Cvičení 8: Delphi - unita Graph2d, propojení formulářů

1 Opakování - CheckedBox, RadioButton, Edit

- komponenty se nacházejí na liště *Standard*
  - pro použití komponent musíme použít unitu `StdCtrl`

1. **CheckedBox**
   - používá se jako zatrhávací políčko
   - vlastnost `checked`, hodnota *true* nebo *false*
2. **RadioButton**
   - používá se jako zatrhávací políčko pro výběr z několika možností
   - `RadioButton`y se seskupují do `RadioGroup`
   - vlastnost `checked`, hodnota *true* nebo *false*
3. **Edit**
   - používá se jako editační okénko pro vstup nebo výstup
   - hodnota je uchována jako řetězec (`string`)
   - hodnota je uchována ve vlastnosti `Text`
   - pro převod řetězce na číslo se používá funkce `VAL`, naopak `STR`

2 Propojení dvou formulářů

Nejdříve je zapotřebí vytvořit nový formulář.
- v menu vybereme *File - New - Form*

Vytvoří se nám nový formulář a nová unita. Dále je zapotřebí tuto novou unitu uložit.
- v menu vybereme *File - Save*, unitu uložíme. (např. *UnitNew.pas*)

Nyní musíme novou unitu *UnitNew* přidat do našeho programu.
- v menu vybereme *Project - Add To Project...* a vybereme *UnitNew.pas*

Nová unita a s tím i nový formulář, který tato unita obsahuje je již přidán. Pokud však chceme používat nový formulář a funkce nové unity v naší stará unitě, je zapotřebí to ve staré unitě překladači oznámit (načíst novou unitu jako knihovnu). Nejlépe to provést v části implementace.

```
implementation

uses
  UnitNew;
```

Nyní můžeme s novým formulářem a unitou pracovat.
Pozn.: Nový formulář má automaticky nastavenu vlastnost Visible na false. Při spuštění není tedy viditelný.

3 Příklad

Vytvořte aplikaci, která bude obsahovat dva formuláře. Na prvním formuláři bude pouze jedno tlačítko. Po stisku tohoto tlačítka se zobrazí nový formulář a v něm se vykreslí elipsa.

- spustíme Delphi, vytvoří se nám nová aplikace
- uložíme projekt a unitu pomocí *File - Save All.* (*Unit1.pas, Project1.dpr*)
- umístíme na formulář (`Form1`) tlačítko (*button*)
- vytvoříme nový formulář (`Form2`) pomocí *File - New - Form*

- uložíme novou unitu jako *Unit2.pas* pomocí *File - Save*
- přidáme novou unitu do projektu (*Project - Add To Project - Unit2.pas*)
- umístíme na nový formulář (`Form2`) obrázek (*image*) z lišty *Additional*
- vrátíme se k formuláři (`Form1`). Pomocí dvojkliku na tlačítko vytvoříme novou událost na stisk tlačítka
- do těla procedury napíšeme:

```
Form2.Visible := True;
Form2.Image1.Canvas.Ellipse(0,0,80,50);
```

- stiskneme *F9* pro spuštění programu
- program se zeptá, jestli chceme přidat unitu obsahující `Form2` jako novou knihovnu. Potvrdíme *OK*.
- za příkaz `implemantation` se nám automaticky vygeneroval kód: `uses Unit2;`
- program znovu spustíme a už nám funguje

4 Přidání unity Graph2d.pas

- nakopírujte soubory *Graph2d.pas* a *Graph2d.dfm* do adresáře, kde máme uložen náš program
- pomocí *Project - Add To Project* přidáme novou unitu *Graph2d.pas* do našeho projektu
- v oblasti implementace, nejlépe hned za příkazem `implementation` přiřadíme novou knihovnu: `uses Graph2d;`
- nový formulář se jmenuje `Draw2d` a k jednotlivým funkcím tohoto formuláře se přistupuje pomocí tečky. Nový formulář nám vytvořil nový objekt, který obsahuje několik vlastností a metod, ke kterým můžeme přistupovat.
- příklad nastavení měřítka: `Draw2d.Scale(-10,10,-10,10);`

5 Důležité procedury unity Graph2d.pas

- Nebudeme se zabývat všemi procedurami a vlastnostmi, které nám unita *Graph2d.pas* nabízí. Při otevření samotné unity zjistíme, že má podrobně psané poznámky, a tudíž podrobný popis není potřeba.
- V unitě je definován nový tip proměnné `TPoint`. `TPoint` nám určuje bod v uživatelských souřadnicích, jako dvě reálné hodnoty. Pokud bude chtít mít bod `A` o souřadnicích [1.2; 8.6], zadáme jej takto:
  - Deklarace proměnné: `var A: TPoint;`
  - Načtení hodnot: `A[1] := 1.2; A[2] := 8.6;`
- Dále se věnujme vlastnostem:
  - `Unit_X, Unit_Y: Double;` – udává kolik pixelů obsahuje jednotka našeho měřítka. Nastavuje se při nastavení měřítka `Scale`.
  - `O: TPoint;` - světové souřadnice uživatelského počátku.
- Důležité procedury a funkce:
  - `procedure InitImage(Width,Height:Integer);`
    Umožňuje nastavení velikosti kreslicího plátna. Velikost je udávána v pixelech. Dále tato procedura nastaví hloubku obrazu, která je nezbytná pro pozdější vykreslení útvarů. **Tuto proceduru je potřeba vždy použít.**
  - `procedure ClearImage(Red,Green,Blue:byte);`
    Vymaže kreslící plátno, resp. vyplní každý bod plátna barvou zadanou pomocí složek `Red,Green,Blue`, kde jejich hodnoty jsou z intervalu $\langle 0; 255 \rangle$.

- `procedure Scale(x1,x2,y1,y2:double);`
  Důležitá procedura, která umožňuje nastavení uživatelských rozměrů kreslící plochy. Pokud budeme chtít kreslit na plátno s x-ovou osou v rozmezí $\langle -20; 30\rangle$ a y-ovou osou v rozmezí $\langle -10; 40\rangle$, spustíme proceduru `Scale(-20,30,-10,40);`.
- `procedure XAxis(x1,x2,y:double;Red,Green,Blue:byte);`
  `procedure YAxis(y1,y2,x:double;Red,Green,Blue:byte);`
  Sestrojí rovnoběžku s osou x, resp. y v barvě `Red,Green,Blue`. Vhodné pro vykreslení souřadných os. Například červenou x-ovou osu vykreslíme takto: `XAxis(-20,30,0,255,0,0);`
- `procedure XScale(x1,x2,y:double;Red,Green,Blue:byte);`
  `procedure YScale(y1,y2,x:double;Red,Green,Blue:byte);`
  Stejné jako předchozí, pouze s tím rozdílem, že osa bude cejchovaná.
- `procedure PutPoint(X:TPoint;Red,Green,Blue:byte);`
  Vykreslí na plátno bod `X` barvou specifikovanou pomocí složek `Red,Green,Blue`.
- `procedure GetPoint(X:TPoint;var Red,Green,Blue:byte);`
  Vrátí barvu bodu `X` pomocí složek `Red,Green,Blue`.
- `procedure Cross(X:TPoint;Size,Red,Green,Blue:byte);`
  Vykreslí křížek v bodě `X` barvou `Red,Green,Blue`, jehož jednotlivá ramena budou mít velikost `Size` pixelů.
- `procedure Line(X,Y:TPoint;Red,Green,Blue:byte);`
  Vykreslí úsečku mezi body `X,Y` barvou `Red,Green,Blue`.
- `procedure Triangle(A,B,C:TPoint;Red,Green,Blue:byte);`
  Vykreslí trojúhelník zadaný body `A,B,C` barvou `Red,Green,Blue`.
- `procedure FillTriangle(A,B,C:TPoint;Red,Green,Blue:byte);`
  Sestrojí trojúhelník a ten pak vyplní danou barvou.
- `procedure Rectangle(A,B,C,D:TPoint;Red,Green,Blue:byte);`
  Vykreslí obdélník mezi body `A,B,C,D`.

- Unita *Graph2D.pas* obsahuje ještě mnoho dalších procedur, hlavně procedur sloužících k transformacím jednotlivých útvarů. Prostudování těchto procedur necháme na samotném čtenáři.

6 Sestrojení úsečky pomocí unity Graph2D.pas

- Vytvoříme si novou aplikaci a na hlavní formulář přidáme tlačítko.
- Přidáme k naší aplikaci unitu *Graph2D.pas*, jak bylo popsáno výše.
- Nyní pomocí dvojkliku na tlačítko vytvoříme reakci na klik na tlačítko.
- Do vzniklé procedury napíšeme tento zdrojový text:

```
procedure TForm1.Button1Click(Sender: TObject);
var A,B :TPoint;
begin
  A[1] := -8; A[2] := -4;
  B[1] :=  5; B[2] :=  7;
  Draw2D.Visible := True;
  Draw2D.InitImage(500,500);
  Draw2D.Scale(-10,10,-10,10);
  Draw2D.XScale(-10,10,0,0,0,255);
  Draw2D.YScale(-10,10,0,0,0,255);
  Draw2D.Line(A,B,255,0,0);
end;
```

7 Samostatný úkol

Seznamte se podrobněji s unitou *Graph2D.pas* a vyzkoušejte si další procedury, které tato unita nabízí.