How to create a transparent TMemo

```
unit TrMemo;

{$R-}

interface

uses
  Messages, Controls, StdCtrls, classes;

const
  TMWM_SpecialInvalidate = WM_USER + 1111;

type
  TTransparentMemo = class(TMemo)
  private
    procedure SpecialInvalidate(var Message: TMessage);
      message TMWM_SpecialInvalidate;
    procedure WMHScroll(var Message: TWMHScroll); message WM_HSCROLL;
    procedure WMVScroll(var Message: TWMVScroll); message WM_VSCROLL;
    procedure WMSetText(var Message: TWMSetText); message WM_SETTEXT;
    procedure CNCTLCOLOREDIT(var Message: TWMCTLCOLOREDIT);
      message CN_CTLCOLOREDIT;
    procedure WMKeyDown(var Message: TWMKeyDown); message WM_KEYDOWN;
    procedure WMEraseBkgnd(var Message: TWMEraseBkgnd); message WM_ERASEBKGND;
  protected
    procedure CreateParams(var Params: TCreateParams); override;
  public
    constructor Create(AOwner: TComponent); override;
  end;

procedure Register;

implementation

uses
  Windows;

{ TTransparentMemo }

procedure TTransparentMemo.WMHScroll(var Message: TWMHScroll);
begin
  inherited;
  PostMessage(Handle, TMWM_SpecialInvalidate, 0, 0);
end;

procedure TTransparentMemo.WMVScroll(var Message: TWMVScroll);
begin
  SendMessage(Handle, TMWM_SpecialInvalidate, 0, 0);
  inherited;
  PostMessage(Handle, TMWM_SpecialInvalidate, 0, 0);
end;

procedure TTransparentMemo.CNCTLCOLOREDIT(var Message: TWMCTLCOLOREDIT);
begin
  with Message do
  begin
    SetBkMode(ChildDC, TRANSPARENT);
    Result := GetStockObject(HOLLOW_BRUSH);
  end;
end;

procedure TTransparentMemo.WMSetText(var Message: TWMSetText);
begin
  inherited;
  if not (csDesigning in ComponentState) then
    PostMessage(Handle, TMWM_SpecialInvalidate, 0, 0)
end;

procedure TTransparentMemo.SpecialInvalidate(var Message: TMessage);
```

```
var
  r: TRect;
begin
  if Parent < > nil then
  begin
    r := ClientRect;
    r.TopLeft := Parent.ScreenToClient(ClientToScreen(r.TopLeft));
    r.BottomRight := Parent.ScreenToClient(ClientToScreen(r.BottomRight));
    InvalidateRect(Parent.Handle, @r, true);
    RedrawWindow(Handle, nil, 0, RDW_FRAME + RDW_INVALIDATE)
  end;
end;

procedure TTransparentMemo.WMKeyDown(var Message: TWMKeyDown);
begin
  SendMessage(Handle, TMWM_SpecialInvalidate, 0, 0);
  inherited;
  PostMessage(Handle, TMWM_SpecialInvalidate, 0, 0);
end;

procedure TTransparentMemo.WMEraseBkgnd(var Message: TWMEraseBkgnd);
begin
  Message.Result := 1
end;

constructor TTransparentMemo.Create(AOwner: TComponent);
begin
  inherited;
  ControlStyle := [csCaptureMouse, csDesignInteractive, csClickEvents,
    csSetCaption, csOpaque, csDoubleClicks, csReplicatable, csNoStdEvents];
end;

procedure TTransparentMemo.CreateParams(var Params: TCreateParams);
begin
  inherited CreateParams(Params);
  with Params do
  begin
   ExStyle := ExStyle or WS_EX_TRANSPARENT and not WS_EX_WINDOWEDGE
    and not WS_EX_STATICEDGE and not WS_EX_DLGMODALFRAME and not
    WS_EX_CLIENTEDGE;
  end;
end;

procedure Register;
begin
  RegisterComponents('cool!', [tTransparentMemo]);
end;

end.
```

Original resource:	The Delphi Pool
Author:	Unknown
Added:	2013-01-27
Last updated:	2013-01-27