Главное меню. Динамическое создание компонент. Компонент PictureBox.

Главное меню. Элемент MenuStrip.

Для создания меню в Windows Forms применяется элемент **MenuStrip**. После создания проекта следует разместить его на форме.

Для создания меню в визуальном режиме предназначен *конструктор меню*, который активизируется при выделении компонента **menuStrip1.**

В версии 2019 поле ввода с текстом **Type Here** появляется на месте будущего пункта меню. В него следует ввести имя создаваемого пункта меню.

После ввода имени и нажатия клавиши **<Enter>** в меню будет создан новый пункт, а рядом и под ним будут выведены новые поля ввода с текстом **TypeHere**, позволяющие создать очередные пункты меню первого или второго уровня.

При выделении созданного пункта меню в окне **Properties** отображаются свойства данного пункта.

Главное меню. Элемент MenuStrip.

Имя пункта меню, в отличие от имен других компонентов, размещаемых в форме, получается не добавлением порядкового номера к имени типа (например, label1), а приписыванием имени команды меню слева к имени типа, например, **fileToolStripMenuItem** (это правило действует даже для пунктов меню с русскими названиями, например, **файлToolStripMenuItem).**

В результате получаются очень длинные имена, поэтому сразу после создания пункта меню желательно изменить имя пункта, указав новое значение его свойства **Name**.

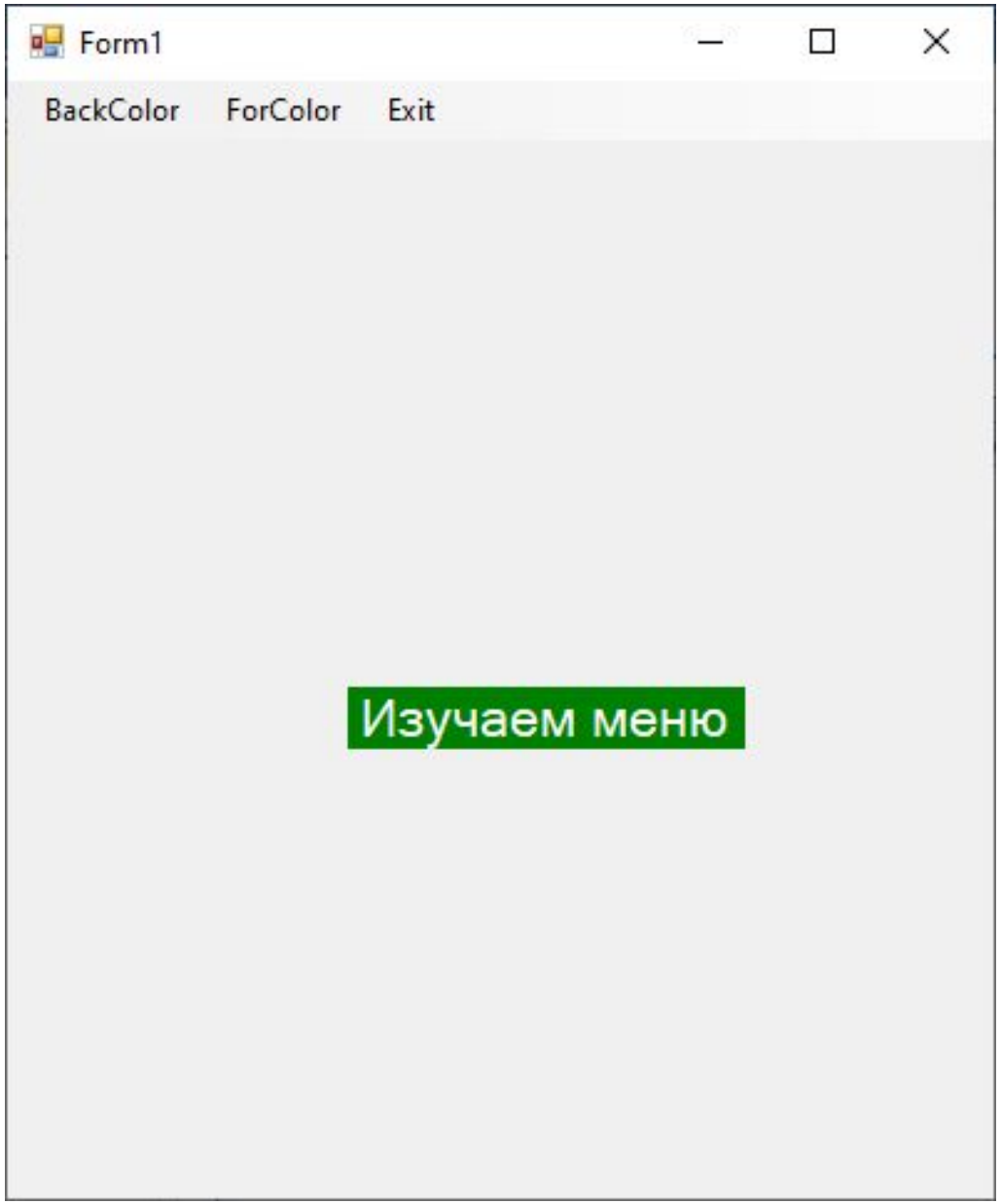
Заметим, что свойство Name располагается в окне **Properties** в *начале списка свойств*; причем подпись к нему заключается в скобки: **(Name).**

Главное меню. Элемент MenuStrip.

На событие Click элемента меню пишется программный код, который будет выполняться при нажатии на пункт меню.

На рисунке представлена форма с пунктами меню, переключающими цвета Label.

Программный код приведен в примере на следующем слайде.



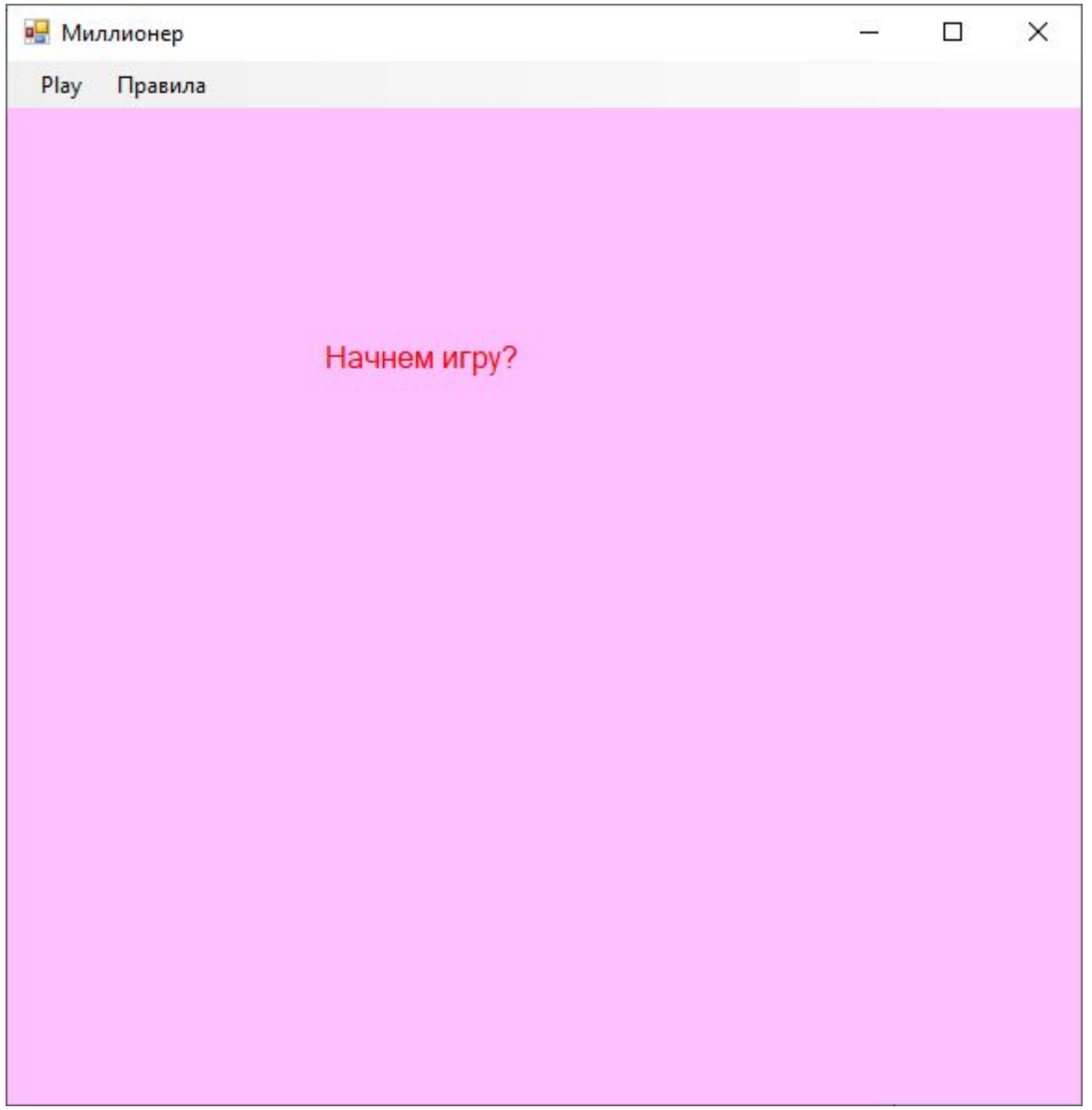
Пример программного кода главного меню.

```
//Обработчики пунктов меню
        private void green_Click(object sender, EventArgs e)
           {  label1.BackColor = Color.Green;   }
         private void blue_Click(object sender, EventArgs e)
           {  label1.BackColor = Color.Blue;    }
        private void white_Click(object sender, EventArgs e)
           {  label1.ForeColor = Color.White;   }
        private void yellow_Click(object sender, EventArgs e)
           {  label1.ForeColor = Color.Yellow;   }
        private void magenta_Click(object sender, EventArgs e)
           { label1.ForeColor = Color.Magenta;   }
        private void exit_Click(object sender, EventArgs e)
           {   //Закрываем проект
               this.Close();
            }
```

Учебная викторина Миллионер

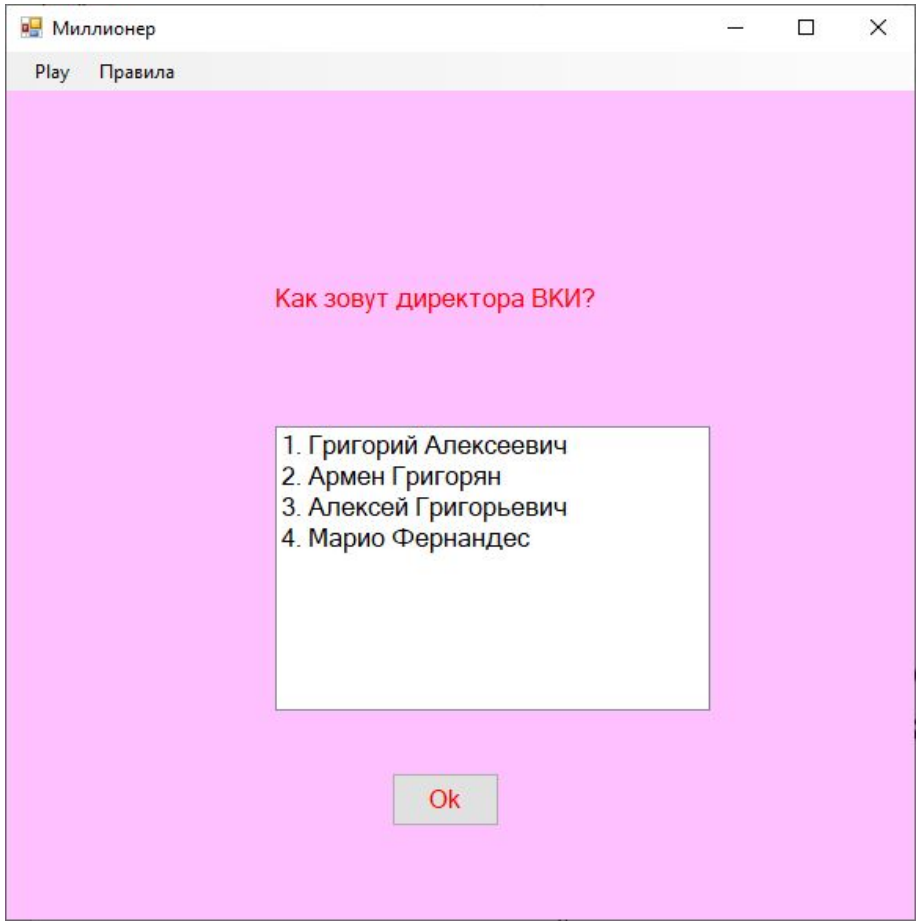
Телевизионная игра «Кто хочет стать миллионером?» хорошо известна. Наша викторина создана на ее основе, поэтому пояснять ее правила мы не будем. Это упражнение направлено на развитие у учащихся навыков работы с рядом компонентов Windows Forms.

В начале на основном окне видны только пункты меню и приглашение начать игру

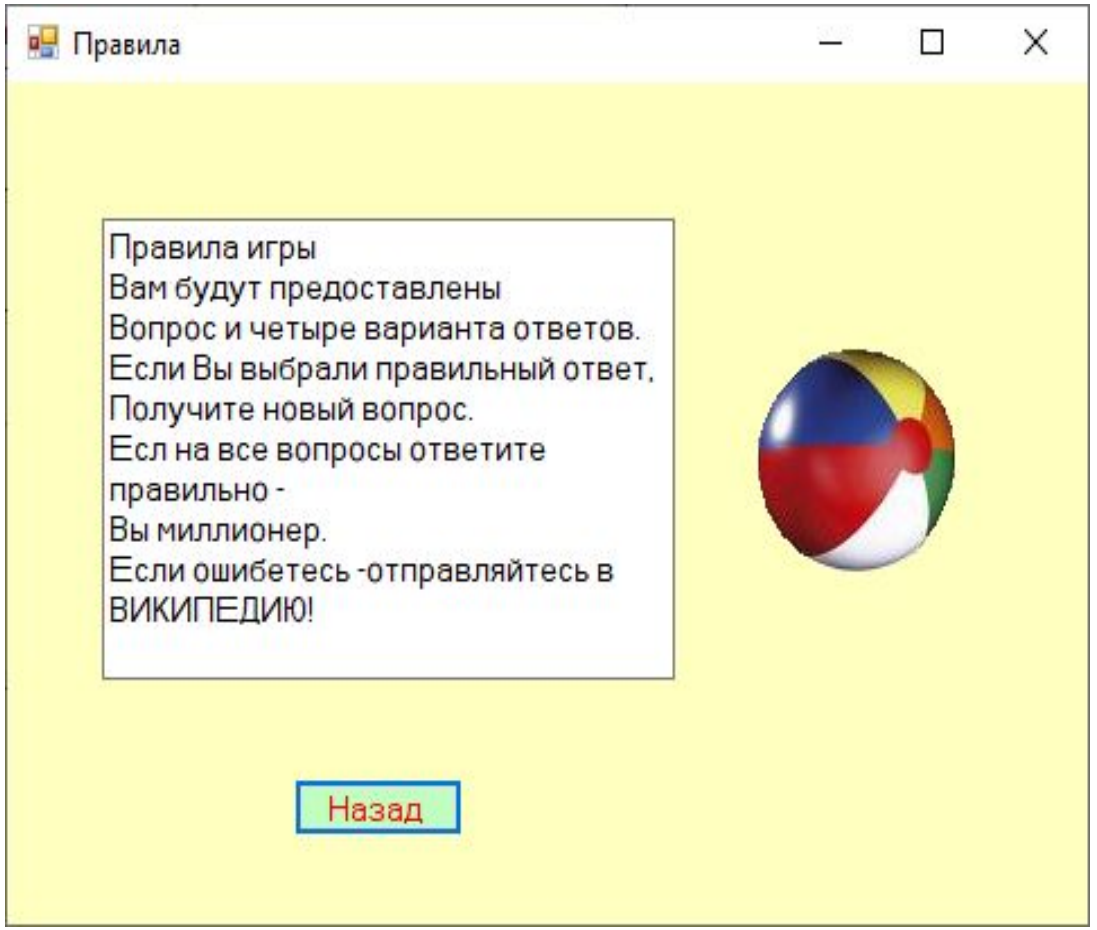


Учебная викторина Миллионер

Выбирая пункт меню «Play», мы начинаем игру. Появляется ListBox с вариантами ответа и сам вопрос в Label



Пунктом меню «Правила» мы вызываем второе окно с текстом помощи



Код викторины миллионер с комментариями.

Код первой формы:

```
public partial class Form1 : Form
  {  public Form2 frm2;
        List<string> ask = new List<string>();//Объявление списка вопросов
        int otvet;
        public Form1()
        {    InitializeComponent();   }
     private void play_Click(object sender, EventArgs e)
        {   label1.Visible = true;//Управление игрой видимо
            listBox1.Visible = true;
            button1.Visible = true;
            ask.AddRange(File.ReadAllLines(@"D:\C#\My_Text\questions.txt"));
            label1.Text = ask[0];//Вопрос
            for (int i = 1; i < 5; i++)//ответы
                listBox1.Items.Add(ask[i]);
            otvet = Convert.ToInt32(ask[5]);//Правильный ответ    }
     private void правила_Click(object sender, EventArgs e)
        {   frm2 = new Form2();//Создаем указатель на форму 2
            frm2.frm1 = this;//Передаем форме 2 указатель на форму 1
            frm2.Show();//Показываем форму 2
            this.Hide();//Прячем форму 1       }
```

Код викторины миллионер с комментариями.

```
private void button1_Click(object sender, EventArgs e)
      {  if (listBox1.SelectedIndex==otvet)
           {   MessageBox.Show("Ok");
               ask.RemoveRange(0, 6);//Удаляем предыдущий набор
               label1.Text = ask[0];//Новый ворос
               listBox1.Items.Clear();
               for (int i = 1; i < 5; i++)
                    listBox1.Items.Add(ask[i]);
               otvet = Convert.ToInt32(ask[5]);
           }
           else
           {   MessageBox.Show("Почитай Википедию");
               label1.Visible = false;//Управление невидимо
               listBox1.Visible = false;
               button1.Visible = false;
           }
       }
   }
```

Код викторины миллионер с комментариями.

Код второй формы:

```
public partial class Form2 : Form
    {         public Form1 frm1;
public Form2()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
         //frm1 = new Form1();
            this.Hide();//Прячем форму 2
            frm1.Show();//Показывем форму 1
        }
    }
```

Задание по программе Миллионер.

Собрать программу на своем компьютере. Дополнить программу условием “окончания игры” (успешным или неуспешным) и выводом результата игры на экран в другой форме. Добавить в МЕНЮ кнопку “выхода из игры”.

Создание компонент программным путём.

Мы уже обсуждали создание компонентов формы с помощью панели управления. Но зачастую есть необходимость создавать компоненты по ходу выполнения программы. Еще один случай: бывает необходимо создать массив компонентов и работать с ним с помощью цикла. Рассмотрим эти вопросы. Начнем с программного создания кнопки.

Кнопка будет создана при нажатии на заготовленную с помощью панели управления кнопку. Код этой программы представлен в примере. На рисунке представлено окно с программно-

Создание компонента Button программным способом

```
private void button1_Click(object sender, EventArgs e)
 { Button MyButton = new Button(); // Создаем кнопку с именем
                                                    //   MyButton
   MyButton.Location = new Point(65, 25);//Задаем ее положение
                                                    // на форме
   MyButton.Text = "Mb"; // Текст
   MyButton.BackColor = Color.Green; // Цвет фона
   MyButton.Size = new Size(70, 40); // Размер кнопки
   MyButton.Font = new Font("Arial", 10, FontStyle.Regular);
//Шрифт
   MyButton.ForeColor = Color.Red; // Цвет текста
   MyButton.Click+= b1_Click; //Создаем указатель на метод кнопки.
   Controls.Add(MyButton);//Добавляем кнопку в коллекцию элем-
овформы
 }
```

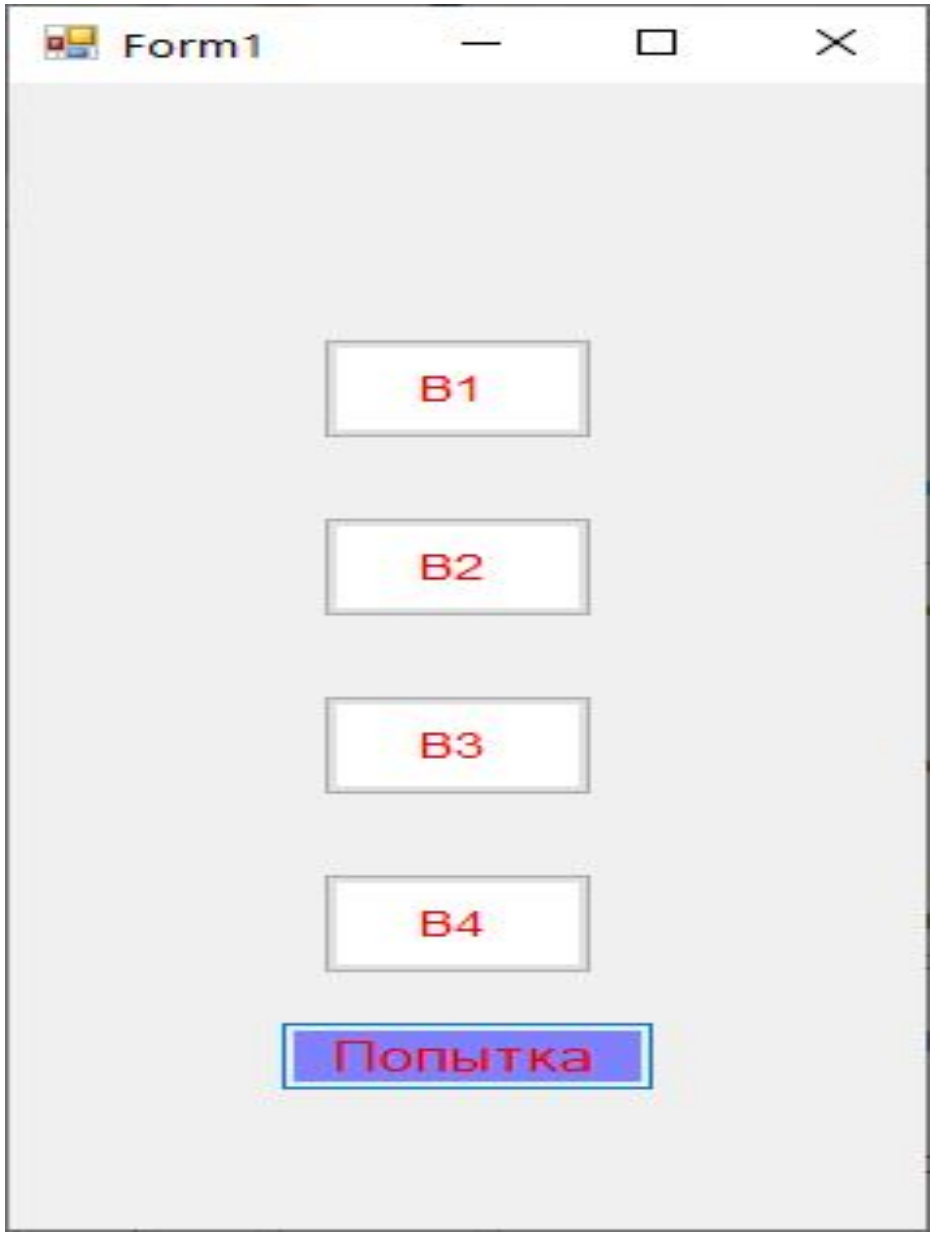
Создание Button программным способом

```
//Обработчик события кнопки MyButton
  public void b1_Click(object sender, EventArgs e)
   {
      //sender-объект, на котором произошло событие.
      //Преобразуем его в кнопку
     Button Tb = sender as Button;
      // Button Tb = (Button)sender;
      // Можно преобразовать и так
     MessageBox.Show(Tb.Text);
 }
```

Создание массива кнопок программным способом

Теперь создадим массив кнопок. Конкретно: создадим четыре кнопки, пронумеруем их и дадим каждой имя.

Программа загадывает случайное число, а играющий должен угадать, какое это число. Для этого он мышью кликает по одной из кнопок. Данный проект – простейший пример динамического создания кнопок и их методов. Форма проекта после начала игры приведена на рисунке.



Код создания массива кнопок программным способом

```
namespace Lohotron
{
    public partial class Form1 : Form
    {
        Button[] b1 = new Button[4]; //Массив кнопок
        int l;//Случайное число

     public Form1()
     {
         InitializeComponent();
     }
```

Код создания массива кнопок программным способом

```
public void MYButton()
        {  //Формируем массив кнопок
            for (int i = 0; i < 4; i++)
            {   b1[i] = new Button();
                b1[i].Left = 80;
                b1[i].Text = "B" + (i + 1).ToString();
                b1[i].Top = 100 + i * 70;
                b1[i].Size = new Size(70, 40);
                b1[i].Tag = i;
                b1[i].Click += b1_Click;
                b1[i].BackColor = Color.White;
                b1[i].ForeColor = Color.Red;
                b1[i].Font = new Font("Arial", 10, FontStyle.Regular);
                b1[i].Hide();  //Прячем кнопки массива
                Controls.Add(b1[i]);
             }
        }
```

Код создания массива кнопок программным способом

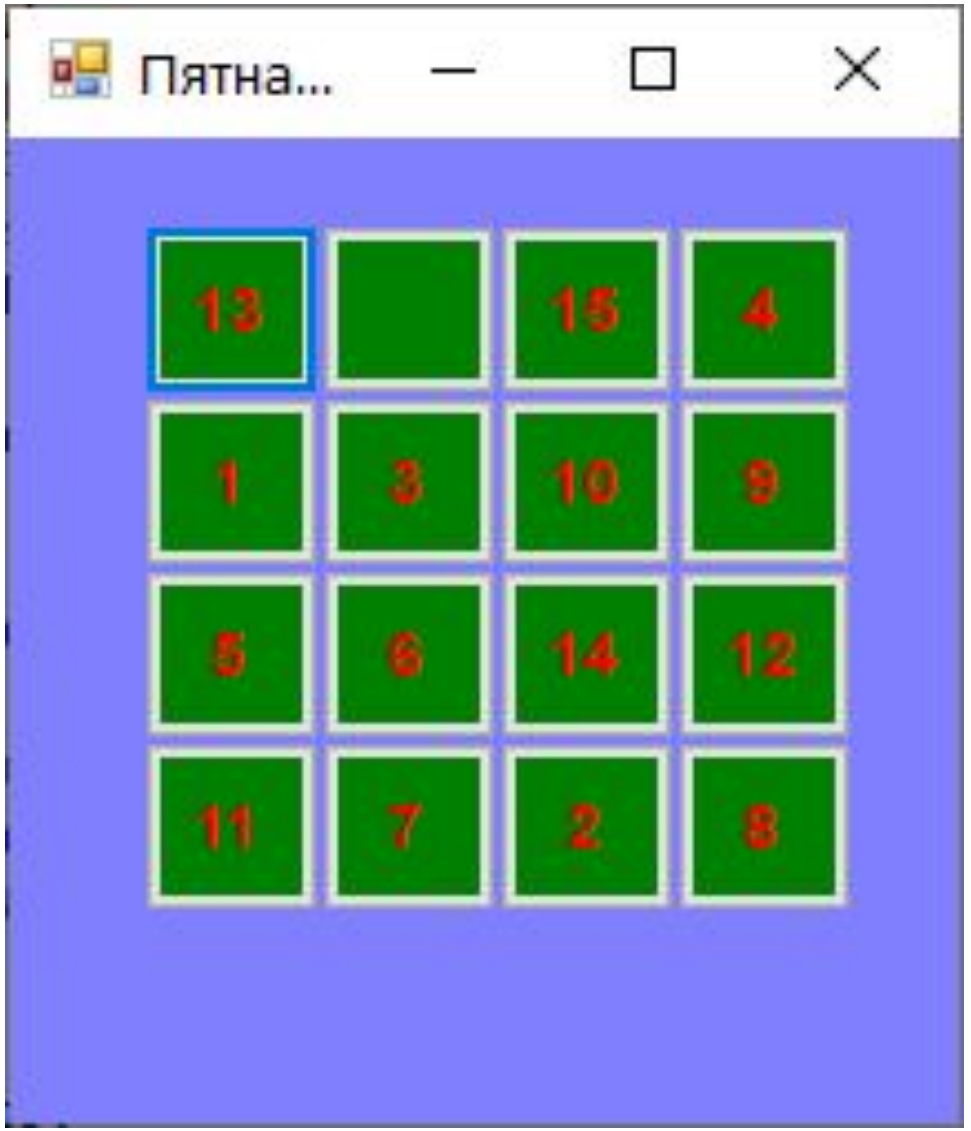
```
private void Form1_Load(object sender, EventArgs e)
        {  MYButton(); //Вызываем MYButton   }

public void b1_Click(object sender, EventArgs e)
        {   Button Tb;
            Tb = (Button)sender; //Преобразование
            if ((int)Tb.Tag == 1) //Преобразуем Tag в int
            {   MessageBox.Show("Победа");
                for (int i = 0; i < 4; i++)
                    b1[i].Hide(); //Прячем все кнопки
            }
            else  Tb.Hide(); //Прячем кнопку
        }
```

Код создания массива кнопок программным способом

```
private void button1_Click(object sender, EventArgs e)
        {
            Random rnd = new Random();
            l = rnd.Next(0, 4);//Загадали число
            for (int i = 0; i < 4; i++)
                b1[i].Show();//Показали кнопки
        }
    }
}
```

Учебная игра «Пятнадцать». Код программы.

В качестве примера динамического создания двумерного массива кнопок приведем разработку известной настольной игрs «Пятнадцать». Ее игровое поле представлено на рисунке.




```
public partial class Form1 : Form
    {   //Контрольный массив
        public int[,] a = new int[4, 4];
        int h = 40;//Размер кнопки
        int x0 = 30;//Угол игрового поля
        int y0 = 20;
        int i0, k0;
        Button[,] bt=new Button[4,4];
                              //Массив кнопок
```

Учебная игра «Пятнадцать». Код программы.

```
public Form1()
  {
      InitializeComponent();
   }
  //Перестановка
  public void swap(ref int a, ref int b)
  {
      int r;
      r = b;
      b = a;
      a = r;
   }
```

```
public void zapolnenie()
  {
     int i, j, k, m;
     bool zap;
     int n = 4;
     Random rnd = new Random();
     for (i = 0; i < 4; i++)
       for (j = 0; j < 4; j++)
            a[i, j] = 0;
```

Учебная игра «Пятнадцать». Код программы.

```
for (i = 0; i < 15; i++)
    {
        zap = true;
        while (zap)
        {
            k = rnd.Next(0, 4);
            m = rnd.Next(0, 4);
            if (a[k, m] == 0)
            {
                a[k, m] = i + 1;
                zap = false;
             }
          }
      }
  }
```

```
private void Form1_Load(objectsender,EventArgs
e)
  {  zapolnenie();
     int i,j;
     for (i=0;i<4;i++)
         for (j=0; j<4; j++)
         {   bt[i,j] = new Button();
             bt[i,j].Left = x0+j*h;
             bt[i,j].Top = y0+i*h;
             bt[i, j].Width = h;
             bt[i, j].Height = h;
             bt[i,j].Font= new Font("Arial", 10,
                                    FontStyle.Bold);
             bt[i, j].BackColor = Color.Green;
             bt[i, j].ForeColor = Color.Red;
```

Учебная игра «Пятнадцать». Код программы.

```
    if (a[i, j] != 0)
          bt[i, j].Text = a[i, j].ToString();
     else
       {  //Пустое поле
          i0 = i;
          k0 = j;
       }
       bt[i,j].Click += bt_Click; //Создаем указатель на метод кнопки.
       Controls.Add(bt[i,j]); //Добавляем кнопку в коллекцию
                                                  //элементов формы
    }
}
```

Учебная игра «Пятнадцать». Код программы.

```
public void bt_Click(object sender, EventArgs e)
        {   int i,k;
            Button Tb;
            Tb = (Button)sender;
            k = (Tb.Left - x0)/h;
            i = (Tb.Top - y0)/ h;
         //Проверка наличия рядом пустого поля
            if (((Math.Abs(i-i0)==1)&&(k==k0))||((Math.Abs(k - k0)==1)&&(i==i0)))
                {  swap(ref a[i,k],ref a[i0, k0]);
                    bt[i, k].Text = "";
                    bt[i0, k0].Text = a[i0, k0].ToString();
                //Новое пустое поле
                    i0 = i;
                    k0 = k;
                }
        }
   }
```

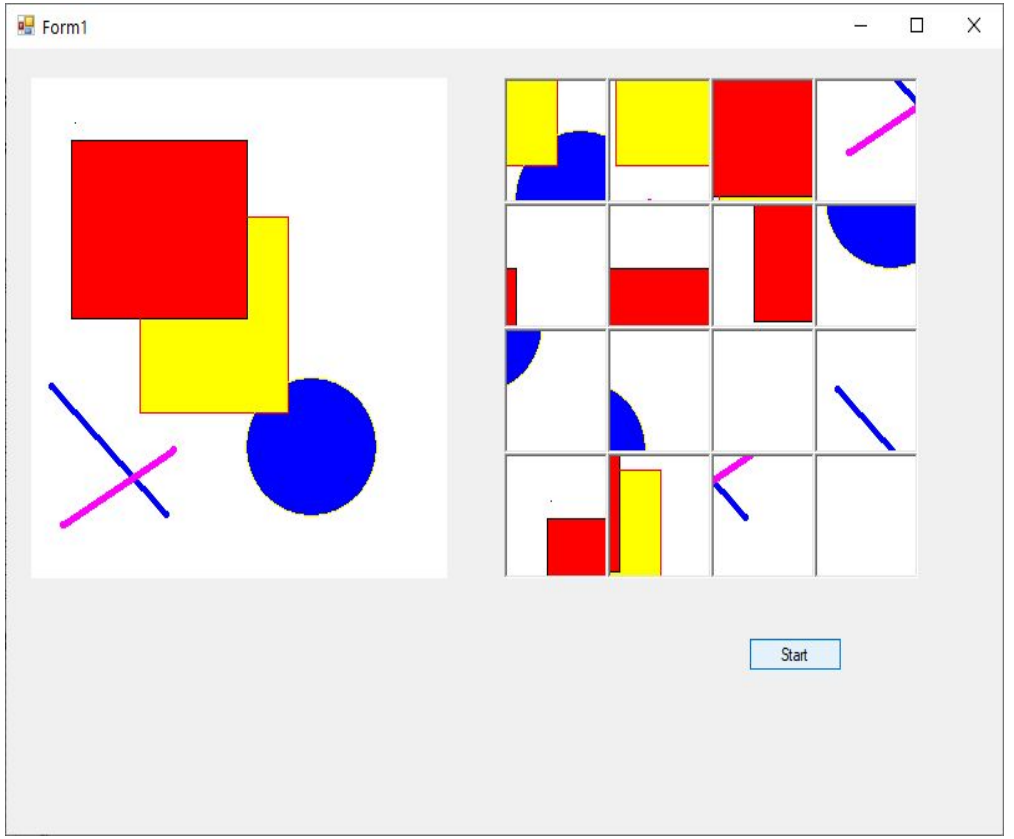
Компонент PictureBox.

Компонент PictureBox обеспечивает отображение иллюстрации (рисунка, фотографии и т. п.). Свойства компонента:

Свойство	Описание
Image	Иллюстрация, которая отображается в поле компонента
SizeMode	Режим отображения иллюстрации (способ масштабирования), если размер иллюстрации не соответствует размеру компонента: Normal — масштабирование не выполняется (если размер компонента меньше размера иллюстрации, то отображается только часть иллюстрации); StretchImage — выполняется масштабирование иллюстрации таким образом, что она занимает всю область отображения (если размер компонента не пропорционален размеру иллюстрации, она будет искажена); AutoSize — размер компонента автоматически изменяется и соответствует размеру иллюстрации; CenterImage — центрирование иллюстрации в поле компонента, если размер иллюстрации меньше размера области отображения
Image.PhysicalDimension	Свойство содержит информацию о размере картинки (иллюстрации), загруженной в поле компонента
Location	Положение компонента (области отображения иллюстрации) на поверхности формы. Уточняющее свойство X определяет расстояние от левой границы области до левой границы формы, уточняющее свойство Y — от верхней границы области до верхней границы клиентской области формы (нижней границы заголовка)
Size	Размер компонента (области отображения иллюстрации). Width определяет ширину области, Height — высоту
BorderStyle	Вид границы компонента: None — граница не отображается; FixedSingle — тонкая; Fixed3D — объемная
Graphics	Поверхность, на которую можно выводить графику (доступ к графической поверхности компонента есть у процедуры обработки события Paint)

Игра Puzzle

В качестве примера использования компонента PictureBox создадим известную настольную игру Puzzle – сборку картинки из ее фрагментов. Выведем на форму рисунок. Затем создадим двумерный массив компонентов PictureBox. С помощью свойства `Pict.Clone(new Rectangle(m * w, k * h, w, h), Pict.PixelFormat)`, копирующего фрагменты рисунка разместим эти фрагменты в элементах массива. С помощью датчика случайных чисел перемешаем фрагменты. Обработчик события OnClick будет осуществлять перестановку фрагментов изображения. В качестве «переменной» для перестановки используем невидимый PictureBox. Игровое поле с перемешанной картинкой приведено на рисунке:



Игра Puzzle. Код программы.

```
namespace Puzzle
{
    public partial class Form1 : Form
    {
        Bitmap Pict;//Переменная для рисунка
        Graphics g;//Графическая поверхность для рисунка
        PictureBox[,] Pb = new PictureBox[4, 4];//МассивPictureBox
        int h, w,i0,j0;
        int clik = 1;//Счет кликов
        public Form1()
        {
            InitializeComponent();
            g = this.CreateGraphics();//Создаем поверхность
        }
```

Игра Puzzle. Код программы.

```
private void button1_Click(object sender, EventArgs e)
  {//Обработчик кнопки Start
     int i, j, k, m;
     bool zap;
     int[,] a = new int[4, 4];//Массив используется для
                                  // перемешивания
     Image bb;
     int x0 = 400;
     int y0 = 20;
     Random rnd = new Random();
```

Игра Puzzle. Код программы.

```
// Если Pb[i,j] есть на форме, мы его удаляем
      for (i = 0; i < 4; i++)
                for (j = 0; j < 4; j++)
                    if (this.Controls.Contains(Pb[i, j]))
                        this.Controls.Remove(Pb[i, j]);
            for (i = 0; i < 4; i++)
                for (j = 0; j < 4; j++)
                    a[i, j] = 0;//Заполняем нулями
```

Игра Puzzle. Код программы.

```
for (i = 0; i < 4; i++)
      for (j = 0; j < 4; j++)
      {//Создаем массив
            Pb[i, j] = new PictureBox();
            Pb[i, j].Left = x0 + j * w;
            Pb[i, j].Top = y0 + i * h;
            Pb[i, j].Width = w;
            Pb[i, j].Height = h;
    //Граница должна быть видна
            Pb[i, j].BorderStyle = BorderStyle.Fixed3D;
            Pb[i, j].Click += Pb_Click);
            Controls.Add(Pb[i, j]);
```

Игра Puzzle. Код программы.

```
zap = true;
        while (zap)
        {
            k = rnd.Next(0, 4);
            m = rnd.Next(0, 4);
            if (a[k, m] == 0) //Если в элементе
                              // массива Pb нет рисунка
            {
                a[k, m] = 1;
                zap = false;
```

Игра Puzzle. Код программы.

```
            //Здесь мы копируем из рисунка фрагмент
          //Копирование идет не с рисунка на форме,
          // а из рисунка в памяти.
          //Поэтому координаты левого верхнего угла - 0,0.
bb = Pict.Clone(new Rectangle(m * w, k * h, w, h),
                                                Pict.PixelFormat);
Pb[i, j].Image = bb;    //Вставляем фрагмент и Pb[i,j]
                    }


                }
            }
} // Закрыли Обработчик кнопки Start
```

Игра Puzzle. Код программы.

```
public void Pb_Click(object sender, EventArgs e)
  {//Обработчик щелчка по Pb[i,j]
     int x0 = 400;
     int y0 = 20;
     PictureBox Pib;
     Pib = (PictureBox)sender;
     if (clik == 1)
     {   j0 = (Pib.Left - x0) / w; //Номер столбца и строки элемента,
         i0 = (Pib.Top - y0) / h; //по которому кликнули - запомнили.
         pictureBox1.Image = Pib.Image;//Изображение сохранили
         clik = 2;
      }
      else
            {//Перестановка
                Pb[i0, j0].Image = Pib.Image;
                Pib.Image = pictureBox1.Image;
                clik = 1;
            }
        }
```

Игра Puzzle. Код программы.

```
private void Form1_Paint(object sender, PaintEventArgs e)
        {//Рисуем картинку
            Pict = new Bitmap("D:/C#/My_Text/puzzle.bmp");
            g.DrawImage(Pict, 20, 20, Pict.Width, Pict.Height);
            //Размеры фрагмента рисунка
            w = Pict.Width / 4;
            h = Pict.Height / 4;
            //Невидимый pictureBox1 для перестановки фрагментов
            pictureBox1.Width = w;
            pictureBox1.Height = h;
            pictureBox1.Visible = false;
        }
    }
```