Making LMD buttons work when clicked

I use Delphi7 and LMD buttons a lot but got annoyed because the settings do not work for all button types.

For example when *ubswin40* is set, the text moves properly with the button and you can use coloured buttons (but with square corners). When *onautodetect* or *ubsWinXP* you get a pale coloured slightly rounded button (the coloured setting does not work) which is fine but irritatingly (to me anyway) the text doesnt move when you click it although the glyph does.

By putting these procedures in place you can get the text to move properly. There is also a useful routine for putting a set of buttons or any other component like edit boxes into an array.

At the top of the unit:

```
procedure TForm1.FormActivate
var
  k:integer;
begin
  for k:=1 to maxbuttons do
  begin
    new(pbutton[k]);
    pbutton[k]^:= tlmdbutton(FindComponent('LMDButton'+IntToStr(k)));
  end;
end;
```

```
procedure TForm1.FormActivate
var k:integer;
begin
for k:=1 to maxbuttons do
   begin
    new(pbutton[k]);
    pbutton[k]^:= tlmdbutton(FindComponent('LMDButton'+IntToStr(k)));
   end;
end;
```

Note the buttons must be *LMDButton1*, *LMDButton2* etc in sequence or you need to allocate them manually eg `pbutton[1]^:=exitbtn` etc.

In procedure *Tform1.close* or wherever you close the form:

```
var
  k:integer;
begin
  for k:=1 to maxbuttons do dispose(pbutton[k]);
end;
```

Copy the procedures below into your program then select all buttons (Click on each holding the shift key down) press F11 and point *mousedown* and *mouseup* events to the following.

These go in the form definitions:

```
procedure LMDButtonMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure LMDButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
```

These go in the program:

```
procedure TForm1.LMDButtonMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  k:integer;
  b:boolean;
begin
  b:=false;
  k:=1;
  while (k<= maxbuttons) and (not b) do
  begin
    {Note sender indicates which button has been clicked}
    if pbutton[k]^ = sender then
    begin
      with pbutton[k]^ do
      begin
        if not (extstyle = ubswin40) then
        begin
          ButtonLayout.OffsetX:=2;   {you might only want this to be 1 since}
          ButtonLayout.Offsety:=2;   {it depends how you have drawn the glyph}
        end;
      end;
      b:=true;
    end;
    k:=k+1;
  end;
end;

procedure TForm1.LMDButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  k:integer;
begin
for k:= 1 to maxbuttons do
  if pbutton[k]^ = sender then
    with pbutton[k]^ do
    begin
      ButtonLayout.OffsetX:=0;
      ButtonLayout.Offsety:=0;
    end;
end;
```

Author:	Alan Bailey
Contributor:	Alan Bailey
Added:	2008-04-29
Last updated:	2008-04-29