BMP File Format

Um ficheiro BMP é constituído por 4 blocos principais de informação:

- Um cabeçalho do ficheiro - BitmapFileHeader
- Um cabeçalho da imagem (DIB) - BitmapInfoHeader
- Opcional: Uma palete de cores dependendo do BitmapInfoHeader
- Um bloco com a imagem – pixel data

De seguida passamos a descrever cada um desses blocos:

```
Class Bitmap{
    BitmapFileHeader bitmapFileHeader;
    BitmapInfoHeader bitmapInfoHeader;
    byte[] rgbQuad;  // color pallete – opcional (ver abaixo)
    byte[] data;     // pixel data
}


Class BitmapFileHeader {
    short type;       // must be 'BM' to declare a bmp-file
    int size;         // specifies the size of the file in bytes
    short reserved1;  // must always be set to zero
    short reserved2;  // must always be set to zero
    int offBits;      // specifies the offset from the
                      // beginning of the file to the bitmap data
}


Class BitmapInfoHeader{
    int size;          // the size of this header (40 bytes)
    int width;         // the bitmap width in pixels
    int height;        //  the bitmap height in pixels
    short planes;      //  the number of color planes being used. Must be set to 1
    short bitCount;    // the number of bits per pixel (color depth) - 1, 4, 8, 16, 24, 32
    int compression;   //  the compression method being used
    int sizeImage;     //  the image size. This is the size of the raw bitmap data
    int xPelsPerMeter; //  the horizontal resolution of the image (pixel per meter)
    int yPelsPerMeter; //  the vertical resolution of the image (pixel per meter)
    int clrUsed;       //  the number of colors in the color palette,
                       //   or 0 to default to 2n
    int clrImportant;  //  the number of important colors used,
                       //   or 0 when every color is important
}
```

RGBQUAD structure (ColorPallete):

Start	Size	Name	Purpose
1	1	rgbBlue	Specifies the blue part of the color.

2	1	rgbGreen	Specifies the green part of the color.
3	1	rgbRed	Specifies the red part of the color.
4	1	rgbReserved	must always be set to zero.

Some Important Notes:

1. Pixels are stored "upside-down" (Last row first).
2. RGB color (24-bit) pixel values are stored with bytes as BGR (blue, green, red).
3. Padding bytes are inserted in order to keep the line of data in multiples of four.
4. The color palette is not used when the bitmap is 16-bit or higher. There is no palette byte array (`byte[] rgbQuad;`) in those BMP files.
5. BitmapInfoHeader - compression method field:

Value	Identified by	Compression method	Comments
0	BI_RGB	none	Most common
1	BI_RLE8	RLE 8-bit/pixel	Can be used only with 8-bit/pixel bitmaps
2	BI_RLE4	RLE 4-bit/pixel	Can be used only with 4-bit/pixel bitmaps
3	BI_BITFIELDS	Bit field	Can be used only with 16 and 32-bit/pixel bitmaps.
4	BI_JPEG	JPEG	The bitmap contains a JPEG image
5	BI_PNG	PNG	The bitmap contains a PNG image

Example of a 2×2 Pixel Bitmap, with 24 bits/pixel encoding

Offset	Size	Hex Value	Value	Description
0h	2	42 4D	"BM"	Magic Number (unsigned integer 66, 77)
2h	4	46 00 00 00	70 Bytes	Size of the BMP file
6h	2	00 00	Unused	Application Specific
8h	2	00 00	Unused	Application Specific
Ah	4	36 00 00 00	54 bytes	The offset where the bitmap data (pixels) can be found.
Eh	4	28 00 00 00	40 bytes	The number of bytes in the header (from this point).
12h	4	02 00 00 00	2 pixels	The width of the bitmap in pixels
16h	4	02 00 00 00	2 pixels	The height of the bitmap in pixels
1Ah	2	01 00	1 plane	Number of color planes being used.
1Ch	2	18 00	24 bits	The number of bits/pixel.
1Eh	4	00 00 00 00	0	BI_RGB, No compression used
22h	4	10 00 00 00	16 bytes	The size of the raw BMP data (after this header)

26h	4	`13 0B 00 00`	2,835 pixels/meter	The horizontal resolution of the image
2Ah	4	`13 0B 00 00`	2,835 pixels/meter	The vertical resolution of the image
2Eh	4	`00 00 00 00`	0 colors	Number of colors in the palette
32h	4	`00 00 00 00`	0 important colors	Means all colors are important
Start of Bitmap Data				
36h	3	`00 00 FF`	0 0 255	Red, Pixel (0,1)
39h	3	`FF FF FF`	255 255 255	White, Pixel (1,1)
3Ch	2	`00 00`	0 0	Padding for 4 byte alignment (Could be a value other than zero)
3Eh	3	`FF 00 00`	255 0 0	Blue, Pixel (0,0)
41h	3	`00 FF 00`	0 255 0	Green, Pixel (1,0)
44h	2	`00 00`	0 0	Padding for 4 byte alignment (Could be a value other than zero)

BMP Integer *versus* JAVA Integer

O ficheiro BMP é conhecido com "Windows and OS/2 bitmap file format" e a representação dos seus inteiros segue o formato little-endian (Least Significant Byte first). No entanto, em Java os dados em ficheiros binários seguem a representação big-endian (Most Significant Byte first).

Se na leitura de bytes é transparente, para converter um inteiro litte-endian (BMP) para bigendian (Java) devemos utilizar os métodos `Short.reverseBytes` e `Integer.reverseBytes`.

Por exemplo:

```
short  bitCount = Short.reverseBytes(fin.readShort());

int size = Integer.reverseBytes(fin.readInt());
```