

29. Neues zum Thema „Prozeduren“ und „Variable“

Du benutzt hoffentlich schon lange eigene Prozeduren und ganz sicher Variable. Nun wird es Zeit, ein paar Feinheiten nachzuliefern. Das mag anfangs Kopfschmerzen bereiten, ist mittelfristig aber easy.

a) lokale und globale Prozeduren

Eine Prozedur kann wiederum eigene Prozeduren enthalten, die einfach am Prozeduranfang (vor dem BEGIN) notiert werden - ohne vorangestellten Formularnamen (denn diese Unterprozedur gehört automatisch zu ihrer Prozedur und die gehört bereits zu einem Formular) und Vorankündigung am Unitanfang. Diese *lokalen* Prozeduren können aber nur von der Prozedur, in der sie deklariert sind, aufgerufen werden.

b) lokale, globale und publice Variable

Weil in jeder Unit, in jeder Prozedur und in jeder Unterprozedur eine eigene Variablendeklaration erfolgen kann, stellt sich die Frage nach dem *Gültigkeitsbereich* dieser Variablen.

- Soll eine Unit-Variable auch in anderen Units benutzt werden, muss sie (ohne vorangestelltes VAR) unter „public“ deklariert werden. Sie wird aus anderen Formularen mit vorangestelltem Namen „ihres“ Formulars angesprochen, z.B. Form1.thegreatbolz := 15;
- Variablen, die wie bisher üblich (hinter VAR) am Anfang einer Unit deklariert werden, können überall in der Unit benutzt werden, nicht aber in anderen Units. Man nennt sie global.
- Die in einer Prozedur (zwischen Prozedurkopfzeile und dem BEGIN der Prozedur, hinter VAR) deklarierten Variablen gelten nur in dieser und den untergeordneten Prozeduren. Diese lokalen Variablen können in keiner anderen Prozedur benutzt werden und erst recht nicht in anderen Units. Der Speicherplatz dieser lokalen Variablen wird beim Verlassen der Prozedur freigegeben, der Variablenwert ist also beim erneuten Prozeduraufruf i.A. nicht mehr verfügbar!
- Wird unter dem Namen einer globalen Variablen zusätzlich innerhalb einer Prozedur eine lokale Variable deklariert, so wird mit dem Variablennamen immer die lokale („näherliegende“) Variable angesprochen. Der Wert der globalen Variablen dieses Namens ist also für die Dauer der Bearbeitung der Prozedur „verdeckt“. Dieses Kuddelmuddel solltest du anfangs aber lieber vermeiden...

c) spezielle Variable: Parameter

Um eine Prozedur mit unterschiedlichen Eingabewerten aufrufen zu können, kann an den Prozedurnamen eine *Parameterliste* angehängt werden. Das kennst du bereits von den Grafikbefehlen. Diese *Parameter* dienen als Platzhalter für die Werte, die beim Aufruf der Prozedur mitgegeben werden. Sie können innerhalb der Prozedur wie lokale Variable benutzt werden, haben aber beim Prozedurstart bereits den übergebenen Wert.

Beispiel:

```
PROCEDURE TForm1.Nonsense;  
VAR a, b : INTEGER;  
    PROCEDURE kunst (howmany: INTEGER);  
        VAR count : INTEGER;  
        BEGIN  
            howmany := howmany + 25;  
            FOR count := 1 TO howmany DO  
                LineTo (Random(600),Random(300));  
            END;  
    BEGIN  
        MoveTo (1,1);  
        kunst (123);  
        MoveTo (600,300);  
        kunst (456);  
    END;
```

30. Software-Auslösung von Ereignissen

Manchmal ist es hilfreich, wenn das Delphi-Programm selbst ein Ereignis auslöst. Vom OnClick-Ereignis eines Buttons kennst du das ja schon länger.

Prinzipiell lässt sich *jedes* Ereignis *jeden* Objekts „von Hand“ auslösen. Du musst aber beachten, dass Delphi eine Information über das Ereignis erwartet, die über einen Parameter „Sender“ geliefert wird. Der enthält bei deinem Aufruf zwar keinen sinnvollen Wert, das macht aber nichts.

Die Ereignisse heißen so, wie es im Objektinspektor (linke Spalte) steht. In deinem Programm könnte also z.B. irgendwo stehen:

- x Edit1.OnChange(Sender);
- x Form1.OnPaint (Sender);

womit das OnChange-Ereignis dieses Edits ausgelöst wird,
womit das OnPaint-Ereignis dieses Formulars ausgelöst wird.