**Grafická plocha (Image)**

**Grafická plocha** - je obdĺžnik vo formulári, do ktorého môžeme rôznymi nástrojmi kresliť, jej obsah môžeme jednoducho uložiť na disk, resp. zo súboru do nej prečítať nejaký obrázok.

Vloženie komponentu Image do formulára:

Nájdeme ho v palete komponentov v záložke **Additional** (malý farebný obrázok). Komponent položíme do plochy kliknutím (dostáva rozmer približne 100x100) a môžeme ho zväčšiť ťahaním za modré krúžky.
**Image** môžeme prispôsobiť veľkosti formulára: v inšpektore tomuto komponentu zmeníme jeho nastavenie **Align** na hodnotu **alClient** - toto nastavenie označuje, že grafická plocha pokryje kompletný formulár a bude sa meniť, ak budeme meniť veľkosť formuláru.

Kresliť do grafickej plochy budeme najčastejšie ako akcie pri zatlačení nejakých tlačidiel - teda napr. do procedúry Button1Click budeme zapisovať grafické príkazy medzi **begin** a **end**. Skôr ako začneme kresliť, vysvetlime si základné pojmy:

• naša grafická plocha má meno **Image1** (podobne ako tlačidlo **Button1** a aj formulár **Form1**)
• každá grafická plocha má svoje plátno a kreslíme práve do tohto plátna  - plátno sa nazýva **Canvas**,
• na plátno kreslíme pomocou pera - **Pen** alebo vyfarbujeme nejaké plochy pomocou štetca **Brush**,
• pero aj štetec majú svoje farby - **Color**, pero má aj svoju hrúbku - **Width**,

**Súradnicový systém plochy**

Príkazy, ktoré pohybujú a kreslia perom väčšinou vyžadujú zadávanie súradníc: súradnicová  sústava je vo Windows inak natočená, ako ju poznáme z matematiky

* ľavý horný roh má súradnice **(0,0)** - teda je to počiatok
* **x**-ová súradnica ide zľava doprava na hornom okraji plochy
* **y**-ová súradnica ide zhora nadol pri ľavom okraji plochy

Šírka grafickej plochy je daná vlastnosťou **Image1.Width** a výška je daná vlastnosťou **Image1.Height**. **x**-ové súradnice ľubovoľného bodu sú celočíselné a sú z intervalu **<0, Image1.Width>**. **y**-ové súradnice ľubovoľného bodu sú tiež celočíselné a sú z intervalu **<0, Image1.Height>**.

**Kreslenie do plochy**

Všetko, čo budeme do plochy kresliť, budeme vlastne kresliť na plátno (**Canvas**).

**Vlastnosti plátna - Canvas:**

* **Brush.Color**

Vo Windows sa farba určuje trojicou čísel, ktoré vyjadrujú v akom množstve treba “namiešať” tri základné farby – **červenú**, **zelenú** a **modrú**, aby vznikla žiadaná farba – množstvá týchto zložiek sa zadávajú celým číslom od **0** do **255**, napr. ak namiešame 0 červenej, 0 zelenej a 255 modrej, vznikne jasná modrá farba; 0 červenej, 0 zelenej a 0 modrej je čierna; všetky zložky po 255 znamená bielu; 255 červenej, 255 zelenej a 0 modrej vyrobí žltú a pod.

V Delphi môžeme farby definovať pomocou **funkcie RGB**, ktorá dostáva 3 čísla v desiatkovej sústave od 0 do 255, popisujúce zastúpenie jednotlivých zložiek základných farieb.
Mená niektorých preddefinovaných farieb:

|  |  |
| --- | --- |
| clBlack  clMaroon clGreen          clOlive            clNavy        clPurple        clTeal        clGray        clSilver        clRed        clLime        clYellow        clBlue        clFuchsia        clAqua        clLtGray        clDkGray        clWhite           | RGB(0,0,0)RGB(128,0,0)RGB(0,128,0)RGB(128,128,0)RGB(0,0,128)RGB(128,0,128)RGB(0,128,128)RGB(128,128,128)RGB(192,192,192)RGB(255,0,0)RGB(0,255,0)RGB(255,255,0)RGB(0,0,255)RGB(255,0,255)RGB(0,255,255)RGB(192,192,192)RGB(128,128,128)RGB(255,255,255)  |

* **Brush.Style**
	+ **bsSolid** - plná výplň
	+ **bsCross** - mriežka
	+ **bsClear** - bez výplne
	+ **bsDiagCross** - šikmá mriežka
	+ **bsBDiagonal** - šikmé čiary vpravo
	+ **bsHorizontal** - vodorovné čiary
	+ **bsFDiagonal** - šikmé čiary vľavo
	+ **bsVertical** - zvislé čiary
* **Pen.Color** - ako farba Brushu
* **Pen.Mode**
	+ **pmBlack** - vždy čierne
	+ **pmWhite** - vždy biele
	+ **pmNop** - bezo zmeny
	+ **pmNot** - inverznou farbou vzhľadom na farbu bodu v Canvase
	+ **pmCopy** - farba pera epecifikovaná vo vlastnosti Color
	+ **pmNotCopy**- inverzná farba pera
* **Pen.Style**
	+ **psSolid** - plná čiara
	+ **psDash** - čiarkovaná
	+ **psDot** - bodkovaná
	+ **psDashDot** - bodkočiarkovaná
	+ **psDashDotDot** - bodkobodkočiarkovaná
	+ **psClear** - žiadna čiara
	! Čiarkovaná a bodkovaná čiara sú možné iba pre hrúbku 1!
* **Pen.Width**

**Metódy plátna - Canvas:**

* **FloodFill(X, Y: Integer; Color: TColor; FillStyle: TFillStyle)** - rozleje z bodu (x,y) farbu Brushu po hranicu Color s vyplnením FillStyle
	+ type **TFillStyle** = (**fsSurface**, **fsBorder**);

• **fsSurface** - vyplní plochu, ktorá má farbu parametra Color. Skončí, ak narazí na inú farbu

• **fsBorder** - vyplní oblasť, ktorá neobsahuje farbu parametra Color. Skončí, keď narazí na farbu Color

* **Rectangle(X1, Y1, X2, Y2: Integer)** - obdĺžnik - teda aj štvorec. Štyri čísla v zátvorkách vyjadrujú súradnice nejakých dvoch protiľahlých vrcholov - prvé dve sú x-ová a y-ová súradnice prvého vrcholu a druhé dve opäť x-ová a y-ová súradnice druhého vrcholu
* **Ellipse(X1, Y1, X2, Y2: Integer)** - elipsa a teda aj kruh - parametre sú ako pri obdĺžniku a znamenajú elipsu vpísanú do obdĺžnika
* **LineTo(X, Y: Integer)** - od momentálnej pozície pera kreslí úsečku do bodu (x, y)
* **MoveTo(X, Y: Integer)** - presunie pero bez kreslenia čiar
* **Polygon(Points: array of TPoint)** - napr. Polygon([Point(10, 10), Point(30, 10), Point(130, 30), Point(240, 120)]);

Prvý najjednoduchší príkaz je asi obdĺžnik - zadáme mu dva protiľahlé vrcholy nejakého obdĺžnika a ten ho nakreslí, pričom strany sú rovnobežné s osami. Vysvetlime si príkaz na nakreslenie obdĺžnika:

*procedure TForm1.Button1Click(Sender: TObject);
begin* ***Image1.Canvas.Rectangle****(100, 50, 300, 150);
 end;*

Vidíme, že príkaz sa skladá z viacerých slov oddelených bodkami - poradie týchto slov a samozrejme aj presný zápis je veľmi dôležitý a vyjadruje toto: ideme pracovať s grafickou plochou **Image1** - budeme kresliť na jej plátno **Canvas** - konkrétne použijeme metódu obdĺžnik - **Rectangle**. Štyri čísla v zátvorkách vyjadrujú súradnice nejakých dvoch protiľahlých vrcholov - prvé dve sú x-ová a y-ová súradnice prvého vrcholu a druhé dve opäť x-ová a y-ová súradnice druhého vrcholu. Treba si zapamätať, že tieto súradnice musia byť vždy celé čísla - ak je niektorá z nich mimo veľkosť grafickej plochy, tak zrejme aj nejaká časť obdĺžnika bude mimo plochy. Po spustení programu (**F9**) sa vo formulári sa objaví

Teraz nakreslime ešte jeden obdĺžnik a to tak, aby sa oba navzájom prekrývali, napr. 

*procedure TForm1.Button1Click(Sender: TObject);
begin
  Image1.Canvas.Rectangle(100, 50, 300, 150);
  Image1.Canvas.Rectangle(40, 100, 240, 200);
end;*

Po spustení vidíme, že druhý obdĺžnik je vyplnený bielou farbou, lebo cez neho nie je vidieť ten prvý.
Pri štarte programu má pero vždy čiernu farbu a hrúbku 1, štetec, ktorý vypĺňa vnútro obdĺžnika má pri štarte bielu farbu. Toto všetko sa dá zmeniť špeciálnymi priraďovacími príkazmi: 

*procedure TForm1.Button1Click(Sender: TObject);
begin
  Image1.Canvas.Pen.Color := clRed;
  Image1.Canvas.Brush.Color := clBlue;
  Image1.Canvas.Rectangle(100, 50, 300, 150);
  Image1.Canvas.Pen.Width := 5;
  Image1.Canvas.Brush.Color := clYellow;
  Image1.Canvas.Rectangle(40, 100, 240, 200);
end;*

Vidíme, že prvý obdĺžnik má tenký červený obvod a je vyplnený modrou farbou, druhý obdĺžnik má hrubý červený obvod a je vyfarbený žltou farbou. Farby a hrúbky meníme priraďovacími príkazmi, napr.

*Image1.Canvas.Pen.Color := clRed;*

Znamená, že v grafickej ploche **Image1**, na jeho plátne **Canvas**, zmeníme pre pero **Pen** jeho farbu **Color** na červenú **clRed**. Úplne rovnako je to s farbou štetca a aj hrúbkou pera.
Nasledovný program nakreslí vedľa seba 3 rôzne veľké štvorce:

*procedure TForm1.Button1Click(Sender: TObject);
begin
  Image1.Canvas.Pen.Width := 5;
  Image1.Canvas.Rectangle(100, 200, 150, 150);
  Image1.Canvas.Rectangle(150, 200, 250, 100);
  Image1.Canvas.Rectangle(250, 200, 400, 50);
end;*

Všimnite si, že hrúbku sme nastavili len raz a odvtedy platí pre všetky kresby. Je jasné, že pomocou Rectangle kreslíme aj štvorce. Niekedy sa musíme so súradnicami "trochu pohrať", aby sme dostali útvary podľa našich predstáv - tu sú štvorce nakreslené tesne vedľa seba a ich spodná strana leží na jednej priamke.

**Práca s textom**

**Práca s textom - nastavenia:**

* **Font.Color** - farba písma
* **Font.Height** – výška písma
* **Font.Name** - názov fontu (používa sa aj pre možnosť nastavenia fontu užívateľom)
* **Font.Size** - veľkosť fontu
* **Style** - do hranatých zátvoriek sa vymenujú atribúty písma – **fsBold**(tučné), **fsItalic**(kurzíva), **fsUnderline**(podčiarknuté), **fsStrikeOut**(prečiarknuté)

**Práca s textom - metódy:**

* **TextExtent(const Text: string): TSize**; - funkcia, ktorá vrátia šírku a výšku textu v pixeloch, ak by bol napísaný v aktuálne nastavenom fonte. Funkcia vracia hodnotu typu record s dvoma zložkami cx, cy.
* **TextHeight(const Text: string): Integer;** - funkcia, ktorá vráti výšku textu v pixeloch podľa aktuálne nastaveného fontu
* T**extOut(X, Y: Integer; const Text: string);** - procedúra, ktorá na pozíciu (x,y) vypíše zadaný text
* **TextRect(Rect: TRect; X, Y: Integer; const Text: string);** - vypíše text do zadaného obdĺžnika, prípadný prečnievajúci text oreže
pr. TextRect ( Rect(20, 30, 70, 50), 'To je bomba');
* **TextWidth(const Text: string): Integer;** - funkcia, ktorá vráti šírku textu v pixeloch podľa aktuálne nastaveného fontu

Často sa nám bude hodiť zmazanie grafickej plochy:

*procedure TForm1.Button2Click(Sender: TObject);
begin
  Image1.Canvas.Brush.Color := clWhite;
  Image1.Canvas.FillRect(Image1.ClientRect);
end;*

Prvý riadok, v ktorom nastavujeme bielu farbu štetca, je tu preto, lebo príkaz **FillRect** zmaže celú plochu farbou štetca a najčastejšie sa nám hodí práve biela - samozrejme, že by sme mohli zmazávať ľubovoľnou farbou.

**Úlohy**

1. Vykresli 3 obdĺžniky: prvý tenkým čiernym perom, 2. tenkým červeným perom a 3. hrubým modrým perom a žltou výplňou
2. Nakresli šachovnicu 3 x 3 políčka.
3. Nakresli domček s červenou strechou.
4. Pomocou príkazov MoveTo a LineTo nakreslite rovnostranný trojuholník so stranou a.

*var
  x,y:integer;
begin
  a := 100;
  x:=100; y:=300;
  Image1.Canvas.MoveTo(x,y);
  Image1.Canvas.LineTo(x+a div 2,y - round(a\*0.877));
  Image1.Canvas.LineTo(x+a,y);
  Image1.Canvas.LineTo(x,y);
end;*

1. Nakresli do plochy vyfarbený trojuholník pomocou príkazu polygon.
2. Vypíš do plochy text zadaný užívateľom.

**Náhodné čísla - random**

Budeme riešiť takúto úlohu: treba nakresliť štvorec so stranou **a = 100**, ktorého ľavý horný roh má pri každom stlačení tlačidla **Button1** inú, náhodnú polohu.
Použijeme náhodný generátor - **funkciu Random**, ktorá zakaždým vráti nejakú inú (možno aj tú istú) hodnotu:

*procedure TForm1.Button1Click(Sender: TObject);
var
  A, X, Y: Integer;
begin
  A := 100;
  X := Random(500);
  Y := Random(400);
  Image1.Canvas.Pen.Width := 5;
  Image1.Canvas.Rectangle(X, Y, X+A, Y+A);
end;*

Príkaz **X := Random(500);** znamená, že vždy keď sa ide vykonať, počítač si "vymyslí" nejaké náhodné číslo z intervalu **<0, 499>** a to priradí do premennej X, podobne je to aj s premennou Y - tej sa priradí náhodná hodnota z intervalu **<0, 399>**.
Teda **Random(N)** vráti hodnotu z intervalu **<0, N-1>**. Takáto procedúra po každom zavolaní nakreslí štvorec na iných súradniciach.
Generovanie náhodných čísel je pred prvým použitím príkazu Random inicializovať príkazom **Randomize**.

**Cyklus For, premenná cyklu**

Chceme nakresliť naraz 10 štvorcov na náhodných pozíciách. Všetky príkazy, ktoré kreslia jeden náhodný štvorec, vložíme do konštrukcie tzv. **for-cyklu**:

*procedure TForm1.Button1Click(Sender: TObject);
var
  A, X, Y, I: Integer;
begin
Randomize;
  for I := 1 to 10 do
  begin
    A := 100;
    X := Random(500);
    Y := Random(400);
    Image1.Canvas.Pen.Width := 5;
    Image1.Canvas.Rectangle(X, Y, X+A, Y+A);
  end;
end;*

Veľmi dôležitý je prvý riadok konštrukcie:

* začína slovom **for**, za ktorým nasleduje priradenie počiatočnej hodnoty pomocnej premennej **I** (tzv. **premenná cyklu**) - u nás **1**, a za slovom **to** je číslo **10**, ktoré vyjadruje, že sa budú opakovať nejaké príkazy, pričom premenná cyklu bude postupne nadobúdať hodnoty od 1 do 10 - teda opakovaní bude 10. Za číslom 10 je ešte slovo **do** a slovo **begin**, ktoré označuje, že nasleduje postupnosť príkazov na opakovanie.
* **premenná cyklu** - u nás premenná **I** - musí byť tiež deklarovaná ešte pred prvým begin v časti **var**.

Ďalej nasledujú riadky, ktoré sa budú opakovane vykonávať 10-krát až po slovo **end** - toto je koniec postupnosti opakovaných príkazov.
Premennú cyklu I môžeme veľmi užitočne využívať aj medzi príkazmi, ktoré sa budú opakovať (tzv. **telo cyklu**) - zrejme bude pri každom ďalšom opakovaní mať ďalšiu nasledujúcu hodnotu, teda postupne 1, 2, 3, ... 10. Napr.

*procedure TForm1.Button1Click(Sender: TObject);
var
  A, X, Y, I: Integer;
begin
Randomize;
  for I := 1 to 10 do
  begin
    A := 10 \* I;
    X := Random(500);
    Y := Random(400);
    Image1.Canvas.Pen.Width := 5;
    Image1.Canvas.Rectangle(X, Y, X+A, Y+A);
  end;
end;*

Každý z desiatich štvorcov bude mať veľkosť strany desaťnásobok poradového čísla, t.j. postupne 10, 20, 30, ... 100. Aby sme si lepšie uvedomili, čo všetko a prečo sa v cykle opakuje, ukážme si upravený variant tohto istého programu:

*procedure TForm1.Button1Click(Sender: TObject);
var
  X, Y, I: Integer;
begin
Randomize;
  Image1.Canvas.Pen.Width := 5;
  for I := 1 to 10 do
  begin
    X := Random(500);
    Y := Random(400);
    Image1.Canvas.Rectangle(X, Y, X+10\*I, Y+10\*I);
  end;
end;*

Príkaz, ktorým sa nastavovala hrúbka čiar štvorca na 5 nemusí byť vo vnútri cyklu a tým sa vykoná 10-krát, ale ak bude pred cyklom, vykoná sa len raz. Zrušili sme premennú **A** a tam kde sa používala, sme miesto nej dali vzorec 10\*i. Hoci sme ušetrili 4 bajty na tejto premennej, niekedy ale takéto šetrenie nemusí pomôcť čitateľnosti programu.

Pri používaní cyklu For by sme si mali zapamätať tieto pravidlá:

* **premennú cyklu** môžeme v príkazoch v tele cyklu používať, ale nesmieme ju tu meniť, t.j. **nesmieme do nej nič priraďovať**,
* **premenná cyklu** po skončení cyklu je už voľná na ďalšie použitie - hovoríme, že má nedefinovanú hodnotu, ak ju chceme používať, musíme jej nejakú hodnotu priradiť,
* premenná cyklu postupne nadobúda hodnoty od počiatočnej **A** až po koncovú **B**:
	+ ak je **A <= B**, tak sa cyklus vykoná presne B-A+1 krát,
	+ ak **A = B**, tak sa vykoná práve raz,
	+ ak **A > B**, tak sa nevykoná ani raz
* počiatočná aj koncová hodnota nemusí byť zadaná len konštantou, ale aj **zložitejším aritmetickým výrazom s premennými**

**Úlohy**

1. Vypíšte  modré slová 'I love Delphi' so sivým tieňom. Nezabudnite urobiť priesvitné pozadie písmen. Predĺžte tieň.



*var
  i,x,y:integer;
begin
  x:=10; y:=100;
  g.Brush.Style:=bsClear;
  g.Font.Height:=60;
  g.Font.Name:='Arial';
  g.Font.Style:=[fsBold];
  g.Font.Color:=clLtGray;
  g.TextOut(x+3,y-3,veta);
  for i:=1 to 15 do g.TextOut(x+i,y-2\*i,veta); // vytvorenie tieňa
  g.Font.Color:=clBlue;
  g.TextOut(x,y,`I love Delphi`);
end;*

1. Nakreslite 10 sústredných kruhov s polomermi 10, 20, 30, ... 100. Každý kruh bude vyfarbený náhodnou farbou.
2. Vykreslite na obrazovku 100 kruhov náhodnej farby - obvod aj výplň majú rovnaké, kruhy majú veľkosti 1 až 100.

[http://melisko.webnode.sk/news/graficka-plocha-image-/](http://melisko.webnode.sk/news/graficka-plocha-image-/?utm_source=copy&utm_medium=paste&utm_campaign=copypaste&utm_content=http%3A%2F%2Fmelisko.webnode.sk%2Fnews%2Fgraficka-plocha-image-%2F)

<http://melisko.webnode.sk/informatika-ii/newscbm_895443/10/>