# Color Tables in GR Programs, version 3

This reference guide describes Color Tables as implemented in version 3 of the GR programs. A color table is a simple case-insensitive text file that contains a ColorTable block.  The ColorTable block tells the GR program which built-in data Category the color table applies to, such as BR, BV, VIL, etc., along with the Units it works with, and the Scale and Offset used to convert the built-in Category's units to the ColorTable units. The file must have an extension of ".pal3.txt" or ".pal3".

ColorTable 3.00 (CT3) supports two color spaces: RGB and HSLuv. RGB is the standard, well-known red+green+blue color space in computer graphics. HSLuv is a newer color space that attempts to make specifying colors more "natural" in computer graphics. Its components are hue, saturation, and lightness. More information on the HSLuv color space is available at the following link:

https://www.hsluv.org/

Here is the default color table for base reflectivity (BR):

```
ColorTable
 {
  Category = "BR"       // this color table is for base reflectivity (BR)
  Units    = "DBZ"      // its units are DBZ
  Step     = 10         // tick marks are spaced 10 dbz apart

  Color[80] =           rgb( 128, 128, 128 )
  Color[70] =           rgb( 255, 255, 255 )
  Color[60] = gradient( rgb( 255,   0, 255 ), rgb( 128,   0, 128 ) )
  Color[50] = gradient( rgb( 255,   0,   0 ), rgb( 160,   0,   0 ) )
  Color[40] = gradient( rgb( 255, 255,   0 ), rgb( 255, 128,   0 ) )
  Color[30] = gradient( rgb(   0, 255,   0 ), rgb(   0, 128,   0 ) )
  Color[20] = gradient( rgb(  64, 128, 255 ), rgb(  32,  64, 128 ) )
  Color[10] = gradient( rgb( 164, 164, 255 ), rgb( 100, 100, 192 ) )
}
```

It begins with three required parameters: Category, Units, and Step. These describe the color table contents to your GR program and how to draw the color bar in the display window. The Color[] statements that follow assign bands of color to data values. The data values are the starting point for the assigned color band. The end value of the color band is the start data value of the next higher-valued Color[] statement. For example, the Color[10] and Color[20] statements together create a color band from 10 dbz to 20 dbz. The Color[10] statement says you want an RGB gradient color band between those two values.

## Describing your Color Table to the Program

You need to describe your color table to the GR program. **There are three required parameters**:

```
Category = "string"

    string must be the name of a built-in data category, e.g., "BR"

Units = "string"

    string must be a string no longer than 15 characters, e.g., "DBZ"

Step = number

    number must be greater than zero, the distance between tick marks shown
```

Two optional parameters describe how to convert the GR program's internal units for a category to those used in your color table:

```
Scale = number

    number is the scale used when converting internal units to those in
    your color table

Offset = number

    number is the offset used when converting internal units to those in
    your color table
```

These values are used to convert internal units to your color table's units with the following equation:

```
converted_value = scale * internal_value + offset
```

Other optional parameters are:

```
Decimals = number

    number is an integer number of places to show after the decimal
    point. The default is zero.

ND = SingleColor
```

```
    SingleColor is used in the nonsmoothed display when the underlying
    data value is No Data (ND)


RF = SingleColor

    SingleColor is used in the nonsmoothed display when the underlying
    data value is Range Folded (RF)
```

The final optional parameter allows you to show text labels instead of numbers on the color bar in the display window:

```
Label[data_value] = "string"

    string is a short string drawn in the center of the color band containing
    the data_value. The string's length should be less than six characters.
```

You can place multiple labels. The Label[] statements must come after the Color[] statements that define the color bands. Labels will be drawn in the center of the color band into which their *data_value* falls. Labels should only be used for data categories that contain discrete text information, for example, the Hydrometeor Classification Algorithm. They are not designed for classifying data into “Light”, “Medium”, and “Heavy” ranges.

## Specifying a Color

The fundamental unit of color is a *SingleColor*. It gives the color as a point in the color space. There are two ways to specify a color, one for each color space:

```
SingleColor = rgb( r, g, b, a )

       Specifies a color in the RGB color space. The A channel is optional.
       R, G, B, and A must be a number from zero to 255. If A is not given,
       it defaults to 255.


SingleColor = hsluv( h, s, l, a )

       Specifies a color in the HSLuv color space. The A channel is optional.
       H must be a number from 0-360. S must be from 0-100, and L must be from
       0-100. If A is given, it must be from 0-255. If A is not given, it
       defaults to 255.
```

The first three numbers are required and can be followed by an optional alpha component. The alpha component gives the opacity as a number from zero (totally transparent) to 255 (fully opaque).

## Color Bands

Internally, the GR programs work with color bands, that is, a start data value and color and an end data value and color. The programs interpolate between the two colors based on the relationship between the input data value and the start/end data values. You specify the start data value, and the program automatically sets the end data value for the band to the next higher data value given. There are three types of color bands you can choose from:

1. **Single color band**, where you give the color for a particular data value as the start color and let the next higher data value's start color be the end color for the band.

2. **Solid color band**, where you give a single color for data values from the start value to the end value.

3. **Gradient color band**, where you give both the start and end colors for the band.

Here's how you define a color band:

```
ColorBand = SingleColor

    Assigns a SingleColor to the start value in a Color[] statement


ColorBand = Solid( SingleColor )

    Sets the entire band to a constant color


ColorBand = Gradient( SingleColor1, SingleColor2 )

    Sets the start color in a band to SingleColor1 and the end color in the
    band to SingleColor2. Each component of the colors is interpolated across
    the color band from the start data value to the end data value.
```

## Assigning a Color Band to a Data Value

The following statement is used to assign a color band to a data value:

```
Color[start_data_value] = ColorBand
```

This statement assigns the given *ColorBand* to *start_data_value*. The program will automatically fill in the band's *end_data_value* from the next higher Color[] statement's *start_data_value*. Note that all the Color[] statements are loaded and then sorted by their *start_data_value* before assigning *end_data_value*, which means your Color[] statements do not need to be in order by their *start_data_value*.

You must have at least two Color[] statements in your color table file.