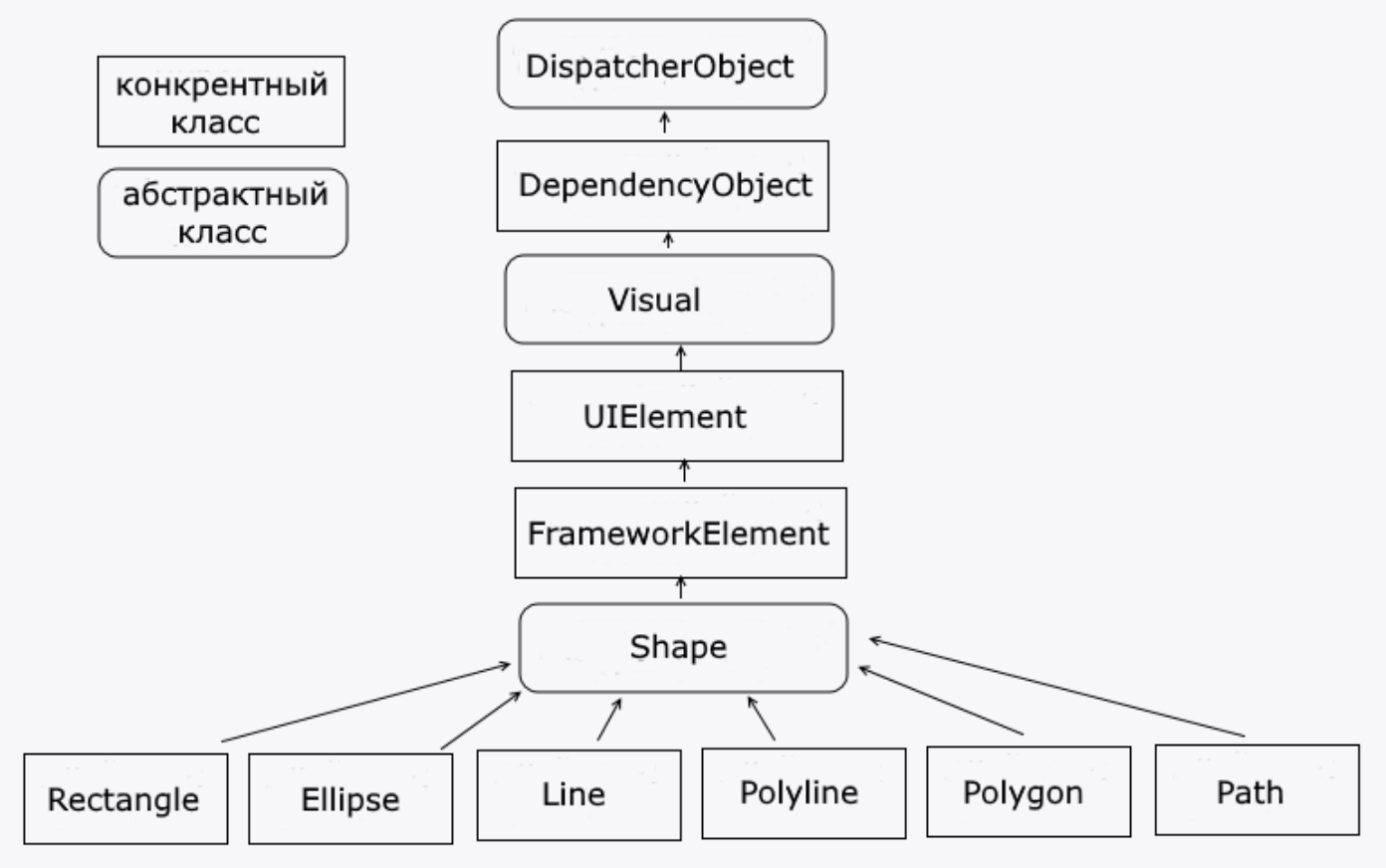
Одним из способов построения двухмерной графики в окне - это использование фигур. Фигуры фактически являются обычными элементами как например кнопка или текстовое поле. К фигурам относят такие элементы
как **Polygon** (Многоугольник), **Ellipse** (овал), **Rectangle** (прямоугольник), **Line** (обычная линия), **Polyline** (несколько связанных линий). Все они наследуются от абстрактного базового класса **System.Windows.Shapes.Shape**:



От базового класса они наследуют ряд общих свойств:

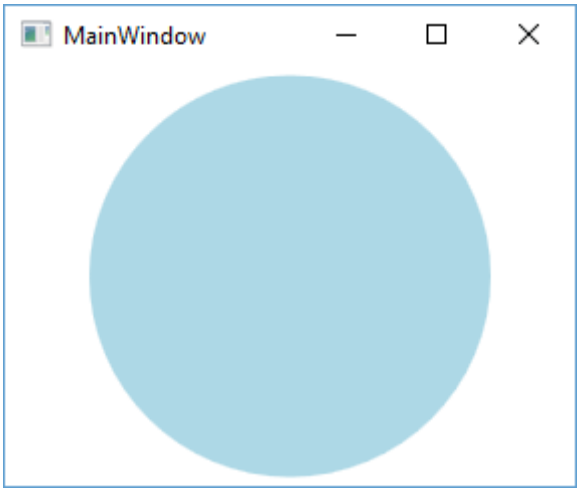
- **Fill** заполняет фон фигуры с помощью кисти - аналогичен свойству **Background** у прочих элементов
- **Stroke** задает кисть, которая отрисовывает границу фигуры - аналогичен свойству **BorderBrush** у прочих элементов
- **StrokeThikness** задает толщину границы фигуры - аналогичен свойству **BorderThikness** у прочих элементов
- **StrokeStartLineCap** и **StrokeEndLineCap** задают для незамкнутых фигур (Line) контур в начале и в конце линии соответственно
- **StrokeDashArray** задает границу фигуры в виде штриховки, создавая эффект пунктира
- **StrokeDashOffset** задает расстояние до начала штриха
- **StrokeDashCap** задает форму штрихов

Ellipse

Ellipse представляет овал:

```
<Ellipse Fill="LightBlue" Width="200" Height="200" />
```

При одинаковой ширине и высоту получается круг:

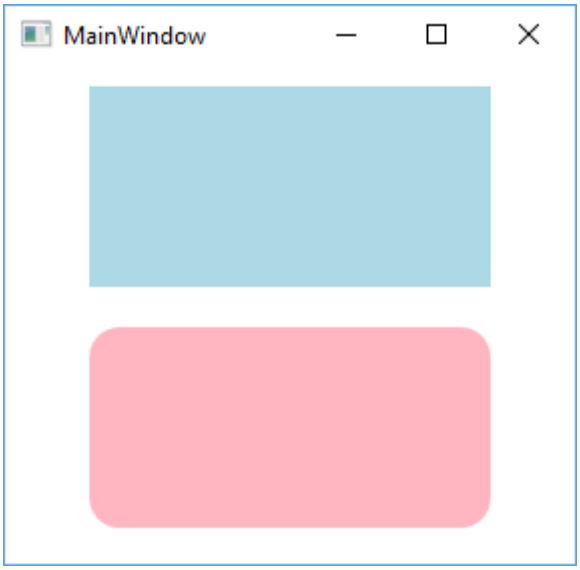


Rectangle

Rectangle представляет прямоугольник::

```
<StackPanel Background="White">
    <Rectangle Fill="LightBlue" Width="200" Height="100" Margin="10" />
    <Rectangle Fill="LightPink" Width="200" Height="100" RadiusX="15" RadiusY="15" Margin="10"
/>
</StackPanel>
```

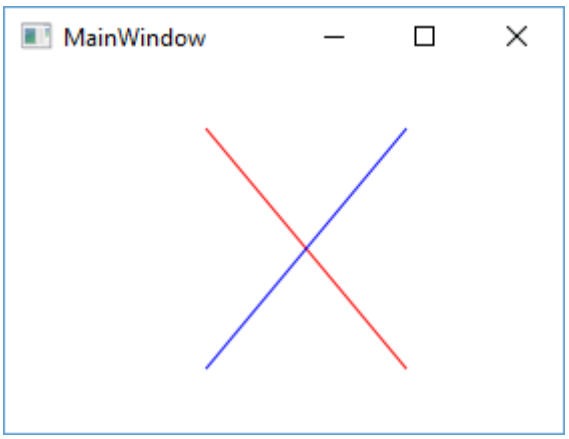
С помощью свойств RadiusX и RadiusY можно округлить углы прямоугольника:



Line

Line представляет простую линию. Для создания линии надо указать координаты в ее свойствах X1, Y1, X2 и Y2. При этом надо учитывать, что началом координатной системы является верхний левый угол:

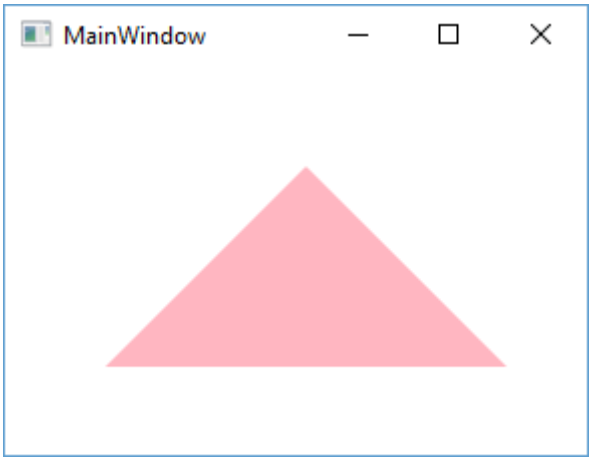
```
<Line X1="100" Y1="30" X2="200" Y2="150" Stroke="Red" />
<Line X1="100" Y1="150" X2="200" Y2="30" Stroke="Blue" />
```



Polygon

Polygon представляет многоугольник. С помощью коллекции Points элемент устанавливает набор точек - объектов типа Point, которые последовательно соединяются линиями, причем последняя точка соединяется с первой:

```
<Polygon Fill="LightPink" Points="50, 150, 150, 50, 250, 150" />
```

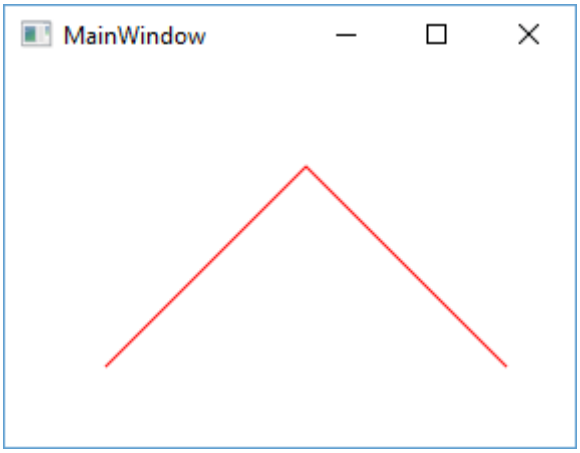


В данном случае у нас три точки (50, 150), (150, 50) и (250, 150), которые образуют треугольник.

Polyline

Polyline представляет набор точек, соединенных линиями. В этом плане данный элемент похож на Polygon за тем исключением, что первая и последняя точка не соединяются:

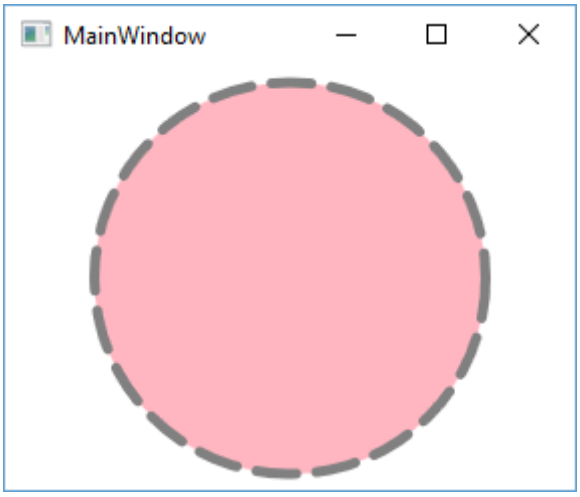
```
<Polyline Stroke="Red" Points="50, 150, 150, 50, 250, 150" />
```



Настройка контура

С помощью ряда свойств мы можем настроить отображение контура. Например:

```
<Ellipse Width="200" Height="200" Fill="LightPink"
        StrokeThickness="5" StrokeDashArray="4 2"
        Stroke="Gray" StrokeDashCap="Round" />
```



Цвет самого контура определяется с помощью свойства `Stroke`, а его толщина - с помощью `StrokeThickness`.

`StrokeDashArray` устанавливает длину штрихов вместе с отступами. Например, `StrokeDashArray="4 2"`устанавливает длину штриха в 4 единицы, а последующего отступа в 2 единицы. И эти значения будут повторяться по всему контуру. При другой установке, например, `StrokeDashArray="1 2 3"` уже задается два штриха. Первый штрих имеет длину в 1 единицу, а второй - в 3 единицы и между ними расстояние в 2 единицы. И так вы можем настроить количество штрихов и расстояний между ними.

`StrokeDashCap` задает форму на концах штрихов и может принимать следующие значения:

- `Flat`: стандартные штрихи с плоскими окончаниями
- `Square`: штрихи с прямоугольными окончаниями
- `Round`: штрихи с округлыми окончаниями
- `Triangle`: штрихи с окончаниями в виде треугольников

Программное рисование

Создание фигур программным образом осуществляется так же, как и создаются и добавляются все остальные элементы:

```
Ellipse el = new Ellipse();
el.Width = 50;
el.Height = 50;
el.VerticalAlignment = VerticalAlignment.Top;
el.Fill = Brushes.Green;
el.Stroke = Brushes.Red;
el.StrokeThickness = 3;
grid1.Children.Add(el);
```

Нарисуем, к примеру, координатную плоскость:

```
Line vertL =new Line();
vertL.X1 = 10;
vertL.Y1 = 150;
vertL.X2 = 10;
vertL.Y2 = 10;
vertL.Stroke = Brushes.Black;
grid1.Children.Add(vertL);
Line horL =new Line();
horL.X1 = 10;
horL.X2 = 150;
horL.Y1 = 150;
horL.Y2 = 150;
horL.Stroke = Brushes.Black;
grid1.Children.Add(horL);
for(byte i = 2;i< 14;i++)
{
    Line a =new Line();
    a.X1 = i * 10;
    a.X2 = i * 10;
    a.Y1 = 155;
    a.Y2 = 145;
    a.Stroke = Brushes.Black;
    grid1.Children.Add(a);
}
for(byte i = 2;i< 14;i++)
{
    Line a =new Line();
    a.X1 = 5;
    a.X2 = 15;
    a.Y1 = i * 10;
    a.Y2 = i * 10;
    a.Stroke = Brushes.Black;
    grid1.Children.Add(a);
}
Polyline vertArr =new Polyline();
vertArr.Points = new PointCollection();
vertArr.Points.Add(new Point(5, 15));
vertArr.Points.Add(new Point(10, 10));
vertArr.Points.Add(new Point(15, 15));
vertArr.Stroke = Brushes.Black;
grid1.Children.Add(vertArr);
Polyline horArr =new Polyline();
horArr.Points = new PointCollection();
horArr.Points.Add(new Point(145, 145));
horArr.Points.Add(new Point(150, 150));
horArr.Points.Add(new Point(145, 155));
horArr.Stroke = Brushes.Black;
grid1.Children.Add(horArr);
```