Eine Undo-Funktion für Visual Basic - Programme

Praktisch alle professionellen Windows-Anwendungen verfügen über eine Undo-Funktion, die es erlaubt die letzten vom Benutzer durchgeführten Änderungen zurückzunehmen. Dieser Tip zeigt Ihnen, wie Sie als Visual Basic Programmierer ohne großen Aufwand Ihre Applikationen mit diesem Feature ausstatten können.

Fügen Sie Ihrem VB-Projekt ein neues Modul hinzu, in das Sie folgende Deklarationen eintragen:

```
Declare Function SendMessage& Lib "User" (ByVal
hWnd%, ByVal wMsg%, ByVal wParam%, lParam As
Any)
Declare Function GetFocus% Lib "User" ()
Global Const WM_USER = &H400
Global Const EM_UNDO = WM_USER + 23
Global Const EM_CANUNDO = WM_USER + 22
Global Const EM_EMPTYUNDOBUFFER = WM_USER + 29
```

Legen Sie in Ihrer Form ein Menü mit dem Name mnu_undo an und versehen Sie es mit einem Hot-Key (Windows-Standard hierfür ist hier Ctrl+Z), damit der User die Undo-Funktion problemlos aufrufen kann. In die Click-Prozedur tragen Sie folgenden Code ein:

```
Sub mnu_undo_Click ()
If SendMessage&(GetFocus(), EM_CANUNDO, 0, 0) =
0 Then
   MsgBox "Zurücknahme der letzten Änderung
leider nicht möglich.", 64, "UNDO"
   Exit Sub
End If
x& = SendMessage&(GetFocus(), EM_UNDO, 0, 0)
End Sub
```

Die beiden Aufrufe der API-Funktion SendMessage bewirken folgendes:

In der IF-Abfrage wird durch Senden der EM_CANUNDO-Nachricht getestet, ob das Control auch die Undo-Funktion unterstützt und diese im momentanen Zustand überhaupt möglich ist. Wenn die Rücknahme möglich ist, ist der Rückgabewert der Funktion ungleich 0, ist sie nicht möglich wird als Wert 0 zurückgegeben. In dem obigen Beispiel wird dann eine MessageBox gezeigt und die Funktion verlassen.

Wenn ein Undo möglich ist, wird durch Senden der EM_UNDO-Nachricht die zuletzt vorgenomme Änderung im aktuellen Control rückgängig gemacht. Das aktuelle Control wird hierbei durch die API-Funktion GetFocus() ermittelt. Ein erneutes Aufrufen der Funktion durch den User würde den ursprünglichen Zustand wiederherstellen.

Wenn Sie aus irgendwelchen Gründen das Undo eines bestimmten Controls verhindern wollen, können Sie dies durch den folgenden Aufruf tun:

```
x& = SendMessage&(ControlName.hWnd,
EM_EMPTYUNDOBUFFER, 0, 0)
```

Die Nachricht EM_EMPTYUNDOBUFFER löscht den Undo-Puffer des angegeben Controls. Wenn Sie statt ControlName.hwnd wie oben beschrieben die Funktion GetFocus() einsetzen, wird der Puffer des aktuellen Controls gelöscht. In der Change-Prozedur eines Controls plaziert, verhindert dieser Aufruf daß z.B. Passwörter zurückgeholt und mißbraucht werden können.

Der beschriebene Code funktioniert einwandfrei mit den bei Visual Basic mitgelieferten Text- und Comboboxen (Style = 0). Bei einigen von Drittanbietern stammenden Controls kann es zu Schwierigkeiten kommen, was aber mit der oben beschriebenen Nachricht `EM_CANUNDO` problemlos getestet werden kann. Beachten Sie hierbei aber, daß diese Nachricht auch dann einen Fehler zurückmeldet, wenn vom Anwender noch keine Änderungen vorgenommen wurden und aus diesem Grund kein Undo möglich ist.

Klaus Rambow, Egelsbach