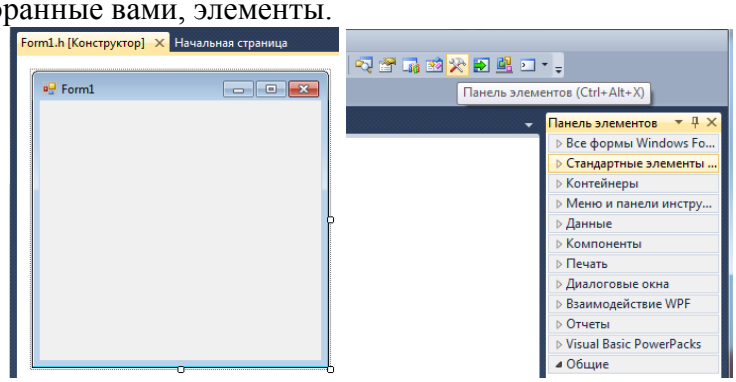
Сценарии ознакомительных практических занятий
Создание приложения Windows Forms в MVS C#

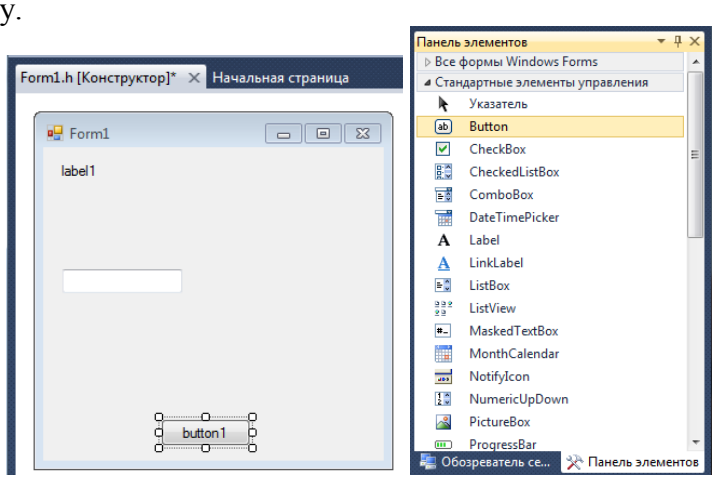
Введение
Вы познакомитесь с тем, как делать программы с всевозможными кнопочками, текстовыми полями, иконками, окошками и прочими элементами, которые все привыкли видеть. Есть огромнейшее количество всевозможных программ и все они содержат форму и огромнейшее количество разных элементов. Вы приобретёте первоначальные навыки и опыт, чтобы создать свою программу. Сначала будут предоставлены задания, в которых рассматриваются самые необходимые элементы и их свойства. Но особых, радикальных различий от “C++” (базовый вариант) вы не встретите. Если говорить вкратце, то в “C++” свойство или метод могут вызываться стрелочкой "->" и двойным двоеточием "::", а в ”C#” всё вызывается точкой “.”

Занятие 1. Элементы button, textBox и label

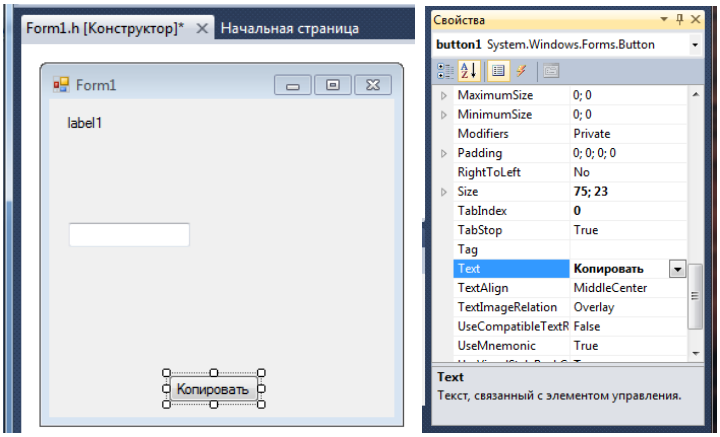
Создание первой простой программы, суть которой в том, что вы пишите в текстовом поле какое-то предложение, нажимаете на копку и на форме появляется надпись – тоже самое предложение. Чтобы создать проект в Windows Forms нужно запустить Microsoft Visual Studio C#, нажать “Создать проект”, выбрать (слева) “CLR”, далее выберите “Приложение Windows Forms” и назовите свой проект.
Итак, познакомимся сначала с интерфейсом компилятора: справа от вас находиться панель всевозможных инструментов (элементов), а слева, собственно, сама форма, на которую вы и будете размещать, выбранные вами, элементы.



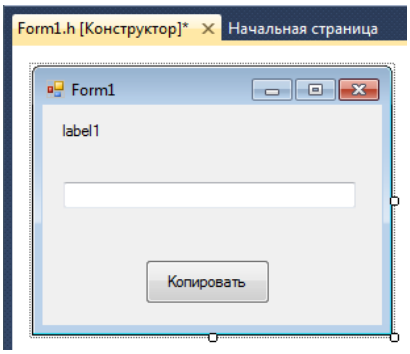
Нажмём на “Стандартные элементы” и выберем из них – "button", "textbox" и "label, перетаскивая их поочерёдно на форму.



Нажав, например, на кнопку "button1", справа вы увидите её свойства, их там очень много. Для начала выберете свойство "Text" и напишите там вместо "button1"– “Копировать”. В дальнейшем большую часть свойств можно изменять программным кодом.



Так же заметили, что размер формы и элементов можно изменять. Сделайте так же как показано здесь.

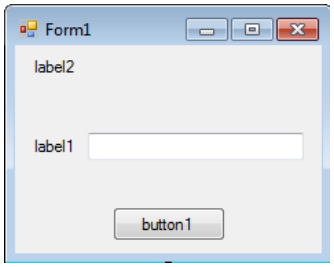


Далее щёлкните два раза по форме – перед вами раскроется новое окно с программным кодом, в котором мы и будем, собственно, писать код. В данном случае откроется событие Form_Load, где вам нужно написать label1.Text = ""; Эта строка говорит о том, что, когда произойдёт запуск (загрузка) программы, текст "label1" станет равным "".

После этого нажмите наверху "Form1.h[Конструктор]". Откроется Событие "button1_Click". Напишите в нём label1.Text = textBox1.Text; Это будет говорить о том, что когда, при загрузке формы, вы нажмёте (кликните) на кнопку произойдёт описанное действие – текст, написанный, в “textBox1” скопируется в "label1". Далее приведён код на "C#".

```
namespace One_1_
{
public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 label1.Text = "";
 }

private void button1_Click(object sender, EventArgs e)
 {
 label1.Text = textBox1.Text;
}

}
}
```

Создание простой программы, в которой при нажатии на кнопку будет “выскакивать” небольшое окошко, сообщающее нам о том, какой текст был записан в текстовом поле. Для этого понадобятся всего лишь два элемента - "button" и "textBox". Создайте проект, назовите его и перенесите на форму "button" и "textBox", а так же два "label". Далее кликните два раза сначала по форме, а затем по кнопке. Вот как это должно выглядеть:



Далее напишите ниже находящийся код. Вы увидите как к тексту "MessageBox" прибавляется текст, написанный вами в "textBox1", далее через запятую мы придаём название “выскочившему окошку”, так же как и названию вашей формы.

```
namespace Two_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 this.Text = "Форма приветствия";
label1.Text = "Name: ";
 label2.Text = "Напишите ваше имя.";
button1.Text = "Ввод";
 }

private void button1_Click(object sender, EventArgs e)
 {
 MessageBox.Show("Здравствуй " + textBox1.Text + "!", "Приветствие");
 }
 }
}
```

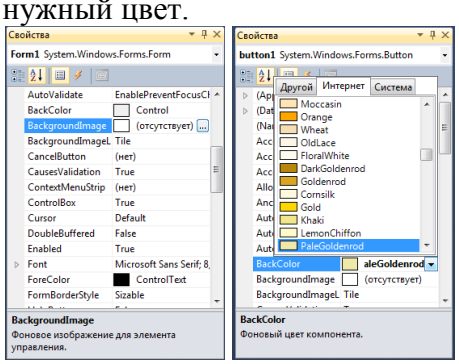
Рассмотрим, по своей сути очень маленький, но одновременно очень актуальный элемент – подсказка "TollTip". Конечно же все на раз замечали, как при наведении курсором мыши на какой-нибудь элемент интерфейса программы, появляется маленькое текстовое окошко, в котором написана какая та информация, касающаяся данного элемента. Для примера возьмём предшествующую программу. Нам останется дописать буквально пару строк. Задача следующая – при наведении курсора мыши на текстовое поле, будет появляться маленькое текстовое поле, сообщающее о том, что здесь нужно ввести ваше имя. Перенесите на форму из панели инструментов два "label", один “button” и nolcказку "ToolTip". После стрелки "toolTip1." указывается стиль подсказки – "Balloon", затем в скобках пишется элемент, возле которого должна появляться подсказки, а после запятой в кавычках пишется сам текст подсказки. Вот код реализации:

```
namespace Three_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 { this.Text = "Форма приветствия";
 label1.Text = "Name: ";
 label2.Text = "Напишите ваше имя.";
button1.Text = "Ввод";
 //------------ реализация ToolTip
 toolTip1.SetToolTip(textBox1, "Введите\nваше имя");
toolTip1.IsBalloon = true;
 }

private void button1_Click(object sender, EventArgs e)
 {
MessageBox.Show("Здравствуй " + textBox1.Text + "!", "Приветствие");
 }
 }
}
```

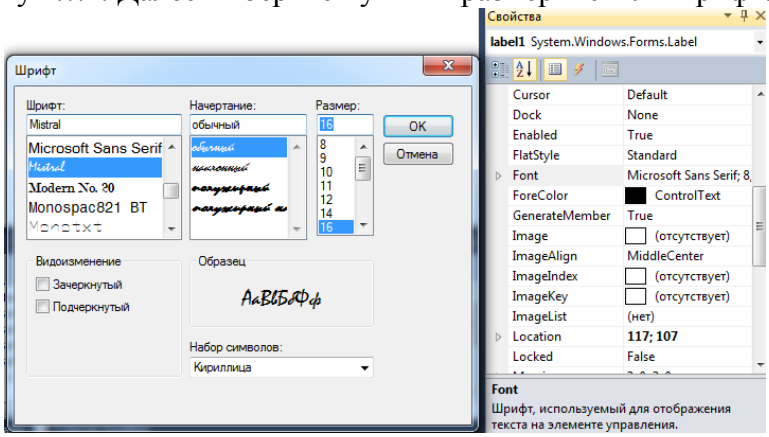
Занятие № 3. Изменение шрифта текста и цвета формы и элементов

Вполне может оказаться, что ваша программа требует более оригинального оформления, чем стандартное. Имеется в виду - задать цвет кнопки, задать фон формы, загрузив изображение. Для наглядного примера создадим проект, в котором будем записывать на фоне изображения текст, задав изображения для заднего фона ("BackGroundImage") формы и изменив цвет кнопки. Для этого нам понадобятся "textBox", "label", кнопка "button" и изображение:

Создав новый проект в "Windows Forms" нажмите на форму, слева вы увидите её свойства – нам нужно "BackGroundImage":
После этого нажмите на кнопку "…" и выберите "Локальный ресурс", после чего нажмите на кнопку “Импорт”. Откроется проводник – вам нужно открыть в нем сохранённое изображение, пример которого был показан выше. Далее нажимаете на элемент "button", выбираете в его свойствах "BackColor" и ставите нужный цвет.



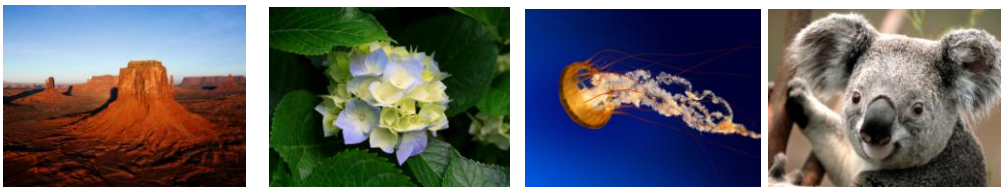
Теперь нужно изменить шрифт элемента "label”. Для этого нажмите на него и выберите свойство "Font", нажав на кнопку “…”. Далее выберите нужный размер и стиль шрифта:



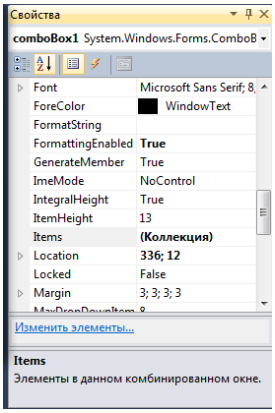
Теперь перейдём к коду.

```
namespace Four_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 this.Text = "Доска объявлений";
label1.Text = "";
 }

private void button1_Click(object sender, EventArgs e)
 {
 label1.Text = textBox1.Text;
 }

 }
 }
```

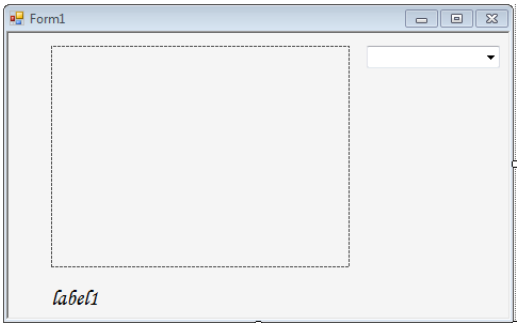
Загрузка изображения в PictureBox при помощи ComboBox
Как загрузить изображение в элемент "PictureBox" через панель инструментов, вы могли узнать ранее. Теперь посмотрим как загружать изображение в коде. Так же вы познакомитесь с функционированием такого элемента, как "comboBox". Суть программы следующая – есть четыре картинки:



Они находятся на диске или на флешке. На форме есть "comboBox", в котором находится некоторый список из четырёх слов. При выборе одного из слов в списке должна появляться картинка, а в "label" её название. Для того что бы занести в "comboBox" некоторый список, нужно найти в панели свойств - свойство "Items" и написать через "enter" слова:



Помимо "comboBox", перенесите на форму элементы – "lable" и "PictureBox". Стиль текста "label" вы можете выбрать сами – ранее это рассматривается. Вот как должна выглядеть заготовка программы:

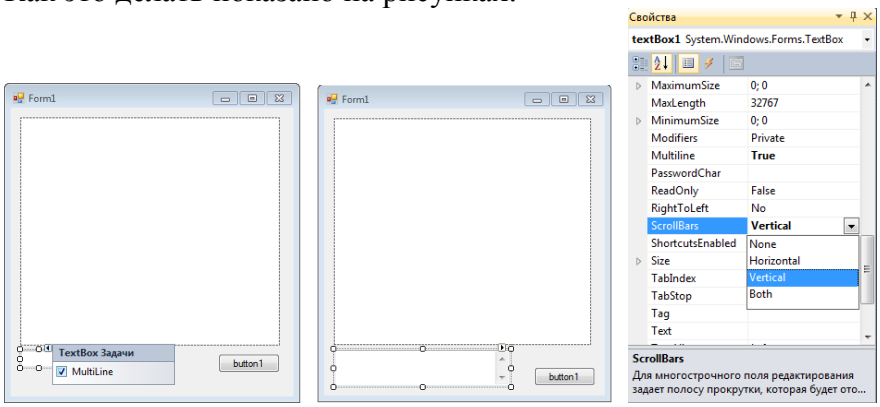


Перейдём к коду.

```
namespace Seven_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 this.Text = "Фото галерея";
label1.Text = "";
 comboBox1.Text = "Список";
 }

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
 {
 switch (comboBox1.SelectedIndex)
 {
 case 0: pictureBox1.Image=Image.FromFile("d:\\7.0.png");
 label1.Text = "Лето"; break;
 case 1: pictureBox1.Image=Image.FromFile("d:\\7.1.png");
 label1.Text = "Солнце"; break;
 case 2: pictureBox1.Image=Image.FromFile("d:\\7.2.png")
; label1.Text = "Море"; break;
 case 3: pictureBox1.Image=Image.FromFile("d:\\7.3.png");
 label1.Text = "Пляж"; break;
 }
```

```
 }

 }
}
```

Рисование текста на PictureBox
Нужно написать в коде, чтобы текст написанный в поле элемента "textBox" появился на поле "PictureBox" определённого размера, определённого цвета и в определённом месте. Создайте проект в приложение "Windows Forms" и перетащите на форму три элемента: 1"textBox", 1"PictureBox" и 1"button". У "textBox" нужно включить режим "Multeline" и свойство "ScrolBars". Как это делать показано на рисунках:

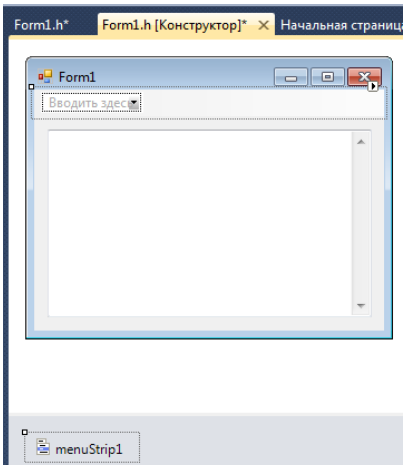


В коде программы задаётся размер, шрифт и цвет отображаемого текста.

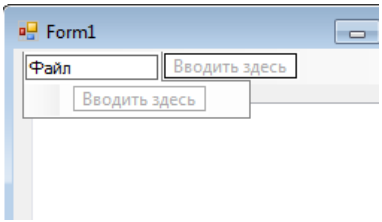
```
namespace Paint_Text_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 Font = new System.Drawing.Font("Times New Roman", 12, FontStyle.Bold);
 button1.Text = "Рисовать";
 }

private void button1_Click(object sender, EventArgs e)
 {
 String Text = String.Format("{0}", textBox1.Text);
 Brush Кисть = new SolidBrush(Color.LimeGreen);
 Graphics G = pictureBox1.CreateGraphics();
 G.TextRenderingHint = System.Drawing.Text.TextRenderingHint.AntiAlias;
 G.DrawString(Text, Font, Кисть, 150, 50); // Координаты размещения текста
}

 }
 }
```

Занятие № 4. Элемент MenuStrip и свойство Anchor, Открытие и запись текстового файла

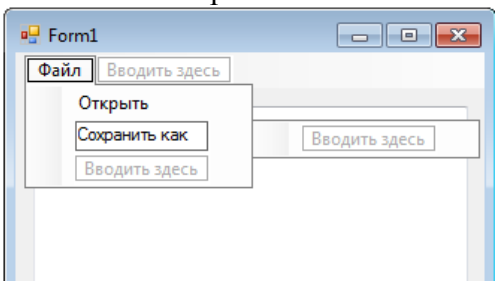
Рассмотрим довольно важный и распространённый элемент "MenuStrip", который вы можете встретить: в текстовом редакторе, выбирая "Файл->Сохранить как", в графическом редакторе "Файл->Вставить" и во многих других программах. "MenuStrip" – это, по сути, выпадающее меню, в котором есть определённые пункты. Так же будет рассмотрено свойство "Anchor" – это свойство, при котором определяется к каким сторонам формы будет привязан элемент, если к правой, то при увеличении или уменьшении размера формы – правая сторона "textBox", так же увеличиваться или уменьшаться. Если поставлена кнопка"button", то она будет перемещаться за стороной формы, к которой привязана. Перенесите на форму элемент "MenuStrip" и элемент "textBox". У элемента “textBox” включите "Multeline" и свойство "ScrollBars" "Vertical".



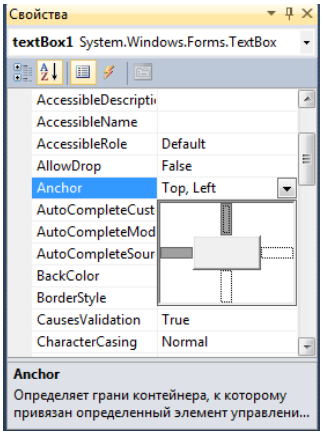
Далее нажмите на текстовое поле "Вводить здесь" и напишите – "Файл:



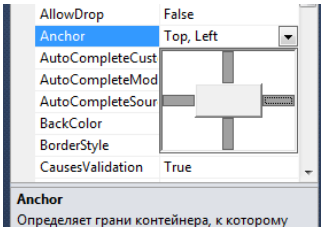
После этого в нижнем, новом появившемся текстовом поле, напишите "Открыть", после чего появиться ещё одно, в котором напишите "Сохранить как":



Теперь займёмся свойством "Achor" – для этого нажмите на "textBox". После чего на панели свойств, расположенной слева, выберите свойство "Achor":

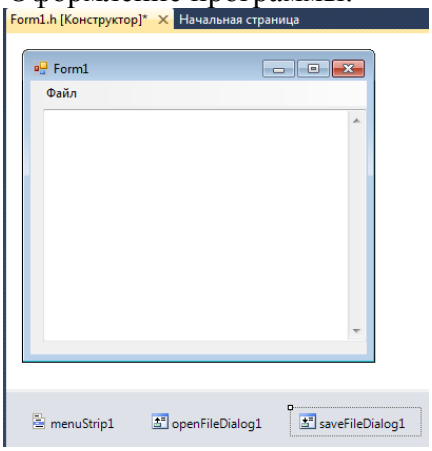


Как в можете видеть - “textBox” привязан к левой и верхней части формы. Теперь выделите так же правую и нижнюю часть, после чего размер элемента “textBox” будет изменяться пропорционально размеру формы:



Запустите программу и проверьте изменённые свойства.

В данной программе мы будем открывать какой-то уже созданный текстовый файл и редактировать его или же сами писать текст и сохранять его, как новый текстовый файл в нужную вам папку. Ещё один явный признак текстового редактора – это если вы что-то написали и нажимайте крестик чтобы выйти – при этом программа спрашивает: “Сохранить изменения”. Создание такого текстового редактора как раз и будет рассматриваться. Для этого понадобятся следующие элементы: "MenuStrip", "textBox", "openFiledialog", "saveFiledialog". Перетащите все эти элементы на форму, назовите заголовок "MenuStrip" "Файл" создайте в нём три пункта: "Открыть", "Сохранить как", "Выход", привяжите "textBox" ко всем сторонам формы, включите "Multeline" и "ScrollBars Vertical". Оформление программы:



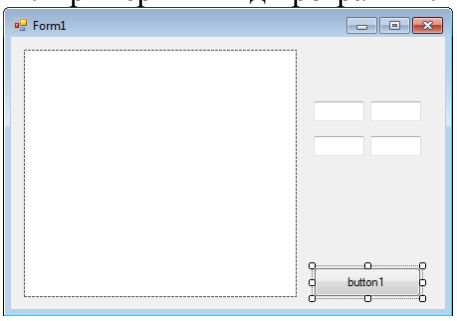
В коде программы будут созданы "MyReader" и "MyWriter", с помощью которых программа будет читать и записывать текст в файл. Помимо этого в коде создаётся кодировка, благодаря которой программа будет понимать русский текст. У формы нужно вызвать событие "FormClousing". Далее представлен код программы на C++, проанализируйте его и адаптируйте для C#.

```
#pragma endregion
private: System::Void Form1_Load(System::Object^ sender, System::EventArgs^ e) {
 this->Text = "Текстовый редактор";
openFileDialog1->FileName = "D:\\ВУЗ\\Text2.txt";
 openFileDialog1->Filter = "Текстовые файлы (*.txt)|*.txt|All files (*.*)|*.*";
 saveFileDialog1->Filter = "Текстовые файлы (*.txt)|*.txt|All files (*.*)|*.*";
 }

private: System::Void открытьToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^
e) {
 openFileDialog1->ShowDialog();
 if (openFileDialog1->FileName == nullptr) return;
 try
 { auto MyReader = gcnew IO::StreamReader(openFileDialog1->FileName,
System::Text::Encoding::GetEncoding(1251));
 textBox1->Text= MyReader->ReadToEnd();
 MyReader->Close();
```

```
 }
 catch (IO::FileNotFoundException^ Ситуация)
 {
MessageBox::Show(Ситуация->Message + "\nФайл не найден", "Ошибка", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);
 }
 catch (Exception^ Ситуация)
 {
MessageBox::Show(Ситуация->Message, "Ошибка", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);
 }
 }

private: System::Void сохранитьКакToolStripMenuItem_Click(System::Object^ sender,
System::EventArgs^ e) {
saveFileDialog1->FileName = openFileDialog1->FileName;
if (saveFileDialog1->ShowDialog() == Windows::Forms::DialogResult::OK) Save();
 }
 void Save()
 {
 try
 {// Создание экземпляра StreamWriter для записи в файл:
auto MyWriter = gcnew IO::StreamWriter(saveFileDialog1->FileName, false,
System::Text::Encoding::GetEncoding(1251));
 MyWriter->Write(textBox1->Text);
 MyWriter->Close(); textBox1->Modified = false;
 }
 catch (Exception^ Ситуация)
 {MessageBox::Show(Ситуация->Message, "Ошибка", MessageBoxButtons::OK,
   MessageBoxIcon::Exclamation);
 }
 }

private: System::Void выходToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e)
{ this->Close();
 }

private: System::Void Form1_FormClosing(System::Object^ sender,
System::Windows::Forms::FormClosingEventArgs^ e) {
 if (textBox1->Modified == false) return;
 auto MeBox = MessageBox::Show("Текст был изменён. \nСохранить изменения?",
 "Простой редактор", MessageBoxButtons::YesNoCancel, MessageBoxIcon::Exclamation);
 if (MeBox == Windows::Forms::DialogResult::No) return;
 if (MeBox == Windows::Forms::DialogResult::Cancel) e->Cancel = true;
 if (MeBox == Windows::Forms::DialogResult::Yes)
 {
 if (saveFileDialog1->ShowDialog() == Windows::Forms::DialogResult::OK)
 { Save(); return;
 }
 else e->Cancel = true;
 }
 }
 };
}
```

Занятие № 5. Рисование линий, треугольника, эллипса и окружности в PictureBox

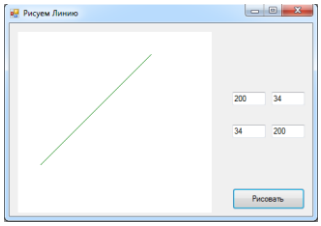
Изучение базовых приёмов рисования в "PictureBox". Рассмотрим как нарисовать линию заданной длины, цвета и в заданном месте поля "PictureBox". Для этого нам понадобятся: 4 "textBox", 1 "button", и конечно же "PictureBox". Примерный вид программы:



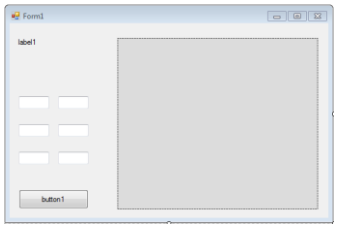
То, как изменить цвет фона элемента (“PictureBox”) рассматривается ниже. В “textBox”ы мы будем записывать начальное и конечное значение координат поля рисования: две координаты – одна точка, ещё две координаты – ещё одна точка, которая соединяется с предыдущей, образуя отрезок, заданного в коде программы цвета. Значение каждой координаты, получаемое из текстового поля, мы будем конвертировать в "int"-Convert::ToInt32(textBox1.Text); А для хранения значения создаётся массив, его нужно объявить/

Приступим к коду самой программы, который вы можете видеть ниже. В нём вызвано событие "PictureBox1_Paint", о том, как вызвать событие элемента рассмотрено. В событии "button1_Click" есть строка - pictureBox1.Refresh(); с помощью неё каждый раз, когда пользователь будет вводить новые координаты, изображение в "PictureBox" будет заново прорисовываться.

```
namespace Risovanie_1_
{
 public partial class Form1 : Form
 {
   int[] m_p = new int[5];
   public Form1()
 { InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 { this.Text = "Рисуем Линию";
    button1.Text = "Рисовать";
 }

private void button1_Click(object sender, EventArgs e)
 {
 m_p[1] = Convert.ToInt32(textBox1.Text);
 m_p[2] = Convert.ToInt32(textBox2.Text);
 m_p[3] = Convert.ToInt32(textBox3.Text);
 m_p[4] = Convert.ToInt32(textBox4.Text);
 pictureBox1.Refresh();
 }

 private void pictureBox1_Paint(object sender, PaintEventArgs e)
 { // Рисуем линию
    e.Graphics.DrawLine(System.Drawing.Pens.Green, m_p[1], m_p[2], m_p[3], m_p[4]);
 }
 }
}
```



После того, как вы узнали как рисовать линию, можно уже будет нарисовать какую-нибудь простую фигуру – например треугольник. Для этого нужно нарисовать всего лишь три точки, а не шесть. Сначала может показать, что нужно рисовать три отрезка по отдельности и пытаться задать такие координаты, что бы получился треугольник – это излишнее и неэффективное нагромождение программы. Поэтому нужно сделать так, чтобы конец одного отрезка являлся началом другого – что, собственно, мы и сделаем. Нужно задавать массив из шести элементов для создания трех точек. Для реализации данного проекта понадобятся: 6 "textBox", 1 "button", 1 "PictureBox", 1 "label". В коде данного проекта задаются отрезки чёрного, красного и белого цвета. Вы можете изменить цвет формы и цвет “PictureBox”, так как цвет линий может слиться с цветом фона. Примерный вид формы программы:



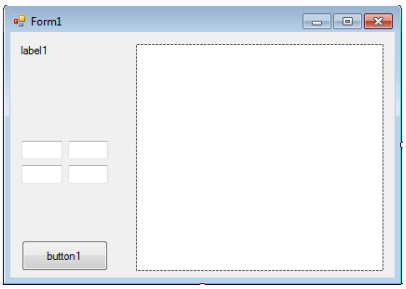
Приступим к коду программы. В этом проекте у элемента "PictureBox" нужно вызвать событие "Paint".

В C# обязательно нужно объявить переменную "index" и задать массив из 7 элементов, а не из шести, как в C++, иначе компилятор выдаст ошибку.

```
namespace Treygolnik_1_
{
 public partial class Form1 : Form
 {
int[] m_p = new int[7];
int index;
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 label1.Text = "Введите координаты";
button1.Text = "Рисовать";
 }

private void button1_Click(object sender, EventArgs e)
 {
 m_p[1] = Convert.ToInt32(textBox1.Text);
 m_p[2] = Convert.ToInt32(textBox4.Text);
 m_p [3] = Convert.ToInt32(textBox2.Text);
 m_p [4] = Convert.ToInt32(textBox5.Text);
 m_p [5] = Convert.ToInt32(textBox3.Text);
 m_p [6] = Convert.ToInt32(textBox6.Text);
```

```
 index = 1;
 pictureBox1.Refresh();
 }

private void pictureBox1_Paint(object sender, PaintEventArgs e)
 {
 if (index == 1)
 {
 e.Graphics.DrawLine(System.Drawing.Pens.Black, m_p[1], m_p[2], m_p[3], m_p[4]);
 e.Graphics.DrawLine(System.Drawing.Pens.Red, m_p[3], m_p[4], m_p[5], m_p[6]);
 e.Graphics.DrawLine(System.Drawing.Pens.White, m_p[5], m_p[6], m_p[1], m_p[2]);
 }
 }

 }
}
```

Теперь будет показан пример создания "Пера" или "Карандаша", которые будет задавать цвет контура фигуры. Чтобы нарисовать эллипс нужно указать координаты верхнего левого угла условного прямоугольника, в котором находится рисуемая фигура, и радиусы. Если горизонтальный и вертикальный радиусы равны, то получится окружность, а если нет, то получится эллипс. Для создания данного проекта понадобятся: 4"textBox", 1"button", 1"label" и 1"PictureBox". Перетащите все перечисленные элементы на форму и оформите программу следующим образом:

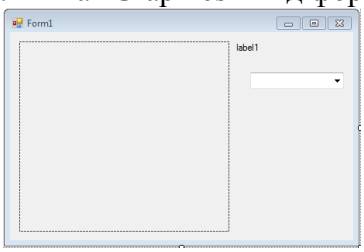


Теперь перейдём к коду. Так же, как и ранее создаём массив из 5-ти элементов, в которых будут хранится данные, вводимые в "textBox". Что бы компилятор понял, что вы хотите нарисовать эллипс – нужно написать "DrawEllipse".
В C# нужно объявить переменную "index", что бы проект был успешно скомпилирован:

```
namespace Ellipse_1_
{
 public partial class Form1 : Form
 {
int[] m_p = new int[5];
int index;
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 this.Text = "Рисование эллипса";
label1.Text = "Введите данные";
button1.Text = "Рисовать";
 }
```

```
private void button1_Click(object sender, EventArgs e)
 {
 index = 1;
 m_p[1] = Convert.ToInt32(textBox1.Text);
 m_p[2] = Convert.ToInt32(textBox2.Text);
 m_p[3] = Convert.ToInt32(textBox3.Text);
 m_p[4] = Convert.ToInt32(textBox4.Text);
 pictureBox1.Refresh();
 }

private void pictureBox1_Paint(object sender, PaintEventArgs e)
 {
 if (index == 1)
 {
 Pen Перо = new Pen(Color.Black);
 e.Graphics.DrawEllipse(Перо, m_p[1], m_p[2], m_p[3], m_p[4]);
 }
 }

 }
}
```

Что бы фигура стала закрашенной, нужно создать "заливку" типа "Brush", указав цвет заливки. Создадим проект, в котором с помощью "comboBox" будем выбирать цвет заливки нарисованного прямоугольника. Для этого на форму нужно перетащить три элемента: "label", "textBox" и "PictureBox". Необходимо задать список "comboBox" в коде программы с помощью массива, а также создать переменную "Графика" типа "Graphics" Вид формы программы:



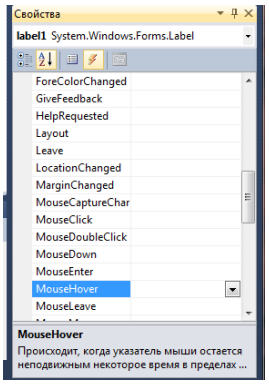
Необходимый код представлен ниже на C++, требуется адаптировать его для C#. В коде вы встретите строку: "Графика->Clear(SystemColors::Control);" – она будет очищать, то есть закрашивать "PictureBox" в цвет "Control".

```
#pragma endregion
private: System::Void Form1_Load(System::Object^ sender, System::EventArgs^ e) {
 this->Text = "Закрашивание фигур";
label1->Text = "Выберите фигуру";
comboBox1->Text = "Фигуры";
array^ Фигуры = gcnew array{"Прямоугольник","Эллипс","Окружность"};
comboBox1->Items->AddRange(Фигуры);
 }

private: System::Void comboBox1_SelectedIndexChanged(System::Object^ sender, System::EventArgs^
e) {
 Graphics ^ Графика = pictureBox1->CreateGraphics();
 Brush ^ Заливка = gcnew SolidBrush(Color::Orange);
Графика->Clear(SystemColors::Control);
 switch (comboBox1->SelectedIndex)
 {
```

```
 case 0:
Графика->FillRectangle(Заливка, 60, 60, 120, 180); break;
 case 1:
Графика->FillEllipse(Заливка, 60, 60, 120, 180); break;
 case 2:
Графика->FillEllipse(Заливка, 60, 60, 120, 120); break;
 }
 }

 };
}
```

Занятие № 6. Событие MauseHover

Сейчас вы познакомитесь с “событием” "MausHover". Идея вполне простая – если вы наводите на один из элементов, на которой описано данное событие, то элемент как-либо изменяется - как напишите, так и изменится. В данном примере, при наведении курсора мыши на надпись "label" после чего изменяется текст и цвет текста, а так же “выскакивает \” “MessageBox”, с каким-нибудь текстом. Создайте проект в приложение “Windows Forms”, после чего перенесите на форму элемент "label". Теперь нажмите на "label", справа вы должны увидеть список его свойств, а помимо этого наверху будет значок похожий на жёлтую молнию, нажимайте на него и выбирайте событие “MauseHover”, два раза кликнув на него. Это выглядит так:



Далее надо будет прописать для текста “label” – text align center, что бы надпись была в середине относительно самой себя – это как в “Microsoft Word” выбрать “По центру”. Теперь перейдём к коду

```
namespace Three_1_
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

private void Form1_Load(object sender, EventArgs e)
 {
 this.Text = "Hover";
 label1.TextAlign = ContentAlignment.MiddleCenter;
 label1.Text = "Не трогай.";
 }

private void Form1_MouseHover(object sender, EventArgs e)
 {
```

```
label1.TextAlign = ContentAlignment.MiddleCenter;
label1.Text = "ERROR!!!";
label1.ForeColor = Color.Red;
MessageBox.Show("Написано же\nНЕтрогать!", "Fatal ERROR!",
MessageBoxButtons.OK, MessageBoxIcon.Error);
}

}
}
```