BMP 文件的编码方式

BMP 是一种与硬件设备无关的图像文件格式，也是我们最常在 PC 机上的 Windows 系统下见到的标准位图格式，使用范围很广泛。它采用位映射存储格式，除了图像深度可选以外，不采用其他任何压缩，因此，BMP 文件所占用的空间很大。它最大的好处就是能被大多数软件“接受”，可称为通用格式。

BMP 在过去是比较普及的图像格式，现在 BMP（Window 位图）图像主要被用在 PC 机运行 Window 时的墙纸。BMP 可以提供无损压缩，压缩方式叫 RLE（游程长度编码的编写），在创建墙纸图像文件时是一个极好的选项。Window 有时在查找以 RLE 压缩文件方式保存的墙纸图像时也会出现识别错误。，因此使用时最好先关闭 RLE 压缩功能。
BMP 文件由文件头、位图信息头、颜色信息和图形数据四部分组成。
1、 BMP 文件头：BMP 文件头数据结构含有 BMP 文件的类型、文件大小和位图起始位置等信息。

```
typedef struct tagBITMAPFILEHEADER{
WORD bfType; // 位图文件的类型，必须为 BM
DWORD bfSize; // 位图文件的大小，以　　为单位
WORD bfReserved1; // 位图文件保留字，必须为 0
WORD bfReserved2; // 位图文件保留字，必须为 0
DWORD bfOffBits; // 位图数据的起始位置，以相对于位图文件头的偏移量表示，以　　为单位
} BITMAPFILEHEADER;
```

2、 位图信息头：BMP 位图信息头数据用于说明位图的尺寸等信息。

```
typedef struct tagBITMAPINFOHEADER{
DWORD biSize; // 本结构所占用　　数
LONGbiWidth; // 位图的宽度，以像素为单位
LONGbiHeight; // 位图的高度，以像素为单位
WORD biPlanes; // 目标设备的级别，必须为 1
WORD biBitCount// 每个像素所需的位数，必须是 1(双色),4(16 色)，8(256 色)或 24(真彩色)之一
DWORD biCompression; // 位图压缩类型，必须是 0(不压缩),1(BI_RLE8 压缩类型)或 2(BI_RLE4 压缩类型)之一
DWORD biSizeImage; // 位图的大小，以　　为单位
LONG biXPelsPerMeter; // 位图水平分辨率，每米像素数
LONG biYPelsPerMeter; // 位图垂直分辨率，每米像素数
DWORD biClrUsed;// 位图实际使用的颜色表中的颜色数
DWORD biClrImportant;// 位图显示过程中重要的颜色数
} BITMAPINFOHEADER;
```

3、颜色表：颜色表用于说明位图中的颜色，它有若干个表项，每一个表项是一个 RGBQUAD 类型的结构，定义一种颜色。

```
typedef struct tagRGBQUAD {
BYTE rgbBlue;// 蓝色的亮度(值范围为 0-255)
BYTE rgbGreen; // 绿色的亮度(值范围为 0-255)
BYTE rgbRed; // 红色的亮度(值范围为 0-255)
```

BYTE rgbReserved;// 保留，必须为 0
} RGBQUAD;
颜色表中 RGBQUAD 结构数据的个数有 biBitCount 来确定:
当 biBitCount=1,4,8 时，分别有 2,16,256 个表项;
当 biBitCount=24 时，没有颜色表项。
位图信息头和颜色表组成位图信息，BITMAPINFO 结构定义如下:
typedef struct tagBITMAPINFO {
BITMAPINFOHEADER bmiHeader; // 位图信息头
RGBQUAD bmiColors[1]; // 颜色表
} BITMAPINFO;
4、 位图数据：位图数据记录了位图的每一个像素值，记录顺序是在扫描行内是从左到右,扫描行之间是从下到上。位图的一个像素值所占的 数:
当 biBitCount=1 时，8 个像素占 1 个 ;
当 biBitCount=4 时，2 个像素占 1 个 ;
当 biBitCount=8 时，1 个像素占 1 个 ;
当 biBitCount=24 时,1 个像素占 3 个 ;
Windows 规定一个扫描行所占的 数必须是 4 的倍数(即以 long 为单位),不足的以 0 填充，一个扫描行所占的 数计算方法:
DataSizePerLine= (biWidth* biBitCount+31)/8; // 一个扫描行所占的 数
DataSizePerLine= DataSizePerLine/4*4; // 数必须是 4 的倍数
位图数据的大小(不压缩情况下):
DataSize= DataSizePerLine* biHeight;
二、BMP 文件分析
1、 工具软件：Hex Workshop 或 UltraEdit
2、 分析：首先请注意所有的数值在存储上都是按"高位放高位、低位放低位的原则"，如 12345678h 放在存储器中就是 7856 3412）。下图是一张图 16 进制数据，以此为例进行分析。在分析中为了简化叙述，以一个字（两个 为单位，如 424D 就是一个字）为序号单位进行，"h"表示是 16 进制数。
424D 4690 0000 0000 0000 4600 0000 2800
0000 8000 0000 9000 0000 0100 1000 0300
0000 0090 0000 A00F 0000 A00F 0000 0000
0000 0000 0000 00F8 0000 E007 0000 1F00
0000 0000 0000 02F1 84F1 04F1 84F1 84F1
06F2 84F1 06F2 04F2 86F2 06F2 86F2 86F2
1：图像文件头。424Dh='BM'，表示是 Windows 支持的 BMP 格式。
2-3：整个文件大小。4690 0000，为 00009046h=36934。
4-5：保留，必须设置为 0。
6-7：从文件开始到位图数据之间的偏移量。4600 0000，为 00000046h=70，上面的文件头就是 35 字=70 。
8-9：位图图信息头长度。
10-11：位图宽度，以像素为单位。8000 0000，为 00000080h=128。
12-13：位图高度，以像素为单位。9000 0000，为 00000090h=144。
14：位图的位面数，该值总是 1。0100，为 0001h=1。
15：每个像素的位数。有 1（单色），4（16 色），8（256 色），16（64K 色，高彩色），24（16M

色，真彩色），32（4096M 色，增强型真彩色）。T408 支持的是 16 位格式。1000 为 0010h=16。
16-17：压缩说明：有 0（不压缩），1（RLE 8，8 位 RLE 压缩），2（RLE 4，4 位 RLE 压缩），3（Bitfields，位域存放）。RLE 简单地说是采用像素数+像素值的方式进行压缩。T408 采用的是位域存放方式，用两个　　表示一个像素，位域分配为 r5b6g5。图中 0300 0000 为 00000003h=3。
18-19：用　　数表示的位图数据的大小，该数必须是 4 的倍数，数值上等于位图宽度×位图高度×每个像素位数。0090 0000 为 00009000h=80×90×2h=36864。
20-21：用象素/米表示的水平分辨率。A00F 0000 为 0000 0FA0h=4000。
22-23：用象素/米表示的垂直分辨率。A00F 0000 为 0000 0FA0h=4000。
2：位图使用的颜色索引数。设为 0 的话，则说明使用所有调色板项。
26-27：对图象显示有重要影响的颜色索引的数目。如果是 0，表示都重要。
28-35：彩色板规范。对于调色板中的每个表项，用下述方法来描述 RGB 的值：
1　　用于蓝色分量
1　　用于绿色分量
1　　用于红色分量
1　　用于填充符(设置为 0)
对于 24-位真彩色图像就不使用彩色表，因为位图中的 RGB 值就代表了每个象素的颜色。但是 16 位 r5g6b5 位域彩色图像需要彩色表，看前面的图，与上面的解释不太对得上，应以下面的解释为准。
图中彩色板为 00F8 0000 E007 0000 1F00 0000 0000 0000，其中：
00FB 0000 为 FB00h=1111100000000000（二进制），是红色分量的掩码。
E007 0000 为 07E0h=0000011111100000（二进制），是绿色分量的掩码。
1F00 0000 为 001Fh=0000000000011111（二进制），是红色分量的掩码。
0000 0000 总设置为 0。
将掩码跟像素值进行"与"运算再进行移位操作就可以得到各色分量值。看看掩码，就可以明白事实上在每个像素值的两个　　16 位中，按从高到低取 5、6、5 位分别就是 r、g、b 分量值。取出分量值后把 r、g、b 值分别乘以 8、4、8 就可以补齐第个分量为一个　　，再把这三个　　按 rgb 组合，放入存储器（同样要反序），就可以转换为 24 位标准 BMP 格式了。

另外，用 ultraedit 可查看几乎所有文件的二进制代码，