

## ПРАКТИЧЕСКАЯ РАБОТА № 18

(4 академических часа)

**Тема: Создание Windows-приложений. Создание графического редактора.**

**Цель задания:**

1. Понять основы работы с графикой в Visual Studio.
2. Научиться использовать функции и типы графической библиотеки GDI+.

**Теоретическая часть.**

.NET Framework реализует расширенный графический интерфейс GDI+, обладающий широким набором возможностей. Для рисования в формах достаточно иметь три объекта - перо, кисть и контекст области, в которой осуществляется рисование.

Класс *Graphics* - основной класс, необходимый для вывода графики и предоставляющий множество методов для изображения геометрических фигур и различных графических объектов, находится в пространстве имен *Drawing*.

Класс *Graphics* не имеет конструкторов, т.к. объекты этого класса зависят от контекста конкретных устройств вывода. Создаются объекты специальными методами разных классов, например, он может быть создан из объекта *Bitmap*, или к нему можно получить доступ, как к некоторому объекту, инкапсулированному в некоторые элементы управления, в том числе, и в объект формы приложения.

Объект **Graphics** можно создать тремя различными способами.

1. Использование метода *CreateGraphics()* формы или элемента управления, на котором надо отобразить графику.

```
Graphics Graph1 = pictureBox1.  
CreateGraphics ( );
```

2. Создание растрового изображения, которое можно сохранить как графический файл.

```
Bitmap image1 = new Bitmap(200,200);  
Graphics Graph1 = Graphics.FromImage(image1);  
pictureBox1.Image = image1;
```

3. Использование события *Paint* формы или элемента управления, которое происходит при их создании или обновлении. В обработчике этого события одним из аргументов является *e* типа *System.Windows.Forms.PaintEventArgs*. В программном коде обработчика события можно объявить создание объекта *Graph1* типа *Graphics* как свойства аргумента *e*.

```
private void pictureBox1_Paint(object sender, System.Windows.Forms.PaintEventArgs e)  
{  
    Graphics Graph1 = e.Graphics;  
}
```

**Объекты для работы с графикой**

1. **Перо.** Объект *Pen* (Перо) определяет цвет и ширину линии рисования. В разделе объявления переменных необходимо определить имя объекта (например, *Pen1*), установить цвет (например, красный *Color.Red*) и ширину линии в пикселях

```
Pen Pen1 = new Pen (Color.Red, 3);
```

2. **Кисть.** Объект *Brush* (кисть) определяет цвет и стиль закрашивания прямоугольников, окружностей и других замкнутых фигур. В разделе объявления переменных необходимо определить имя объекта (например, *Brush1*) и установить тип закрашки и цвет (например, сплошная закрашка синего цвета *SolidBrush (Color.Blue)*).

```
SolidBrush Brush1 = new SolidBrush(Color.Blue);
```

3. **Цвет.** Цвет устанавливается как значение свойства *Color*. Можно установить цвет с использованием нескольких десятков цветовых констант.

```
Pen1.Color = Color.Green;  
Brush1.Color = Color.Yellow;
```

Цвет пера или кисти можно также установить с использованием элемента управления *ColorDialog*:

```
ColorDialog1.ShowDialog();  
Pen1.Color = ColorDialog1.Color;
```

### Графические методы

Графические фигуры рисуются с использованием графических методов. Замкнутые фигуры, такие как прямоугольники или эллипсы, состоят из двух частей - контура и внутренней области. Контур рисуется с использованием заданного пера, а внутренняя область закрашивается с использованием заданной кисти.

1. *DrawLine()* - метод рисования линии, аргументами которого являются перо определенного цвета и толщины (например, *Pen1*), а также координаты концов линии *X1*, *Y1* и *X2*, *Y2*.

```
Graph1.DrawLine (Pen1, X1, Y1, X2, Y2);
```

2. *DrawRectangle()* - метод рисования прямоугольника, аргументами которого являются перо определенного цвета и толщины (например, *Pen1*), а также координаты левого верхнего угла *X1*, *Y1*, ширина *Width* и высота *Height*.

```
Graph1.DrawRectangle (Pen1, X1, Y1, Width, Height);
```

3. *FillRectangle()* - метод закрашки прямоугольника с использованием кисти определенного цвета.

```
Graph1.FillRectangle(Brush1, X1, Y1, Width, Height);
```

4. *DrawEllipse()* - метод рисования окружности или эллипса, аргументами которого являются перо определенного цвета и толщины (например, *Pen1*), а также координаты левого верхнего угла описанного прямоугольника *X1*, *Y1*, ширина *Width* и высота *Height*.

```
Graph1.DrawEllipse(Pen1, X1, Y1, Width, Height);
```

5. *FillEllipse()* - метод закрашки окружности или эллипса с использованием кисти определенного цвета.

```
Graph1.FillEllipse(Brush1, X1, Y1, Width, Height);
```

6. Для рисования точки с заданными координатами *X1* и *Y1* можно использовать методы *DrawRectangle (Pen1, X1, Y1, 1, 1)* или *DrawEllipse(Pen1, X1, Y1, 1, 1)*, в которых аргументы *Width* и *Height* равны 1.

7. *Graph1.Clear()* – метод заданным цветом (например, белым) стирает изображения в области рисования.

```
Graph1.Clear(Color.White);
```

### Рисование текстом

Метод *Drawstring()* позволяет выводить текст в область рисования. Аргументами метода является строка текста, шрифт, кисть и координаты начала строки. Объекты шрифт (например, *drawFont*) и кисть (например, *drawBrush*) необходимо объявить.

```
Font drawFont = new Font("Arial"12);  
Brush drawBrush = new SolidBrush(Color.Black);
```

### Задание

Создать приложение графический редактор, реализующее следующие функции:

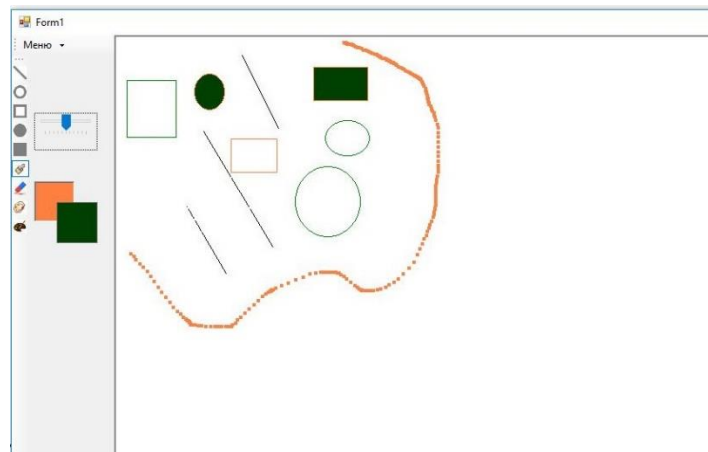
- создание, просмотр и сохранение изображений;
- рисование различных фигур с заливкой и без заливки;
- ввод координат осуществлять щелчками по графическому полю;
- для рисования фигур использовать меню и панель инструментов;

- обеспечить возможность установки для пера и кисти любого цвета из полной палитры цветов.

### Технология работы

1. Разработку приложения необходимо начать с разработки формы:
  - определить размеры и положение формы;
  - задать заголовок формы;
  - определить цветовую гамму;
  - разместить меню и определить пункты.
  - создать панель инструментов: кнопки, обеспечивающие рисование графических примитивов и очистку поля рисования.
2. Определить и поместить на форму поле рисования - элемент управления, который будет использоваться в качестве области рисования.
3. Добавить необходимые окна диалогов (открытие, сохранение, выбор цвета).
4. Выполнить настройку свойств элементов управления.
5. Написать соответствующие обработчики событий.
6. Выполнить тестирование приложения.

### Пример:



### Контрольные вопросы:

1. Существует ли возможность рисования на форме?
2. Перечислите форматы графических файлов, которые можно загружать в объект PictureBox.
3. Способы создания объекта Graphics.
4. От чего зависит количество пикселей, отображаемых на экране?
5. Какие инструменты рисования (объекты) позволяют строить изображения?