Delphi – lekce 1

Prostředí Delphi

Delphi je souhrnem několika dílčích prvků, které se navzájem doplňují a to jsou:

- kompilátor
- linker
- debugger
- IDE
- knihovna komponent VCL (Visual Component Library)
- běhová knihovna RTL (Run Time Library)

Aby bylo jasné, co je cílem každé součásti malé shrnutí. Programátor napíše program (většinou) v **IDE**, kde s jeho pomocí i navrhne i uživatelské rozhraní (formuláře) přičemž jednotlivé prvky jsou tvořeny komponentami **VCL**. Programátor pak za pomoci jazyka Object Pascal a **RTL** vytváří reakce na uživatelské vstupy (např. kliknutí na tlačítko, vstup textu atd). Program je uložen v souborech .pas, popis formuláře je uložen v souborech dfm. Formulář je tedy v základě tvořen dvojicí pas a dfm.

Často je výhodné opakující se kód separovat do zvláštních jednotek. Tento kód je možno následně používat z různých míst programu (tj. např. různých formulářů).

Kompilátor slouží k překladu uvedených zdrojových textů (.pas) do předkompilovaných souborů (.dcu). Zároveň je jeho cílem kontrola správnosti zápisu kódu a jiné věci. Pokud proběhla fáze kompilace – tj. všechny potřebné pas soubory jsou přeloženy do dcu (tady jen upozorním, že se překládají jen změněné pas soubory, případně soubory, které nějak závisí na změněných souborech, v ostatních případech se použijí již přeložené dcu soubory) zavolá se **linker**. Jeho cílem je spojit přeložené dcu soubory, připojit k nim popisy formuláře (dfm) a další soubory do výsledného EXE.

Zároveň linker (pokud je zapnuta optimalizace) vyhazuje nepotřebný kód, včetně např. celých bloků kódu. Kód zahrnutý do výsledného binarního souboru je po kompilaci (tím budu označovat kompilaci+linkování) označen tečkou vlevo na řádku (v Delphi 2007 modrou – viz obrázek).

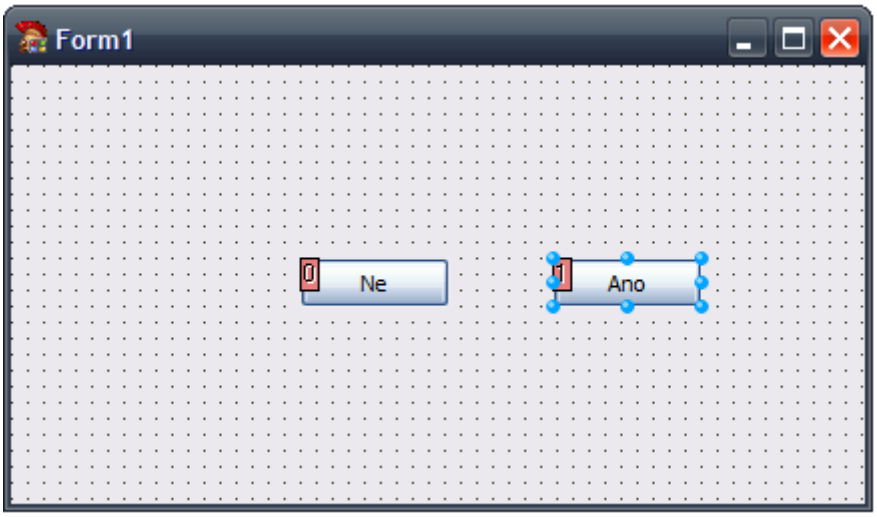
Tyto řádky také označují místo, kde se dá nastavit **breakpoint** pro **debugger** (nastavení se provede kliknutím na modrou tečku). Debugger slouží k ladění programu a breakpoint označuje místo, kde se má program zastavit. V okamžiku zastavení se pak dají zkoumat proměnné atd, a pak pokračovat v běhu (viz. hlavní menu Run). Na obrázku je nastaven breakpoint na řádek 41.

```
29      procedure TForm1.btn1Click(Sender: TObject);
30    30 begin
31       |  ShowMessage('NE');
32       end;
33
34      procedure TForm1.btn2Click(Sender: TObject);
35      begin
36       |  ShowMessage('ANO');
37       end;
38
39      procedure TForm1.btn2MouseEnter(Sender: TObject);
40    40 begin
41    41   if btn2.Left > 200 then
42       |    btn2.Left := 22
43       |  else
44       |    btn2.Left := 270;
45       |
46       end;
```

Programování v Delphi je *událostmi řízené programování*. To znamená, že většinou reagujete na nějakou událost (např. kliknutí myší uživatele). Každá komponenta VCL má několik událostí, které se liší dle komponenty (např. tlačítko má OnClick, tedy událost, která se vyvolá kliknutím na tlačítko).

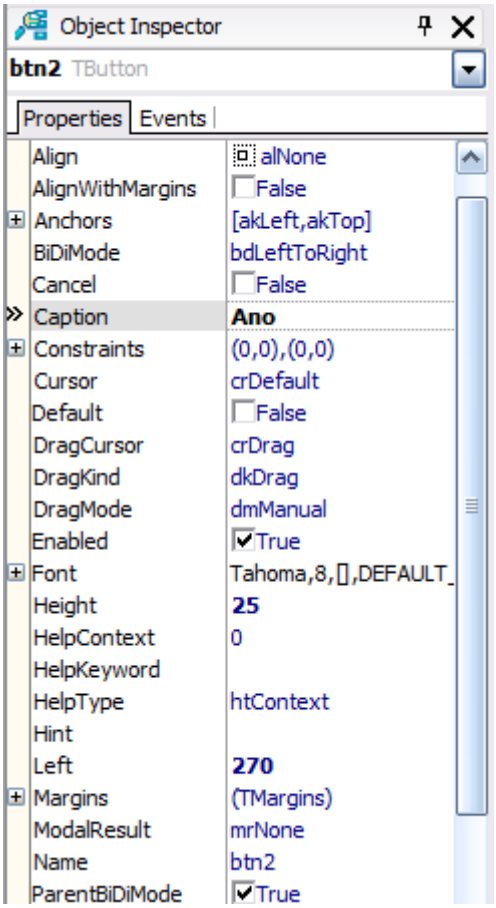
Kromě událostí má každá komponenta i vlastnosti (property), které specifikují její chování. Např. tlačítko má property Caption, která určuje popisku tlačítka, nebo Left a Right (pozice vpravo a vlevo).

Udělejme malý příklad. Mějme formulář se dvěma tlačítky (btn1, btn2 – property Name). btn2.Caption mějme ‘ANO’, btn1.Caption zase ‘NE’. Formulář v IDE bude vypadat nějak takto.



Mezi kódem pro formulář a uvedeným zobrazením formuláře se přepínáme pomocí klávesy F12.

Máme tedy zobrazen formulář a nyní klikněte jednou na tlačítko ANO a v Object Inspectoru se vybere tlačítko btn2.



Object Inspector slouží k editaci komponent. Nyní můžeme editovat vlastnosti (záložka Properties) a události (záložka Events). Tučně jsou zobrazeny změněné položky. Kliknutím na záložku Events vybereme události a dvojklikem na OnClick vygenerujeme kostru obsluhy kliknutí na tlačítko.

Mezi begin a end napíšeme

```
  ShowMessage('ANO');
```

takže kód pak vypadá jak na prvním obrázku. Tento řádek zobrazí hlášku ANO.

Z menu Run vybereme položku Run (zapamatujeme si klávesovou zkratku F9) a program spustíme. Po kliknutí na tlačítko Ano se vypíše ANO. Program ukončíme, popř. z IDE můžeme použít klávesovou zkratku CTRL+F2 (viz menu Run).

Podobně obsloužíme druhé tlačítko. Zatím je program celkem nuda, takže zkusíme něco lepšího.

Pro tlačítko btn2 vygenerujeme obsluhu pro **OnMouseEnter**. Tato událost se vyvolá když tlačítko je zaměřeno myší. Do obsluhy napíšeme kód dle prvního obrázku:

```
if btn2.Left > 200 then
  btn2.Left := 22
else
  btn2.Left := 270;
```

Teď aby program dobře fungoval, je ještě třeba nastavit pozici tlačítka btn1.Left na 144 a btn2.Left na 270. To lze jednak v Object Inspectoru, druhak přímo přetažením patřičných komponent (při pohybu se aktualizují vlastnosti určující souřadnice tlačítka).

Jakmile program spustíme, nastavíme breakpoint dle obrázku a uvidíme, že se nám program zastavuje, kde potřebujeme, tj. při zaměření tlačítka myší.

Program je přiložen. Mimochodem myší nedostižné tlačítko se dá stisknout pomocí výběru přes TAB a potvrzení mezerníkem. Tomu se budeme věnovat někdy příště.

Domácí úkol:

Nejdříve jsem chtěl, aby byl cílem program, kde tlačítko bude kopírovat pozici myši na formuláři, ale zkusíme něco jiného.

Mějme na formuláři tlačítko, které se po kliknutí posune doprava o 20 pixelů, ale tak, že kdyby se po přesunu dotknulo pravého okraje formuláře tak se místo toho nastaví na pozici 5 formuláře.