5.3 GIF格式

GIF(Graphics Interchange Format)的原义是“[图像](http://baike.baidu.cn/view/42116.htm)互换格式”，是CompuServe公司在 1987年开发的[图像文件格式](http://baike.baidu.cn/view/3413757.htm)。GIF文件的数据，是一种基于[LZW算法](http://baike.baidu.cn/view/401141.htm)的连续色调的无损[压缩格式](http://baike.baidu.cn/view/2954654.htm)。其[压缩率](http://baike.baidu.cn/view/354638.htm)一般在50%左右，它不属于任何[应用程序](http://baike.baidu.cn/view/330120.htm)。目前几乎所有相关[软件](http://baike.baidu.cn/view/37.htm)都支持它，公共领域有大量的软件在使用GIF[图像](http://baike.baidu.cn/view/42116.htm)文件。GIF[图像](http://baike.baidu.cn/view/42116.htm)文件的数据是经过压缩的，而且是采用了可变长度等压缩算法。GIF格式的另一个特点是其在一个GIF文件中可以存多幅彩色[图像](http://baike.baidu.cn/view/42116.htm)，如果把存于一个文件中的多幅[图像数据](http://baike.baidu.cn/view/286852.htm)逐幅读出并显示到[屏幕](http://baike.baidu.cn/view/207233.htm)上，就可构成一种最简单的动画。

从整体结构上来说，GIF文件由全局信息表、调色板数据、局部信息表和像素数据4部分组成。全局信息表和局部信息表的长度是固定的，全局信息表的长度恒为13，而局部信息表的长度恒为10字节。一个GIF文件可能存储着多幅图像，但整个文件只有一个全局信息表而每幅图像则各自拥有一个局部信息表，局部信息表以逗号做前导符，整个文件以分号结尾。

5.3.1GIF文件结构

GIF文件结构的典型结构如表1所示。为方便下文说明，在构件左边加了编号。

表1 GIF文件结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Header |  | GIF文件头 |  |
| 2 | Logical Screen Descriptor |  | 逻辑屏幕描述块 |  |
| 3 | Global Color Table |  | 全局彩色表 |  |
|  | … 扩展模块(任选) … |  |  |  |
| 4 | Image Descriptor |  | 图形描述块 |  |
| 5 | Local Color Table |  | 局部彩色表(可重复n次) | 可 |
| 6 | Table Based Image Data |  | 表式压缩图像数据 | 重 |
| 7 | Graphic Control Extension |  | 图像控制扩展块 | 复 |
| 8 | Plain Text Extension |  | 无格式文本扩展块 | n |
| 9 | Comment Extension |  | 注释扩展块 | 个 |
| 10 | Application Extension |  | 应用程序扩展块 |  |
|  | … 扩展模块(任选) … |  |  |  |
| 11 | GIF Trailer |  | GIF文件结束块 |  |

按块的特征可以分为三类：控制块(Control Block)，图形描绘块(Graphic-Rendering Block)和专用块(Special Purpose Block)。

1、控制块

控制块包含有用来控制数据流(Data Stream)或者设置硬件参数的信息，其成员包括：

* GIF文件头(Header)
* 逻辑屏幕描述块(Logical Screen Descriptor)
* 图形控制扩展块(Graphic Control Extension)
* 文件结束块(Trailer)

2、图形描绘块

包含有用来描绘在显示设备上显示图形的信息和数据，其成员包括：

* 图像描述块(Image Descriptor)
* 无格式文本扩展块(Plain Text Extension)
* 全局调色板、局部调色板、图像压缩数据和图像说明扩充块

3、特殊用途数据块

包含有与图像处理无关的信息，其成员包括：

* 注释扩展块(Comment Extension)
* 应用扩展块(Application Extension)

除了在控制块中的逻辑屏幕描述块(Logical Screen Descriptor)和全局彩色表(Global Color Table)的作用范围是整个数据流(Data Stream)之外, 所有其他控制块仅控制跟在它们后面的图形描绘块。

5.3.2 构件详解

1、GIF文件头

文件头描述块(Header)定义GIF数据流(GIF Data Stream)，它的结构如图1所示。文件头描述块(Header)由GIF标记域(Signature)和版本号(Version)域组成，是一个由6个固定字节组成的数据块，它们用来说明使用的文件格式是GIF格式及当前所用的版本号。GIF标记域(Signature)存放的是“GIF”，版本号域存放的是1987年5月发布的“87a”或者1989年7月发布的“89a”，或者更加新的版本号。

typedef struct gifheader{  
        BYTE bySignature[3];  
        BYTE byVersion[3];  
     } GIFHEADER;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | 字节号 | | 域的名称 | | 数据类型 | |
|  | | | | | | | | 0 | |  | |  | |
| Signature | | | | | | | | 1 | | GIF标记 | | 3 Bytes | |
|  | | | | | | | | 2 | |  | |  | |
|  | | | | | | | | 3 | |  | |  | |
| Version | | | | | | | | 4 | | 版本号 | | 3 Bytes | |
|  | | | | | | | | 5 | |  | |  | |

图1 标记/版本数据块的结构

2、逻辑屏幕描述块

逻辑屏幕描述块(Logical Screen Descriptor)包含定义图像显示区域的参数，包括背景颜色信息。这个数据块中的坐标相对于虚拟屏幕的左上角，不一定是指显示屏的绝对坐标，这就意味可以参照窗口软件环境下的窗口坐标或者打印机坐标来设计图像显示程序。逻辑屏幕描述块的结构如图2所示：

typedef struct gifscrdesc{  
        WORD wWidth;  
        WORD wDepth;  
        struct globalflag{  
              BYTE PalBits :3;  
              BYTE SortFlag :1;  
              BYTE ColorRes :3;  
              BYTE GlobalPal :1;  
           }    GlobalFlag;  
        BYTE byBackground;  
        BYTE byAspect;  
     }    GIFSCRDESC;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | | 6 | 5 | 4 | | 3 | | 2 | 1 | 0 | 字节号 | 域的名称 | 类型 | |
| Logical Screen Width | | | | | | | | | | | 0 | 逻辑屏幕宽度 | Unsigned | |
|  | | | | | | | | | | | 1 | (以像素为定单位) |  | |
| Logical Screen Height | | | | | | | | | | | 2 | 逻辑屏幕高度 | Unsigned | |
|  | | | | | | | | | | | 3 | (以像素为定单位) |  | |
| G | CR | | | | S | | Size | | | | 4 | 包装域 | | 见图3 |
| Background Color Index | | | | | | | | | | | 5 | 背景颜色索引 | Byte | |
| Pixel Aspect Ratio | | | | | | | | | | | 6 | 像素宽高比 | Byte | |

 图2 屏幕描述块的结构

逻辑描述块包含7个字节。字节0和字节1用来说明逻辑显示屏的宽度，字节3和字节4用来说明逻辑显示屏的高度，字节4用来描述彩色表的属性，字节5用来指定背景颜色索引，字节6用来计算像素的宽高比。现作如下说明：

1) 屏幕描述块中的第5个字节称为包装域(Packed Fields)，它的位结构如图2所示，它由4个子域组成：

(1) 全局彩色表标志(Global Color Table Flag )域G用来说明是否有全局彩色表存在。如果G＝1，表示有一个全局彩色表(Global Color Table)将紧跟在这个逻辑屏幕描述块(Logical Screen Descriptor)之后；这个标志也用来选择背景颜色索引(Background Color Index)。如果G＝1，背景颜色索引(Background Color Index)域中的值就用作背景颜色的索引。

(2) 彩色分辨率(Color Resolution)域CR用来表示原始图像可用的每种基色的位数(实际值减1)。这个位数表示整个调色板的大小，而不是这幅图像使用的实际的颜色数。例如，如果该域的值CR＝3，说明原始图像可用每个基色有4位的调色板来生成彩色图像。

(3) 彩色表排序标志(Sort Flag)域S用来表示全局彩色表(Global Color Table)中的颜色是否按重要性(或者称使用率)排序。如果S＝0，表示没有重要性排序；如果S＝1表示最重要的颜色排在前。这样做的目的是辅助颜色数比较少的解码器能够选择最好的颜色子集，在这种情况下解码器就可选择彩色表中开始段的彩色来显示图像。

(4) 全局彩色表大小(Size of Global Color Table)域Size表示表示每个像素的位数，它用来计算全局彩色表(Global Color Table)中包含的字节数。在全局彩色表标志(Global Color Table Flag)域G＝0时就不需要计算，G＝1时就要计算彩色表的大小，具体计算见下文的“3、全局彩色表”。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Global Color Table Flag | Color Resolution | | | Sort Flag | Size of Global Color Table | | |

图3 逻辑屏幕描述块中的包装域结构

2) 屏幕描述块中的第6个字节是背景颜色索引(Background Color Index)，它是彩色表的一个索引值，用来指定背景颜色。如果全局彩色表标志(Global Color Table Flag)域G＝0，这个域的值也设置为0。

3) 像素宽高比(Pixel Aspect Ratio)域中的值是一个因数，是计算原始图像像素的宽高比的一个近似值。如果该域的值范围为1～255，如果不等于0，宽高比的近似值按下式计算：

Aspect Ratio = (Pixel Aspect Ratio + 15) / 64  
像素宽高比(Pixel Aspect Ratio)定义成像素的宽度与高度之比，比值的范围在4:1～1:4之间，其增量为1/64。

3、全局彩色表

由于一个GIF文件可以包含多幅彩色图像，每幅彩色图像也许都包含适合自身特点的彩色表，所以一个GIF文件可以有好几个彩色表。但归纳起来只有两类：全局彩色表(Global Color Table)或局部彩色表(Local Color Table)。全局彩色表可用于图像本身没有带彩色表的所有图像和无格式文本扩展块(Plain Text Extension)，而局部彩色表只用于紧跟在它后面的一幅图像。在处理全局彩色表和局部彩色表时需要注意下面一些规则。

(1)如果GIF文件包含全局彩色表(Global Color Table)，而且要显示的图像本身又带有局部彩色表，那末显示该幅彩色图像时就用它自己的彩色表，而不用全局彩色表。在这种情况下，解码器就首先保存全局彩色表(Global Color Table)，然后使用局部彩色表(Local Color Table)来显示图像，最后再回复全局彩色表(Global Color Table)。

(2)全局彩色表(Global Color Table)和局部彩色表(Local Color Table)都是可选择的。由于这个原因，解码器最好要保存全局彩色表(Global Color Table)，一直到出现另一个全局彩色表(Global Color Table)为止。这样做之后，对于包含完全没有彩色表的一幅或者多幅彩色图像的GIF文件就可以使用最后保存的全局彩色表(Global Color Table)进行处理。

(3)如果同类型的图像能够使用相同的彩色表来显示，编码器就要尽可能使用一个全局彩色表(Global Color Table)；如果没有彩色表可用，解码器就可以使用计算机系统提供的彩色表或者解码器自身的彩色表。

(4)全局彩色表(Global Color Table)存在与否由逻辑屏幕描述块(Logical Screen Descriptor)中字节5的全局彩色表标志(Global Color Table Flag )域G的值确定。如果存在，彩色表就紧跟在逻辑屏幕描述块(Logical Screen Descriptor)之后。彩色表的表项数目等于2(*n*+1)，其中n=b2b1b0，每个表项由三个字节组成，分别有R、G、B的相对强度，因此彩色表的字节数等于3×2(*n*+1)，

彩色的结构如图4所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 字节号 | 域的名称 | 数据类型 |
| red intensity | 0 | 红色索引 000 | Byte |
| green intensity | 1 | 绿色索引 000 | Byte |
| blue intensity | 2 | 蓝色索引 000 | Byte |
| red intensity | 3 | 红色索引 001 | Byte |
| green intensity | 4 | 绿色索引 001 | Byte |
| blue intensity | 5 | 蓝色索引 001 | Byte |
| … | … | … |  |
| … | … | … |  |
| red intensity | 745 | 红色索引 255 | Byte |
| green intensity | 746 | 绿色索引 255 | Byte |
| blue intensity | 767 | 蓝色索引 255 | Byte |

图4 彩色表结构

局部彩色表与全局彩色表有相同的存储格式。

4、图像描述块

GIF图像文件格式可包含数量不限的图像，而且也没有一个固定的存放顺序，仅用一个字节的图像分隔符(Image Separator)来判断是不是图像描述块。每一幅图像都由一个图像描述块(Image Descriptor)、可有可无的局部彩色表(Local Color Table)和图像数据组成。每幅图像必须要落在逻辑屏幕描述块(Logical Screen Descriptor)中定义的逻辑屏(Logical Screen)尺寸范围里。

图像描述块(Image Descriptor)之前可以有一个或者多个控制块，例如图形控制扩展块(Graphic Control Extension)，其后可以跟着一个局部彩色表(Local Color Table)。无论前后是否有各种数据块，图像描述块(Image Descriptor)总是带有图像数据。

图像描述块(Image Descriptor)的结构如图5所示。

typedef struct gifimage{  
　　        WORD wLeft;  
　　        WORD wTop;  
　　        WORD wWidth;  
　　        WORD wDepth;  
　　        struct localflag{  
　　              BYTE PalBits     : 3;  
　　              BYTE Reserved    :2;  
　　              BYTE SortFlag    :1;  
　　              BYTE Interlace :1;  
　　              BYTE LocalPal    :1;   
　　           }    LocalFlag;  
　　     }    GIFIMAGE;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 字节号 | | 域的名称 | | 类型 | | |
| Image Separator | | | | | | | | | 0 | | 图像分隔符 | | Byte | |
| Image Left Position | | | | | | | | | 1 | | 图像左边位置 | | Unsigned | |
|  | | | | | | | | | 2 | | (以像素为定单位) | |  | |
| Image Top Position | | | | | | | | | 3 | | 图像顶部位置 | | Unsigned | |
|  | | | | | | | | | 4 | | (以像素为定单位) | | |  |
| Image Width | | | | | | | | | 5 | | 图像宽度 | | Unsigned | |
|  | | | | | | | | | 6 | | (以像素为定单位) | |  | |
| Image Height | | | | | | | | | 7 | | 图像高度 | | Unsigned | |
|  | | | | | | | | | 8 | | (以像素为定单位) | |  | |
|  | | | | | | | | | 9 | | 包装域 | | 见图6 | |

图5 图像描述块的结构

在图5中，图像分隔符(Image Separator)用来标识图像描述块的开始，该域包含固定的值：0x2C；图像左边位置(Image Left Position)是相对于逻辑屏幕(Logical Screen)最左边的列号，逻辑屏幕最左边的列好定义为0；图像顶部位置(Image Top Position) 是相对于逻辑屏幕(Logical Screen)顶部的行号，逻辑屏幕顶部的行号定义为0。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Local Color Table Flag | Interlace Flag | Sort Flag | Reserved | | Size of Local Color Table | | |

图6 图像描述块中的包装域结构

图像描述块(Image Descriptor)中的第9个字节称为包装域(Packed Fields)字节，它的位结构如图6所示，它由5个子域组成：

(1) 局部彩色表标志(Local Color Table Flag )域L用来说明是否有局部彩色表存在。如果L＝1，表示有一个局部彩色表(Local Color Table)将紧跟在这个图像描述块(Image Descriptor)之后;如果G＝0，表示图像描述块(Image Descriptor)后面没有局部彩色表(Local Color Table)，该图像要使用全局彩色表(Global Color Table)。

(2) 交插显示标志(Interlace Flag)域I用来表示该图像是不是交插图像(Interlaced Images)。如果I＝0，表示该图像不是交插图像，如果I＝1表示该图像是交插图像。使用该位标志可知道图像数据是如何存放的。GIF文件格式定义了两种数据存储方式：一种是按图像行连续顺序存储，这个顺序与显示器上显示行的顺序相同；另一种按交插方式存储。交插图像按行分成如下所示的4组(Group)：

**Group 1：**每隔8行组成一组，从第0行开始显示 /第1遍交插

**Group 2：**每隔8行组成一组，从第4行开始显示 /第2遍交插

**Group 3：**每隔4行组成一组，从第2行开始显示 /第3遍交插

**Group 4：**每隔2行组成一组，从第1行开始显示 /第4遍交插

由于显示图像需要较长的时间，使用这种方法存放和显示图像数据，人们就可以在图像显示完成之前看到这幅图像的概貌，而不觉得显示时间长。图8说明了这种交插图像的存储和显示顺序。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 行号 | 像 点 | 交插遍次 | | | |
| 0 | …………………………………… | 1 |  |  |  |
| 1 | …………………………………… |  |  |  | 4 |
| 2 | …………………………………… |  |  | 3 |  |
| 3 | …………………………………… |  |  |  | 4 |
| 4 | …………………………………… |  | 2 |  |  |
| 5 | …………………………………… |  |  |  | 4 |
| 6 | …………………………………… |  |  | 3 |  |
| 7 | …………………………………… |  |  |  | 4 |
| 8 | …………………………………… | 1 |  |  |  |
| 9 | …………………………………… |  |  |  | 4 |
| 10 | …………………………………… |  |  | 3 |  |
| 11 | …………………………………… |  |  |  | 4 |
| 12 | …………………………………… |  | 2 |  |  |
| 13 | …………………………………… |  |  |  | 4 |
| 14 | …………………………………… |  |  | 3 |  |
| 15 | …………………………………… |  |  |  | 4 |
| 16 | …………………………………… | 1 |  |  |  |
| 17 | …………………………………… |  |  |  | 4 |
| 18 | …………………………………… |  |  | 3 |  |
| 19 | …………………………………… |  |  |  | 4 |

图7 交插图像显示顺序

(3) 彩色表排序标志(Sort Flag)域的含义与全局彩色表(Global Color Table)中(Sort Flag)域的含义相同。

(4) 保留(Reserved)。

(5) 局部彩色表大小(Size of Local Color Table)域的值用来计算局部彩色表(Global Color Table)中包含的字节数。

5、局部彩色表

局部彩色表(Local Color Table)用于紧跟在它后面的图像。彩色表是否存在取决于图像描述块(Image Descriptor)中局部彩色表标志(Local Color Table Flag)位的设置。彩色表的结构和大小与全局彩色表(Global Color Table)完全相同。

6、表基图像数据

GIF图像采用了LZW算法对实际的图像数据进行压缩。为了提高压缩编码的效率，对LZW编码器输出的代码采用可变长度码VLC(variable-length-code)，不是用位数高度的代码来表示输出，而且代表码字的位数是可变的。

表基图像数据(Table Based Image Data)由LZW最小代码长度(LZW Minimum Code Size)和图像数据(Image Data)组成，如图8所示。LZW最小代码长度域的值用来确定图像数据中LZW代码使用的初始位数。图像数据(Image Data)由数据子块(Data Sub-blocks)序列组成。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  | | 域的名称 | | 类型 | |
| LZW Minimum Code Size | | | | | | | | |  | | LZW最小代码长度 | | Byte |

|  |  |  |  |
| --- | --- | --- | --- |
| Image Data |  | 图像数据 | Data Sub-blocks |

图8 图像数据的存储格式

 数据子块(Data Sub-blocks)的结构如图9所示，这是一个可变长度的数据块，其长度由块大小域(Block Size)域中的值确定，字节数在0～255之间。

|  |  |  |  |
| --- | --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 字节号 | 域的名称 | 数据类型 |
| Block Size | 0 | 块大小 | Byte |
|  | 1 |  | Byte |
|  |  |  | Byte |
| Data Values |  | 数值 | Byte |
|  |  |  | Byte |
|  | … |  | … |
|  | … |  | … |
|  |  |  | Byte |
|  | 多 |  | Byte |
|  | 到 |  | Byte |
|  | 255 |  | Byte |

图9 数据子块的结构

7、 图形控制扩展块

图形控制扩展块(Graphic Control Extension)包含处理图形描绘块时要使用的参数，它的结构如图10所示。现说明如下：

(1) 扩展导入符Extension Introducer)用于识别扩展块的开始，域中的值是一个数值等于0x21的固定值。

(2) 图形控制标签(Graphic Control Label)用于标识当前块是一个图形控制扩展块，域中的值是一个数值等于0xF9的固定值。

(3) 块大小(Block Size)用来说明该扩展块所包含字节数，该字节数是从这个块大小(Block Size)域之后到块结束符之间的字节数。

　typedef struct gifcontrol{  
　　        BYTE byBlockSize;  
　　        struct flag{  
　　              BYTE Transparency     : 1;  
　　              BYTE UserInput        : 1;  
　　              BYTE DisposalMethod : 3;  
　　              BYTE Reserved         : 3;  
　　           }    Flag;  
　　        WORD wDelayTime;  
　　        BYTE byTransparencyIndex;  
　　        BYTE byTerminator;  
　　     }    GIFCONTROL;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 字节号 | | 域的名称 | | | 类型 |
| Extension Introducer | | | | | | | | | 0 | | 扩展导入符 | Byte | |
| Graphic Control Label | | | | | | | | | 1 | | 图形扩展标签 | Byte | |
|  | | | | | | | | |  | |  |  | |
| Block Size | | | | | | | | | 0 | | 块大小 | Byte | |
| <Packed Fields> | | | | | | | | | 1 | | 包装域 | | See below |
| Delay Time | | | | | | | | | 2 | | 延时时间 | Unsigned | |
|  | | | | | | | | |  | |  |  | |
| Transparent Color Index | | | | | | | | | 3 | | 透明彩色索引 | Byte | |
|  | | | | | | | | |  | |  |  | |
| Block Terminator | | | | | | | | | 0 | | 块结束符 | Byte | |

图10 图像描述块的结构

(4) 包装域的结构如图10所示。处理方法(Disposal Method)规定图形显示之后译码器要用表2中所述方法进行处理。

|  |  |
| --- | --- |
| 域值 | 处理方法 |
| 0 | 没有指定要做任何处理 |
| 1 | 不处理，图形留在原处 |
| 2 | 显示图形的区域必须要恢复成背景颜色 |
| 3 | 恢复成以前显示的图形 |
| 4～7 | (未定义) |

表2 包装域规定的处理方法

用户输入标志(User Input Flag)域表示在继续处理之前是否需要用户输入响应。在延时时间(Delay Time)和用户输入标志(User Input Flag)都设置为1的情况下，继续处理的开始时间取决于用户响应输入在前还是延时时间结束在前。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved(保留) | | | Disposal Method(处理方法) | | | User Input Flag | Transparent Color Flag |

图11 图形控制扩展块的包装结构

(5) 透明(Transparency Flag)表示是否给出透明索引(transparency index)

(6) 延时时间(Delay Time)用来指定在图形显示之后继续处理数据流之前的等待时间，一百分之一秒为单位。

(7) 当且仅当透明标志位设置为1时，透明索引(Transparency Index)用来指示处理程序是否要修改显示设备上的相应象点。当且仅当透明标志位设置为1时，就要修改。

(8) 块结束符(Block Terminator)表示该图形控制扩展块(Graphic Control Extension)结束，它是由一个字节组成的数据块，该域的值是一个固定的值：0x00，因此称为零长度数据子块(zero-length Data Sub-block)。

8、无格式文本扩展块

无格式文本扩展块(Plain Text Extension)包含文本数据和描绘文本所须的参数。文本数据用7位的ASCII字符编码并以图形形式显示。扩展块的结构如图12所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 字节号 | 域的名称 | 数据类型 |
| Extension Introducer (0x21) | 0 | 扩展导入符 | Byte |
| Plain Text Label (0x01) | 1 | 无格式文本标签 | Byte |

|  |  |  |  |
| --- | --- | --- | --- |
| Block Size | 0 | 块大小 | Byte |
| Text Grid Left Position | 1 | 文本网格左列位置 | Unsigned |
|  | 2 |  |  |
| Text Grid Top Position | 3 | 文本网格顶行位置 | Unsigned |
|  | 4 |  |  |
| Text Grid Width | 5 | 文本网格宽度 | Unsigned |
|  | 6 |  |  |
| Text Grid Height | 7 | 文本网格高度 | Unsigned |
|  | 8 |  |  |
| Character Cell Width | 9 | 字符单元宽度 | Byte |
| Character Cell Height | 10 | 字符单元高度 | Byte |
| Text Foreground Color Index | 11 | 文本颜色索引 | Byte |
| Text Background Color Index | 12 | 文本背景颜色索引 | Byte |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Plain Text Data |  | 无格式文本数据 | Data Sub-blocks |
|  |  |  |  |

图12 无格式文本扩展块结构

9、注释扩展块

注释扩展块(Comment Extension)域的内容用来说明图形、作者或者其他任何非图形数据和控制信息的文本信息。

注释扩展块的结构如图13所示。其中的注释数据是序列数据子块(Data Sub-blocks)，每块最多255个字节，最少1个字节。

|  |  |  |  |
| --- | --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 字节号 | 域的名称 | 数据类型 |
| Extension Introducer (0x21) | 0 | 扩展导入符 | Byte |
| Comment Label (0xFE) | 1 | 注释标签 | Byte |

|  |  |  |  |
| --- | --- | --- | --- |
| Comment Data | 0 | 注释数据 |  |
|  |  |  | Data Sub-blocks |
|  | … |  |  |
|  | N－1 |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Block Terminator |  | 块结束符 | Byte |

 图13 注释扩展块

10、应用扩展块

应用扩展块(Application Extension)包含制作该图像文件的应用程序的相关信息，它的结构如图14所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 字节号 | 域的名称 | 数据类型 |
| Extension Introducer (0x21) | 0 | 扩展导入符 | Byte |
| Extension Label (0xFF) | 1 | 扩展标签 | Byte |

|  |  |  |  |
| --- | --- | --- | --- |
| Block Size | 0 | 块大小 | Byte |
|  | 1 |  |  |
|  | 2 |  |  |
|  | 3 |  |  |
| Application Identifier | 4 | 应用程序标识符 | 8 Bytes |
|  | 5 | (程序名称) |  |
|  | 6 |  |  |
|  | 7 |  |  |
|  | 8 |  |  |
|  | 9 |  |  |
| Appl. Authentication Code | 10 | 应用程序识别码 | 3 Bytes |
|  | 11 |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Application Data |  | 应用数据 | Data Sub-blocks |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Block Terminator | 0 |  | Byte |

图14应用扩展块

11、GIF文件结束块

结束块(GIF Trailer)表示GIF文件的结尾，它包含一个固定的数值：0x3B。它具有如图15所示的结构。

|  |  |  |
| --- | --- | --- |
| 7 6 5 4 3 2 1 0 | 域的名称 | 数据类型 |
| GIF Trailer = 0x3B | GFI文件结束块 | Byte |

 图15 GIF文件结束块

5.3.3速差表

表03 GIF文件格式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 块的名称 | 需要 | 标签 | 扩展 | 版本号. |
| Application Extension(应用扩展) | Opt. (\*) | 0xFF (255) | yes | 89a |
| Comment Extension(注释扩展) | Opt. (\*) | 0xFE (254) | yes | 89a |
| Global Color Table(全局彩色表) | Opt. (1) | none | no | 87a |
| Graphic Control Extension(图形控制扩展) | Opt. (\*) | 0xF9 (249) | yes | 89a |
| Header(文件头) | Req. (1) | none | no | N/A |
| Image Descriptor(图像描述) | Opt. (\*) | 0x2C (044) | no | 87a (89a) |
| Local Color Table(局部彩色表) | Opt. (\*) | none | no | 87a |
| Logical Screen Descriptor(逻辑屏幕描述块) | Req. (1) | none | no | 87a (89a) |
| Plain Text Extension(无格式文本扩展) | Opt. (\*) | 0x01 (001) | yes | 89a |
| Trailer(文件结束) | Req. (1) | 0x3B (059) | no | 87a |

Unlabeled Blocks(无标号块)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Header(文件头) | Req. (1) | none | no | N/A |
| Logical Screen Descriptor(逻辑屏幕描述块) | Req. (1) | none | no | 87a (89a) |
| Global Color Table(全局彩色表) | Opt. (1) | none | no | 87a |
| Local Color Table(局部彩色表) | Opt. (\*) | none | no | 87a |

Graphic-Rendering Blocks(图像描绘块)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Plain Text Extension(无格式文本扩展) | Opt. (\*) | 0x01 (001) | yes | 89a |
| Image Descriptor(图像描述块) | Opt. (\*) | 0x2C (044) | no | 87a (89a) |

Control Blocks(控制块)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Graphic Control Extension(图形控制扩展) | Opt. (\*) | 0xF9 (249) | yes | 89a |

Special Purpose Blocks(专用块)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trailer(结束) | Req. (1) | 0x3B (059) | no | 87a |
| Comment Extension(注释扩展) | Opt. (\*) | 0xFE (254) | yes | 89a |
| Application Extension(应用程序扩展) | Opt. (\*) | 0xFF (255) | yes | 89a |

表中：Req. (1)表示最多出现1次

Opt. (\*)出现次数大于等于0次