

# *ЗАНЯТИЕ 3*

*ЦВЕТОВАЯ МОДЕЛЬ RGB.  
СЛУЧАЙНЫЕ ЧИСЛА.  
ЦИКЛ FOR СО СЧЕТЧИКОМ.*

# *Случайное число. Класс Random.*

Для генерации случайных чисел в программе используется класс **Random**. В таблице приведены конструкторы класса **Random**:

<i>Название</i>	<i>Описание</i>
<b>Random ( )</b>	Создает экземпляр класса, основываясь на значении текущего времени.
<b>Random (Int32)</b>	Создает экземпляр класса, основываясь на начальном значении, заданном в качестве входного параметра

# *Некоторые методы класса Random*

Метод	Описание
<b>Next ()</b>	Возвращает случайное не отрицательное целое число от 0 до int.
<b>MaxValue.</b>	Инициализация датчика случайных чисел основывается на системном времени.
<b>Next (Int32)</b>	Возвращает случайное неотрицательное целое число, не превышающее значение аргумента.
<b>Next (Int32, Int32)</b>	Возвращает целое число, равномерно распределенное в интервале от 1-го параметра до 2-го параметра.
<b>NextDouble()</b>	Возвращает дробное число от 0.0 до 1.0

# *Пример «Случайный кружок».*

Программа, рисует круг случайным радиусом из интервала (20; 100), в случайном месте и любым цветом.

## Пример «Случайный кружок».

```
namespace Случайный_кружок
{
    public partial class Form1 : Form
    {
        int r, x, y;
        int fw, fh; // Ширина и высота формы
        Random rnd = new Random();
        //создается объект класса Random
        Pen pen1 = new Pen(Color.Blue);
        Graphics gr;

        public void krug(int x, int y, int r)
        {
            pen1.Color = Color.Blue;
            gr.DrawEllipse(pen1, x, y, 2 * r, 2 * r);
        }
    }
}
```

```
private void button1_Click(object sender,
                               EventArgs e)
{
    r = rnd.Next(20, 100);
    x = rnd.Next(fw - 2 * r); // Следим,
    //чтобы кружок не выходил за форму
    y = rnd.Next(fh - 2 * r);
    krug(x, y, r);
}

public Form1()
{
    InitializeComponent();
    gr = this.CreateGraphics();
    this.BackColor = Color.Aquamarine;
    fw = this.Width; //Текущая ширина формы
    fh = this.Height; //Текущая высота формы
}
}
```

# Цикл со счетчиком *for*.

Цикл *for* используется, как правило, если известно число повторов цикла:

```
for (<начальные значения>; <условия работы цикла>; <инкремент>)  
{ <тело цикла> }
```

Цикл *for* выполняет <тело цикла> до тех пор, пока условие работы цикла не станет равным *false*. В случае зависания программы (зацикливания) нажмите **Shift+F5**.

Пример использования оператора *for* :

Ряд кружков. Нарисуйте ряд кружочков (например 10 штук) с радиусом 40:

```
private void Form1_Paint(object sender, PaintEventArgs e)  
{ SolidBrush myBrush = new SolidBrush(Color.Red);  
  myBrush.Color = Color.Red;  
  y = 50;  
  x = 0;  
  for (int i=1; i<=10; i++)  
    { g.FillEllipse(myBrushred, x, y, 2*r, 2*r);  
      x += 2 * r;  
    }  
}
```

# Использование цвета. Цветовая модель RGB.

Существует набор системных цветов, которые берутся из структуры *Color*.

## Примеры:

```
Color col;
```

```
col = Color.Red; // красный
```

```
col = Color.Aquamarine; // аквамарин
```

```
col = Color.LightGoldenrodYellow; // светлый золотисто-желтый
```

```
col = Color.Tomato; // томатный
```

Иногда в процессе работы программы форма должна перекрашиваться.

Перекрашивание формы программно, например, в зеленый цвет:

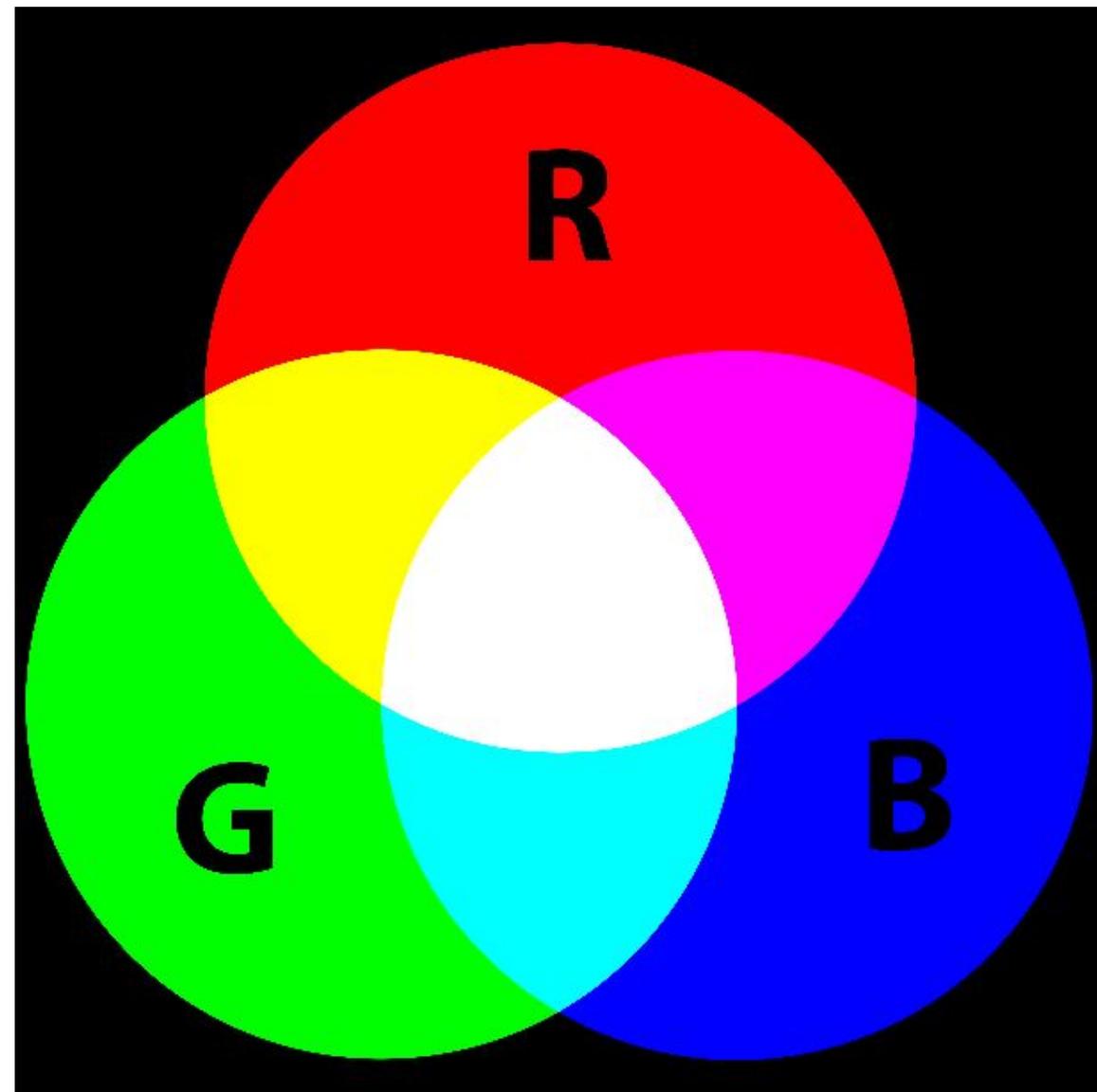
```
public Form1()
{
    InitializeComponent();
    this.BackColor = Color.Green;
}
```

Если понадобится изменить цвет, например, кнопки Button1 на тёмно-бордовый, код будет таким:

```
button1.BackColor = Color.Maroon;
```

# Цветовая модель RGB.

**RGB** — это аддитивная цветовая модель, которая синтезирует цвета, используя смешивание трёх основных цветов (Красного — **Red**, Зеленого — **Green**, Синего — **Blue**) с чёрным, вследствие чего получаются новые цвета и оттенки. Зависит получаемый цвет от интенсивности этих трёх основных цветов. Если смешать Красный, Зеленый и Синий в максимальной насыщенности, получится белый цвет. Если не смешивать их, то остаётся чёрный.



# Создание цвета. Метод *Color.Argb*

Можно создавать свои цвета, задавая интенсивность каждого цвета с помощью метода *Color.FromArgb*. Первый аргумент этого метода задает интенсивность красного, второй – интенсивность зеленого, и последний – интенсивность голубого цвета. Интенсивность в числовой форме для удобства применения задается числами от 0 (минимальная интенсивность) до 255 (максимальная интенсивность). Чтобы закрасить фон формы в чёрный цвет, используя данный метод, надо написать вот такую строку:

```
public Form1()
{
    InitializeComponent();
    this.BackColor = Color.FromArgb(0, 0, 0);
}
```

## Примеры

```
Color col;
```

```
col = Color.FromArgb(23, 56, 78);
```

Или

```
col = Color.FromRgb(0, 0, 0); // черный
```

```
col = Color.FromRgb(0, 255, 0); // зеленый
```

```
col = Color.FromRgb(255, 0, 0); // красный
```

```
col = Color.FromRgb(0, 0, 255); // синий
```

```
col=Color.FromRgb(255, 255, 0); // желтый
```

```
col=Color.FromRgb(255, 0, 255);
```

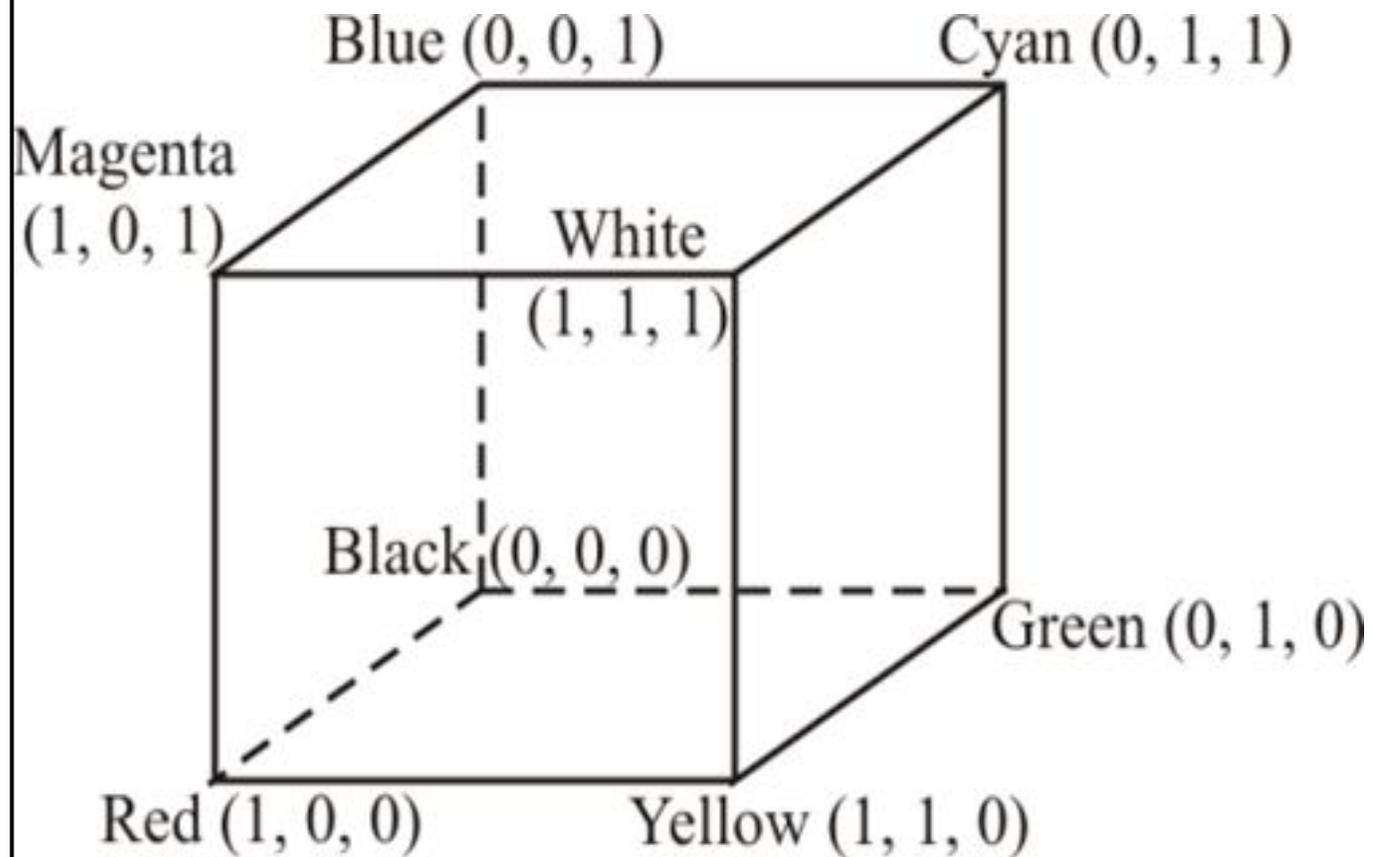
```
                // фиолетовый (magenta)
```

```
col = Color.FromRgb(0, 255, 255); // циан
```

```
col=Color.FromRgb(255, 255, 255); // белый
```

# *Модель RGB в виде трехмерного куба.*

Цветовым пространством модели RGB является единичный куб в трехмерной системе координат. На рисунке изображен куб и трехмерная система координат. На главной диагонали куба получаются серые цвета от черного до белого. Главная диагональ идет из точки с координатами  $(0,0,0)$  в точку с координатами  $(1, 1, 1)$ . Начало координат на рисунке  $(0, 0, 0)$  – черный цвет, точка с координатами  $(1, 1, 1)$  имеет белый цвет. Главная диагональ куба характеризуется равным вкладом трех базовых цветов:  $r$  – красный,  $g$  – зеленый и  $b$  – синий.



# *Задание цвета случайным образом*

**byte r, g, b;**

**...**

**Random rnd = new Random();**

**Brush br1= new Brush(Color.Black); // При создании кисти**

**// используем любой цвет**

**r= rnd.Next(15;256); // Отступили от нуля на 15,**

**// чтобы избежать темных оттенков**

**g= rnd.Next(15;256);**

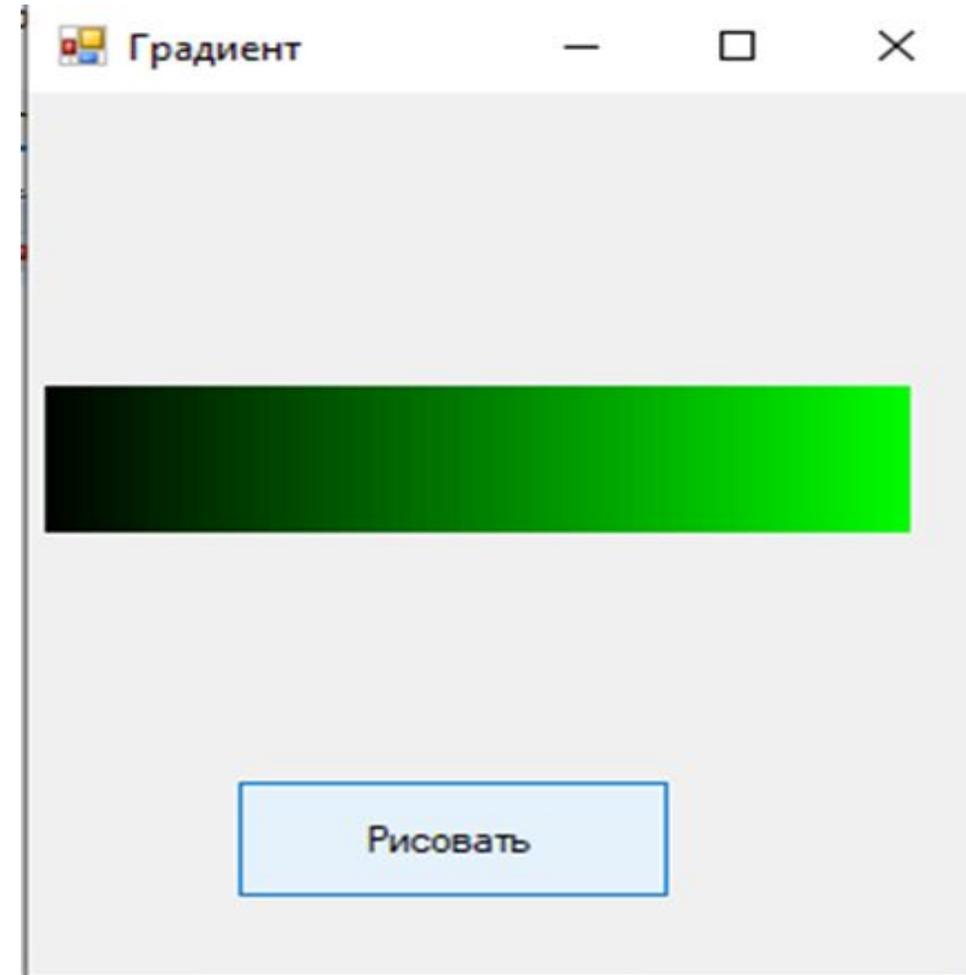
**b= rnd.Next(15;256);**

**br1.Color= Color.FromArgb(r,g,b);**

**...**

**Пример.** Нарисуем длинную полосу, покрашенную градиентным образом от темно-зеленого до светло-зеленого.

```
namespace Градиент1
{
    public partial class Form1 : Form
    {
        int y, a, r, g, b;
        int x = 0;
        SolidColorBrush br1 = new SolidColorBrush(Color.Black);
        private void button1_Click(object sender,
                                    EventArgs e)
        {
            drawpole();
        }
    }
}
```



# Пример. Продолжение.

```
int n = 50, h = 50; // 50 полосок
Graphics gr;
public void drawpole()
{
    a = 5; // Шаг или ширина одного
           // прямоугольника
    r = 0; // Оттенки красного цвета
    b = 0; // Оттенки синего цвета
    y = 100;
```

```
for (int i=1; i<=n; i++)
    { g = i*a; // Оттенки зеленого цвета
      x +=a;
      br1.Color = Color.FromArgb(r, g, b);
      gr.FillRectangle(br1, x, y, a, h);
    }
}
public Form1()
{ InitializeComponent();
  gr = this.CreateGraphics();
}
}
}
```

# *Прозрачность.*

В одном из так называемых перегруженных вариантов метода **FromArgb** (то есть с четырьмя входными параметрами) первый параметр задает прозрачность объекта. Прозрачность задается от 0 до 255. Совсем прозрачный соответствует числу 255. Например:

```
Color col;
```

```
col = Color.FromArgb(127, 23, 56, 78);
```

В этом примере создается голубовато-серый цвет с примерно 50% прозрачностью.

Можно создать также цвет с прозрачностью на основе цвета, имеющегося в структуре **Color**, например:

```
Color col;
```

```
col = Color.FromArgb(128, Color.Tomato);
```

# ***Задания.***

**Задание 2. «Градиент с прозрачностью».** Измените пример 2. Нарисуйте ниже еще одну градиентную полосу, задавая в ней прозрачность.

**Задание 3. «Ряд разноцветных кружков».** Измените задачу «Ряд кружочков» так, чтобы все кружочки имели случайный цвет.

**Задание 4. «Мыльные пузыри».** Напишите программу, рисующую случайное количество окружностей (не больше 20), в случайном месте, случайным радиусом (не больше 100) и случайным цветом (r,g,b).

**Задание 5. «Воздушные шары».** Измените проект «Мыльные пузыри» так, чтобы окружности стали закрашенными, и шары не должны выходить за границы формы.

**Задание 6. «Поле».** Нарисуйте сетку поля для различных игр, состоящее из незакрашенных клеточек размерностью 5 x 5 (как шахматная доска, но все клеточки одинаковые).

**Задание 7. «Шахматная доска».** Нарисуйте шахматную доску.