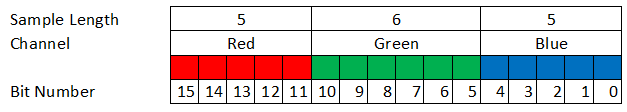
**BoBo Fetts 16-bit mat specs**

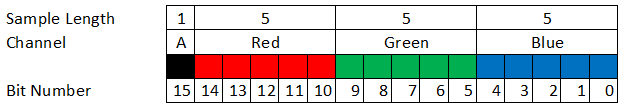
Oct 31,2021

**Known 16-bit mat formats supported by JediKnight:**

565 RGB:



1555 ARGB:



Note: The following code is in Delphi format

The header for a 16-bit mat is:

TMatHeader = record

tag:array[0..3] of ANSIchar; // 'MAT ' - notice space after MAT

ver:Longint; // Apparently - version = 0x32 ('2')

mat\_Type:Longint; // 0 = colors(TColorHeader) , 1= ?, 2= texture(TTextureHeader)

record\_count:Longint; // record\_count {number of textures or colors

cel\_count: Longint; // cel\_count { In color MATs, it's 0, in TX ones, it's equal to numOfTextures

ColorMode:Longint; // {ColorMode, Indexed = 0 RGB = 1 RGBA = 2

bits:LongInt; // = 16 {bits/pixel}

redbits:longint; // {red bits per pixel} {ignored by game engine}

greenbits:longint; // {green bits per pixel} {ignored by game engine}

bluebits:longint; // {blue bits per pixel} {ignored by game engine}

shiftR:longint; // bit index to red color channel, shift left during conversion { = 11 or 8} {ignored by game engine}

shiftG:longint; // bit index to green color channel, shift left during conversion { = 5 or 4} {ignored by game engine}

shiftB:longint; // bit index to blue color channel, shift left during conversion { = 0 } {ignored by game engine}

RedBitDif: longint; // bits shifted right during conversion from 8bit to 5bit =3 {ignored by game engine}

GreenBitDif: longint; // bits shifted right during conversion from 8bit to 6bit =2 {ignored by game engine}

BlueBitDif: longint; // bits shifted right during conversion from 8bit to 5bit =3 {ignored by game engine}

alpha\_bpp:longint; //=0 {ignored by game engine}

alpha\_sh:longint; //=0 shift left during conversion {ignored by game engine}

alpha\_BitDif:longint; //=0 shifted right during conversion {ignored by game engine}

end;

TTextureHeader = record

textype: longint; {0 = color, 8= texture}

transparent\_color: : longint; transparent\_color {With 8-bit images, is an index into the palette. .}

pads: array[0..2] of longint;

unk1tha: word; {ignored by game engine}

unk1thb: word;

unk2th: longint; //=0

unk3th: longint; //=4 {ignored by game engine}

unk4th: longint; //=4 {ignored by game engine}

cel\_idx: longint; //=0 for first texture. Inc. for every texture in mat

end;

TTextureData = record

SizeX: longint; {horizontal size of first MipMap, must be divisable by 2}

SizeY: longint; {Vertical size of first MipMap ,must be divisable by 2}

Transparent: longint; {1: transparent on, else 0: transparent off}

Pad: array[0..1] of longint;{padding = 0 }

{padding = 0 }

NumMipMaps: longint; {Number of mipmaps in texture largest one first.}

end;