**Урок №14. Графика в Delphi**

   Работа с графикой в Delphi предполагает обращение к свойству Canvas компонентов. Для программист **Canvas** в Delphi – это холст, который дает доступ к каждому пикселю. Конечно, рисовать попиксельно для работы с графикой в **Delphi** не приходится, система Delphi предоставляет мощные средства работы с графикой, облегчающие задачу программиста.

   В работе с графикой в Delphi, у вас в распоряжении находится канва(холст, а точнее свойство **Canvas** Delphi компонентов), кисть(свойство **Brush**), карандаш(свойство **Pen**) компонента на которым мы собираемся рисовать. У**Pen** и **Brush** есть свойство **Color**(цвет) и **Style**(стиль). Чтоб получить доступ к шрифтам, предоставлено свойство холста(**Canvas**) **Font**.

**Свойства объекта *Pen*(Карандаш)**

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| **Color** | Цвет линии |
| **Width** | Толщина линии(задается в пикселах) |
| **Style** | Вид линии: *psSolid ―* сплошная; *psDash ―* пунктирная, длинные штрихи; *psDot -* пунктирная, короткие штрихи;*psDashDot ―* пунктирная, с чередованием длинного и короткого штрихов; *psDashDotDot* ― пунктирная с чередованием одного длинного и двух коротких штрихов; *psClear* ― линия не отображается(используется если не надо изображать границу области, например прямоугольника) |

**Свойства объекта *Brush*(Кисть)**

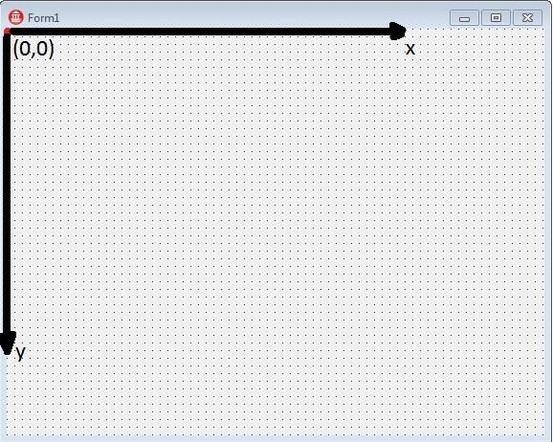
|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| **Color** | Цвет закраски замкнутой области. |
| **Style** | Стиль закраски области:― сплошная заливка; штриховка:*bsHorizontal―*горизонтальная; *bsVertical―*вертикальная;*bsFDiagonal―*диагональная с наклоном линии вперед;*bsBDiagonal* *―*диагональная с наклоном линии назад;*bsCross―*диагональная клетка. |

**Основные свойства класса *TFont***

|  |  |
| --- | --- |
| **Свойство** | **Определяет** |
| **Name** | Шрифт, который используется для отобраения текста. В качестве значения следует брать название шрифта, например *Arial*. |
| **Size** | Размер шрифта |
| **Style** | Стиль начертания символов. Задается с помощью констант:*fsBold*(полужирный), *fsItalic*(курсив), *fsUnderline*(подчеркнутый),*fsStrikeOut*(перечеркнутый).  Свойство **Style** является множеством, что позволяет комбинировать необходимые стили. Например, инструкция, которая устанавливает стиль "полужирный курсив", выглдит так:  **Font.Style** := [fsBold, fsItalic] |
| **Color** | Цвет символов. В качестве значения можно использовать константу типа *TColor*. |

  Конечно, не все компоненты в **Delphi** имеют эти свойства. На вкладке **Additional**расположен специализированный компонент **TImage**, специально предназначенный для рисования, но также свойство **Canvas**имеют, например, такие компоненты как **ListBox**, **ComboBox**, **StringGrid**, а также и сама Форма, которая размещает наши компоненты!

  Холст для рисования **Canvas**, представляет собой перевернутую систему координат XOY, где О эта верхняя левая точка компонента или формы(смотря где мы рисуем).



**Константы *TColor***

|  |  |
| --- | --- |
| **Цвет** | **Константа** |
| Бирюзовый | *clAqua* |
| Черный | *clBlack* |
| Синий | *clBlue* |
| Ярко-розовый | *clFuchsia* |
| Зеленый | *clGreen* |
| Салатовый | *cLime* |
| Каштановый | *clMaroon* |
| Темно-синий | *clNavy* |
| Оливковый | *clOlive* |
| Фиолетовый | *clPurple* |
| Красный | *clRed* |
| Серебристый | *clSilver* |
| Зелено-голубой | *clTeal* |
| Белый | *clWhite* |

  Основное свойство такого объекта как Canvas **Delphi** - Pixels[i, j] типа **TColor**, то есть это двумерный массив точек (пикселов), задаваемых своим цветом.

  Рисование на канве происходит в момент присвоения какой-либо точке канвы заданного цвета. Каждому пикселу может быть присвоен любой доступный для Windows цвет.

*Например:  
  
   Image1.Canvas.Pixels[100, 100] := clRed;*

приведёт к рисованию красной точки с координатами [100, 100]. Узнать цвет пиксела можно обратным присвоением:

*Color := Image1.Canvas.Pixels[100,100];*

  Для стандартных цветов в **Delphi** определён набор текстовых констант. Увидеть его можно, открыв в Инспекторе Объектов свойство Color, например, той же Формы.

**Свойства класса *TCanvas***

|  |  |
| --- | --- |
| *Свойство* | *Описание* |
| ***Pen*** | Карандаш. Определяет цвет, стиль и толщинц линии, которую рисует, например метод *Lineto* |
| ***PenPos*** | Положение(координаты) карандаша |
| ***Brush*** | Кисть. Определяет цвет и стиль закраски области, например прямоугольника, который рисует метод*Rectangle*. |
| ***Font*** | Шрифт. Определяет шрифт, который используется для вывода текста, например методом *TextOut*. |

**Методы класса *TCanvas***

|  |  |
| --- | --- |
| **Метод** | **Действие** |
| ***Lineto****(x, y)* | Рисует линию из текущей точки в точку с указанными координатами (перемещение указателя текущей точки в нужную обеспечивает метод ***Moveto***). Цвет динии определяется свойством *Pen.Color .* |
| ***Rectangle****(x1 ,y1 , x2, y2)* | Рисует прямоугольник. Параметры *x1*, *y1*указывают координаты верхней левой точки, а*x2*, *y2* координаты нижней правой точки. Цвет границы прямоугольника определяет свойство*Pen.Color ,* цвет закраски области *―* свойство*Brush.Color* . |
| ***RoundRect****(x1, y1, x2, y2, x3, y3)* | Рисует прямоугольник со скругленными углами. Параметры *x1*, *y1* указывают координаты верхней левой точки, *x2*, *y2* координаты нижней правой точки, а *x3*, *y3* радиус скругления. Цвет границы прямоугольника определяет свойство *Pen.Color ,*цвет закраски области *―* свойство *Brush.Color* . |
| ***Ellipse****(x1, y1, x2, y2)* | Рисует эллиспс(окружность). Параметры *x1*, *y1*указывают координаты верхней левой точки, а*x2*, *y2* координаты нижней правой точки прямоугольника в который вписана окружность. Цвет границы прямоугольника определяет свойство *Pen.Color ,* цвет закраски области *―*свойство *Brush.Color* . |
| ***Arc****(x1, y1, x2, y2, x3, y3 ,x4, y4)* | Рисует дугу. Параметры *x1*, *y1*, *x2* и *y2* задают эллипс, частью которого является дуга, параметры *x3*, *y3*, *x4* и *y4 ―* начальную и конечную точку дуги. Цвет дуги определяет свойство *Pen.Color*. |
| ***Pie****(x1 ,y1 ,x2 ,y2 ,x3 ,y3 ,x4 ,y4)* | Рисует сектор. Параметры *x1*, *y1*, *x2* и *y2* задают эллипс, частью которого является сектор, параметры *x3*, *y3*, *x4* и *y4 ―*границы сектора. Цвет границы сектора определяет свойство*Pen.Color ,* цвет закраски сектора *―* свойство*Brush.Color* . |
| ***FillRect****(aRect)* | Рисует закрашенный прямоугольник. Параметр*aRect*(тип *TRect*) определяет положение и размер прямоугольника. Цвет закраски области определяет свойство *Brush.Color* . |
| ***FrameRect****(aRect)* | Рисует контур прямоугольника. Параметр*aRect*(тип *TRect*) определяет положение и размер прямоугольника. Цвет контура определяет свойство *Brush.Color* . |
| ***Polyline****(points, n)* | Рисует ломаную линию. *Points ―* массив типа*TPoint*. Каждый элемент массива представляет собой запись, поля *x* и *y* которые содержат координаты точки перегиба ломаной. *n ―*количество звеньев ломаной. Метод*Polyline*вычерчивает ломанную линию, последовательно соединяя прямые точки, координаты которых находятся в массиве: перевую со второй, вторую с третьей, третью с четвертой и т.д. |

**Графические примитивы**

  Картинку, чертеж или схему моно рассматривать, как совокупность графических *примитивов:* точек, линий, окружностей, прямоугольников, дуг, а также букв (текста).

  Вычерчивание графических примитивов на графической поверхности, например компонента *Image*(закладка *Additional*), выполняют соответствующие методы класса **TCanvas**.

  Инструкция, обеспечивающая вычерчивание графического элемента, в общем виде выглядит так:

*Объект*.Canvas.***Метод****(Параметры)*;

*Объект* определяет объект, на поверхности которого нужно нарисовать графический элемент. В качестве объекта можно указать компонент *Image.*

***Метод****–* это имя метода, который обеспечивает рисование нужного графического элемента.

*Параметры,* в большинстве случаев, определяют положение графического элемента на графической поверхности и его размер.

Например в результате выполнения инструкции

*Image1.Canvas.****Rectangle****(10,20,60,40);*

  В поле компонента  *Image1* будет нарисован прямоугольник шириной 50 и высотой 20 пикселов, левый верхний угол которого будет находиться в точке(10,20).

  При записи инструкций, обеспечивающих вывод графики, удобно использовать инструкцию **with***,* которая позволяет сократить количество набираемого кода. Например, вместо:

*Image1.Canvas.Brush.Color := clGreen;*

*Image1.Canvas.Rectangle(20, 20, 46, 70);*

*Image1.Canvas.Brush.Color := clWhite;*

*Image1.Canvas.Rectangle(45, 20, 71, 70);*

*Image1.Canvas.Brush.Color := clRed;*

*Image1.Canvas.Rectangle(70, 20, 96, 70);*

*Image1.Canvas.Brush.Style := bsClear;*

*Image1.Canvas.Font.Name := 'Tahoma';*

*Image1.*Canvas*.Font.Size := 10;*

*x := 20+(75- Image1.Canvas.TextWidth('Италия'))****div***2*;*

*Image1.Canvas.Textout(x,70+Font.Size, 'Италия');*

Можно написать вот так:

**With** Image1.Canvas **do**

**Begin**

  Brush.Color := clGreen;

  Rectangle*(*20, 20, 46, 70*);*

  Brush.Color := clWhite;

  Rectangle(45, 20, 71, 70);

  Brush.Color := clRed;

  Rectangle(70, 20, 96, 70);

  Brush.Style := *bsClear*;

  Font.Name := 'Tahoma';

  Font.Size := 10;

  x := 20+(75- TextWidth('Италия')) div 2;

  Textout(x,70+Font.Size, 'Италия');

**End;**

**Результат выполнения программы:**

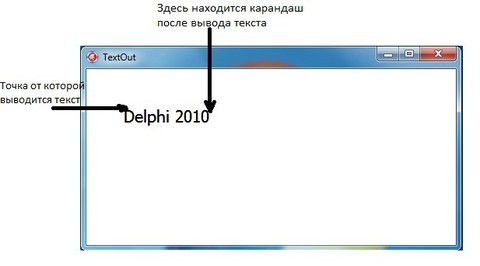


**Текст**

  Вывод строки текста на графическую поверхность объекта обеспечивает метод*TextOut*. Инструкция вызова метода *TextOut* в общем виде выглядит следующим образом:

**Объект**.Canvas.TextOut(x, y, '\*Текст\*');

  Параметры *x, y* определяют координаты точки графической поверхности, от которой выполняется вывод текста.



  Шрифт, используемый для отображения текста, определяет свойство Font графической поверхности, на которую текст выводится. Свойство*Font*представляет собой объект типа **TFont**. В таблице ниже перечисленысвойства класса **TFont**.