How to add text completion capability to a TComboBox

Answer 1

The Netscape Communicator location box, The Windows 98 Run dialog, and other programs, have implemented a very user friendly feature known commonly as text completion. This document describes how to add similar functionality to a *TComboBox*. The most elegant and reusable way to add this functionality is by descending from *TComboBox* and overriding the *ComboWndProc* to handle the *WM_KEYUP* message. By adding a new property *TextCompletion*, the functionality can be toggled to act like a regular *TComboBox*. Below is the component unit that implements text completion in a *TComboBox*. This unit can be installed as is.

```
unit CompletingComboBox;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TCompletingComboBox = class(TComboBox)
  private
    FTextCompletion: Boolean;
    function GetTextCompletion: Boolean;
    procedure SetTextCompletion(const Value: Boolean);
  protected
    {override the WndProc() so that we can trap KeyUp events}
    procedure ComboWndProc(var Message: TMessage; ComboWnd: HWnd;
      ComboProc: Pointer); override;
  public
    {Public declarations}
  published
    property TextCompletion: Boolean
      read GetTextCompletion write SetTextCompletion;
end;

procedure Register;

implementation

procedure Register;
begin
  RegisterComponents('Standard', [TCompletingComboBox]);
end;

{TCompletingComboBox}

function TCompletingComboBox.GetTextCompletion: Boolean;
begin
  Result := fTextCompletion;
end;

procedure TCompletingComboBox.SetTextCompletion(const Value: Boolean);
begin
  fTextCompletion := Value;
end;

procedure TCompletingComboBox.ComboWndProc(var Message:TMessage;
ComboWnd: HWnd; ComboProc: Pointer);
var
  rc, len: Integer;
begin
  inherited;
  case Message.Msg of
```

```
    WM_KEYUP:
    begin
      {test to see if its a character that should not be processed}
      if (Message.WParam <> 8) and (Message.WParam <> VK_DELETE) and
      (Message.WParam <> VK_SHIFT) and (FTextCompletion = True) then
      begin
        {Use CB_FINDSTRING to locate the string in the Items property}
        rc := Perform(CB_FINDSTRING, - 1, Integer(PChar(Caption)));
        {if its in there then add the new string to the Text and select
        the portion that wasn't typed in by the user}
        if rc <> CB_ERR then
        begin
          {store the length of the current string}
          len := Length(Text);
          {set the new string}
          ItemIndex := rc;
          {highlight the rest of the text that was added}
          SelStart := len;
          SelLength := Length(Text) - len;
          {return 0 to signify that the message has been handled}
          Message.Result := 0;
        end;
      end;
    end;
  end;
end;

end.
```

Answer 2

Performing autocompletion in a combobox:

```
procedure TForm1.ComboBox1Change(Sender: TObject);
var
  oldpos: Integer;
  item: Integer;
begin
  with Sender as TComboBox do
  begin
    oldpos := selstart;
    item := Perform( CB_FINDSTRING, - 1, lparam( Pchar( text )));
    if item >= 0 then
    begin
      onchange := nil;
      text := items[item];
      selstart := oldpos;
      sellength := gettextlen - selstart;
      onchange := combobox1change;
    end;
  end;
end;

procedure TForm1.ComboBox1KeyPress(Sender: TObject; var Key: Char);
var
  oldlen: Integer;
begin
  if key = #8 then
    with sender as TComboBox do
    begin
      oldlen := sellength;
      if selstart > 0 then
      begin
        selstart := selstart - 1;
        sellength := oldlen + 1;
      end;
    end;
end;
```

Tip by Peter Below

Original resource:	The Delphi Pool
Author:	Anon & Peter Below
Added:	2009-10-26
Last updated:	2009-10-26