Высота кругов», для управления положением специальных точек;
* Специальная точка «Гравитон[Цветовой круг]» которая меняет цвет частицы, которая в неё попала.

Приложение было всесторонне протестировано и отлажено. Все поставленные в рамках курсовой работы задачи были выполнены.

В рамках выполнения данной работы был получен обширный спектр сведений об объектно-ориентированном языке программирования C#, разработки графических приложений и эмуляции сложных динамических систем.

# Список использованных литературных источнинков

1. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 c.
2. Вагнер, Билл С# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2013. - 320 c.
3. Ишкова, Э. А. Самоучитель С#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 c.
4. Биллинг В.А. Основы программирования на C#. - Т.: Интернет-университет информационных технологий, Бином, 2012. - 488 с.
5. Евдокимов П.В. C# на примерах. - М.: Наука и техника,2016. - 304 с.
6. Нейгел К., Ивьен Б., Глинн Д., Уотсон К., Скиннер М. C# 4.0 и платформа .NET 4, 2011.
7. Система частиц, часть 1. [Электронный ресурс] // aqua.tealeaf.su : чаинка, 2021. URL: http://aqua.tealeaf.su/particle-system.html (дата обращения: 01.12.2021).