# Delphi – lekce 7

**Minimum z Object Pascalu (část 2)**

**Řízení toku programu**

Programátor musí být schopen nějak ovlivňovat běh programu a k tomu má několik možností:

- cykly
- větvení
- volání podprogramů

## Větvení

V podstatě máme dvě možnosti, příkaz **IF** a **CASE**.

```
if boolean-výraz then
   příkaz1
else
  příkaz2;
```

Kde oba příkazy mohou být složené příkazy (tj. posloupnost begin, end) a část else je nepovinná.

```
case ordinální-typ of
polozka1: příkaz1;
polozka2, polozka3: příkaz2;
polozka4..polozka10: příkaz3;
else
  příkaz4;
end;
```

Všimněte si, že parametrem case může být i interval. Ohledně ordinálního typu viz minulá lekce.

## Cykly

Máme tři možnosti: While, Repeat – until a For.

**Repeat**

syntaxe: repeat příkaz1; ...; příkazn; until výraz;

Pokud výraz vrátí True je cyklus ukončen. Důležité je, že test je proveden **až po**, tj. nejmeně jedna iterace cyklu je provedena.

```
repeat
  Write('Vložte hodnotu (0..9): ');
  Readln(I);
until (I >= 0) and (I <= 9);
```

**While**

syntaxe: while výraz do příkaz(y)

```
while not ds.Eof do
begin
   // zpracování řádku
   ds.Next;
end;
```

Hlavní rozdíl proti repeat-until je provedení testu **před** provedením prvního příkazu, tj. nemusí být proveden ani jedna iterace.

**For**

Tento cyklus (v základní variantě) potřebuje přesný počet iterací.
for počitadlo := hodnotaod to hodnotado do příkaz (např. 1 to 10)
nebo sestupně
for počitadlo := hodnotaod downto hodnotado do příkaz (např. 10 downto 1)

Počitadlo je proměnná ordinálního typu (tj. i například výčet nebo char). Hodnotado je vypočtena (v případě výrazu jen jednou před začátkem provádění).

```
for I := ListBox1.Items.Count - 1 downto 0 do
    ListBox1.Items[I] := UpperCase(ListBox1.Items[I]);
```

**For – in**

Novější Delphi (myslím Delphi 2005+) umí používat cyklus **for** jako iteraci přes jiný typ (např. string, set, pole nebo kolekci jako je následník TList).

```
var
  C: Char;
  S1, S2: String;

begin
  S1 := 'řetězec';

  for C in S1 do
    S2 := S2 + C;
```

```
end;
```

## Volání podprogramů

Podprogramy (procedury a funkce) slouží ke zlepšení čitelnosti kódu, opakované využití kódu atd. Obecně se snažíme udržovat jednotlivé části programu do délky jedné stránky. Pokud kód překročí stránku zdrojového kódu, mělo by se silně uvažovat o rozdělení do procedur.

Procedura i funkce může mít parametry, funkce navíc i návratovou hodnotu. Parametry mohou být prakticky libovolného typu (včetně složitých datových typů jako je record nebo array).

deklarace:

procedure Jmeno (par1, par2….);

function Jmeno2(par1, par2….):typ;

kde par1 je specifikace předávaného parametru (promenna:typ), např. s:string s případnou specifikací jak se má kompilátor chovat k předané proměnné uvnitř podprogramu.

Př.:

```
procedure Test(const s1:string; var i2:Integer; s3: string);
begin
 // s1 – se nedá měnit a navíc const napovídá kompilátoru, že může dobře optimalizovat
// i2 – proměnná se dá měnit a její hodnota při opuštění podprogramu se přenese do volaného kódu
// s3 – proměnná se dá měnit, ale její hodnota je na konci ztracena
end;
```

Parametr i2 je předáván **odkazem**, s1 a s3 **hodnotou**.

U funkce je návratová hodnota reprezentována proměnnou Result nebo parametrem exit.

```
function Sum(i1, i2: Integer):Integer;
begin
  Result := i1 + i2;
// nebo exit i1 + i2; // nejsem si jist zda od Delphi 2007 nebo až od Delphi 2009
end;
```

### Metody

Pokud je procedura nebo funkce definována na třídě tak se jedná o **metodu**. Metody jsem rozebral na delphi.cz (http://delphi.cz/post/Object-Pascal-zacatecnici.aspx).

## Nejčastější procedury a funkce z RTL (sysutils)

| | |
|---|---|
| Abort | Tichá výjimka, ukončení provádění aktuální akce |
| CreateDir | `CreateDir('C:\test'); vytvoří adresář` |
| `CreateGuid` | ID:GUID;<br><br>`if CreateGuid(ID) = S_OK then`<br>`  Edit1.Text := GUIDToString(ID);` |
| CurrToStr,<br>FloatToStr | Konverze currency na string a float na string |
| Date | Datum |
| DateTimeToStr,<br>DateTimeToString,<br>DateToStr | Převod TDateTime na string |
| DayOfWeek | Den v týdnu (pozor: neděle je první) |
| DecodeDate,<br>DecodeTime | TDateTime na složky |
| DeleteFile | Smazání souboru |
| DirectoryExists | Existuje adresář? |
| EncodeDate,<br>EncodeTime | složky na TDateTime |
| ExtractFileExt,<br>ExtractFileName,<br>ExtractFilePath | Práce s názvem souboru (vrátí příslušnou část) |
| FileExists | Existence souboru |
| FindFirst,<br>FindNext,<br>FindClose | Hledání souboru, v Delphi 2010 lépe přes objektový přístup |
| ForceDirectories | Vytvoří celou adresářovou cestu |
| **Format** | Efektivní formátování řetězců<br>Format('Cislo: %d', [i]); |

| | |
|---|---|
| **FreeAndNil** | Uvolnění objektu a nastavení na nil |
| IntToHex, **IntToStr** | Integer na string, resp. string v hexa |
| IncMonth | Zvyš měsíc v datumu |
| LowerCase, AnsiLowerCase | string na malé písmena, Ansi – včetně češtiny |
| **Now** | Aktuální čas |
| QuotedStr | vrací zadaný řetezec v uvozovkách, třeba pro SQL |
| StrToInt, **StrToIntDef** | String na integer, *Def s def. hodnotou (zamezí výjimce) |
| Trim, TrimLeft, TrimRight | Odříznutí mezer (a spol.) z řetezce |
| ChDir | Změna adresáře |
| MaxInt, MinInt | rozsah integeru |
| MkDir | vytvoř adresář |
| **Pos** | existuje část řetězce v řetězci?<br>if Pos(' ', S) > 0 then |
| Random | Vrátí náhodné číslo |
| StringOfChar | Vytvoří řetězec ze znaku a jeho četnosti (StringOfChar('A', 10); => 'AAAAAAAAAAA' |
| **Copy** | vrátí část řetězce<br>s := 'textova polozka';<br><br>s2 := copy(s, 2, 4); // vrati xtov |

příklad:

- na formulář vložte 3x TEdit a tlačítko
- po stisku tlačítka:
    - se zjistí aktuální datum, převede se na řetezec (jen datum) a první 3 znaky naplní první edit

- druhý edit = hexa hodnota maximalní hodnoty typu integer zjištěná programově

- získá se text ze třetího editboxu a zkusí se převést na číslo.
  pokud je číslo menší jak 10 vypíše „hodnota je v menší než 10 (xx)“
  pokud je číslo v intervalu 10 – 100 vypíše „hodnota je v intervalu 10 - 100 (xx)“
  >100 - vypíše „hodnota je větší jak 100 (xx)“
  neplatná hodnota > neplatná hodnota
  Výpis přes ShowMessage, místo xx zadaná hodnota
  (použít case a StrToInt resp. StrToIntDef, případně obsluhu vyjímek)
  Otestovat pro tyto texty: 1, 10, 99, 100, -1, te02