Для создания трехмерной формы обрабатываем события **WMNCPaint** и **WMNCHitTest**.

При этом форма должна иметь свойство **BorderStyle** равным *Sizeable*, так как код использует область границ для создания 3D эффекта и предоставляет пользователю возможность изменения размера формы.

Для запрещения изменения размеров формы вы должны включить обработчик события **WMNCHitTest**
, для обратного эффекта не включайте его в ваш код.

```
    { ... }  protected    procedure WMNCPaint(var Msg: TWMNCPaint); message WM_NCPAINT;    procedure WMNCHitTest(var Msg: TWMNCHitTest); message WM_NCHITTEST; { ... }  procedure TForm1.WMNCPaint(var Msg: TWMNCPaint); var  DC: HDC;  Frame_H: Integer;  Frame_W: Integer;  Menu_H: Integer;  Caption_H: Integer; Frame: TRect;  Extra: Integer;  Canvas: TCanvas; begin  { Задаем значения некоторым параметрам окна }  Frame_W := GetSystemMetrics(SM_CXFRAME);  Frame_H := GetSystemMetrics(SM_CYFRAME);  if (Menu  nil) then    Menu_H := GetSystemMetrics(SM_CYMENU)  else    Menu_H := -1;  Caption_H := GetSystemMetrics(SM_CYCAPTION);  GetWindowRect(Handle, Frame);  Frame.Right := Frame.Right - Frame.Left - 1;  Frame.Left := 0;  Frame.Bottom := Frame.Bottom - Frame.Top - 1;  Frame.Top := 0;  { Позволяем нарисовать стандартные границы формы }  inherited;  { Перерисовываем область границ в 3-D стиле }  DC := GetWindowDC(Handle); Canvas := TCanvas.Create;  try    with Canvas do    begin      Handle := DC;      { Левая и верхняя граница }      Pen.Color := clBtnShadow;      PolyLine([Point(Frame.Left, Frame.Bottom), Point(Frame.Left, Frame.Top),        Point(Frame.Right, Frame.Top)]);      { Правая и нижняя граница }      Pen.Color := clWindowFrame; PolyLine([Point(Frame.Left, Frame.Bottom),        Point(Frame.Right, Frame.Bottom), Point(Frame.Right, Frame.Top - 1)]);      { Левая и правая граница, 1 пиксел скраю } Pen.Color := clBtnHighlight;      PolyLine([Point(Frame.Left + 1, Frame.Bottom - 1), Point(Frame.Left + 1, Frame.Top + 1),        Point(Frame.Right - 1, Frame.Top + 1)]);      { Правая и нижняя граница, 1 пиксел скраю }      Pen.Color := clBtnFace; PolyLine([Point(Frame.Left + 1, Frame.Bottom - 1),        Point(Frame.Right - 1, Frame.Bottom - 1),        Point(Frame.Right - 1, Frame.Top)]);      { Разность области изменяемых границ }    for Extra := 2 to (GetSystemMetrics(SM_CXFRAME) - 1) do      begin        Brush.Color := clBtnFace;        FrameRect(Rect(Extra, Extra, Frame.Right - Extra + 1, Frame.Bottom - Extra + 1));      end;      { Левая и верхняя граница области заголовка }      Pen.Color := clBtnShadow;      PolyLine([Point(Frame_W - 1, Frame_H + Caption_H + Menu_H - 1), Point(Frame_W - 1, Frame_H - 1),        Point(Frame.Right - Frame_W + 1, Frame_H - 1)]);  { Левая и верхняя граница области заголовка }      Pen.Color := clBtnHighlight; PolyLine([Point(Frame_W - 1, Frame_H + Caption_H + Menu_H - 1),        Point(Frame.Right -
```

```
Frame_W + 1, Frame_H + Caption_H + Menu_H - 1), Point(Frame.Right - Frame_W + 1, Frame_H - 1)]); end; finally Canvas.Free; ReleaseDC(Handle, DC); end; { try-finally } end; procedure TForm1.WMNCHitTest(var Msg: TWMNCHitTest); var HitCode: LongInt; begin inherited; HitCode := Msg.Result; if ((HitCode = HTLEFT) or (HitCode = HTRIGHT) or (HitCode = HTTOP) or (HitCode = HTBOTTOM) or (HitCode = HTTOPLEFT) or (HitCode = HTBOTTOMLEFT) or (HitCode = HTTOPRIGHT) or (HitCode = HTBOTTOMRIGHT)) then begin HitCode := HTNOWHERE; end; Msg.Result := HitCode; end;
```