How to create gradient colour schemes

Just cut and paste the routines below into a unit somewhere and make the function declarations at the top of your unit.

You can use *GetGradientColor2* to get a color that is somewhere between two other colors. For example, to get the color that is 50% between Red and Blue, do this:

```
var
  MyColor: TColor;
begin
  R1 := 255;
  G1 := 0;
  B1 := 0;
  R2 := 0;
  G2 := 0;
  B2 := 0;
  Percent := 0.5;
  MyNewColor := GetGradientColor2(R1, G1, B1, R2, G2, B2, Percent);
```

You could put percent in a loop from 0 to 1, and get all the colors as a nice gradient.

Function *GetGradientColor3* works in a similar manner, except that you can do a gradient between 3 colors, such as between red to yellow to blue. This can help prevent the colors from loosing intensity when you go between say blue and red, where the purple would otherwise be darker.

```
function ColorFromRGB(Red, Green, Blue: Integer): Integer;
  {Returns the color made up of the red, green, and blue components.
  Red, Green, and Blue can be from 0 to 255.}
begin
  {Convert Red, Green, and Blue values to color.}
  Result := Red + Green * 256 + Blue * 65536;
end;

function GetPigmentBetween(P1, P2, Percent: Double): Integer;
  {Returns a number that is Percent of the way between P1 and P2}
begin
  {Find the number between P1 and P2}
  Result := Round(((P2 - P1) * Percent) + P1);
  {Make sure we are within bounds for color.}
  if Result > 255 then
    Result := 255;
  if Result < 0 then
    Result := 0;
end;

function GetGradientColor2(R1, G1, B1, R2, G2, B2, Percent: Double): Integer;
  {Gets a color that is inbetween the colors defined by (R1,G1,B1)
  and (R2,G2,B2) Percent ranges from 0 to 1.0 (i.e. 0.5 = 50%)
  If percent = 0 then the color of (R1,G1,B1) is returned
  If Percent = 1 then the color of (R2,G2,B2) is returned
  if Percent is somewhere inbetween, then an inbetween color is returned.}
var
  NewRed, NewGreen, NewBlue: Integer;
begin
  {Validate input data in case it is off by a few thousanths.}
  if Percent > 1 then
    Percent := 1;
  if Percent < 0 then
    Percent := 0;
  {Calculate Red, green, and blue components for the new color.}
  NewRed := GetPigmentBetween(R1, R2, Percent);
  NewGreen := GetPigmentBetween(G1, G2, Percent);
  NewBlue := GetPigmentBetween(B1, B2, Percent);
  {Convert RGB to color}
  Result := ColorFromRGB(NewRed, NewGreen, NewBlue);
end;

function GetGradientColor3(R1, G1, B1, R2, G2, B2, R3, G3, B3,
  Percent: Double): Integer;
```

```
  {Gets a color that is inbetween the color spread defined (R1,G1,B1),
  (R2,G2,B2) and (R3,G3,B3). This is similar to GetGradientColor2,
  except that it allows you to specify 3 colors instead of 2.}
begin
  {Use GetGradient2 to do most the work}
  if Percent < 0.5 then
    Result := GetGradientColor2(
      R1, G1, B1, R2, G2, B2, Percent * 2
    )
  else
    Result := GetGradientColor2(
      R2, G2, B2, R3, G3, B3, (Percent - 0.5) * 2
    );
end;
```

Original resource:	The Delphi Pool
Author:	Carl Olsen
Added:	2009-11-06
Last updated:	2009-11-06