附录B　绘 图 结 构

本附录将说明3种绘图相关文件格式的设计信息：图标文件、位图文件和对话框模板。每种描述的前面将讲述这种文件的通常结构，随后将在每一部分着重按照字节顺序来讲解。

B.1　图标和光标

图标和光标文件使用一种公共的文件格式。单类型代码 (在文件头中)与双类型代码是不同的。图标和光标文件都可能包含多个对象的信息。从最抽象的角度看，它们由下列两部分组成。

- 文件头部分
- 图片数据部分

但是，对于文件中的所有图标 (或光标)来说，每部分都包括一个重复的结构。

B.2　文件头部分

文件头的开始几个字节说明了文件的类型以及文件中包含的对象数目。

0000	0000
0002	0001=图标，0002=光标
0004	文件中包含的图标和光标的数目

在这种“整体”信息的后面，紧跟着文件中包含的图标或光标的“目录”。

单独的图标和光标头

“目录”中的每个项目是为单个对象指定维数和颜色格式的更详细的头结构。这些项目还包括指向实际位图数据的指针。下列结构对于文件中的每个图标或光标都会重复一次。

0000	宽度　　高度
0002	2=B&W，16=10色，0=256色
0004	光标热点的x值
0006	光标热点的y值
0008	图标/光标数据的大小
000a	“
000c	图标/光标数据在文件中的偏移
000e	“

B.3　图片数据部分

文件的这个部分对于每个图标或光标都有一个独立的位图数据。每个对象的图片数据由

三部分结构组成：头、颜色表和一个像素值表。这三部分结构对文件中的每个图标和光标都会重复一次。

B.3.1 图片数据头

这个40字节的头说明了对象的图片特征：大小、形状以及彩色平面的数目。

0000	头大小(40字节)
0002	
0004	图标/光标宽度
0006	
0008	高度*2
000a	
000c	平面数
000e	位数据像素指针的大小
0010	
0012	
0014	第1个平面中的字节数
0016	
0018	第2个平面中的字节数
001a	
001c	第3个平面中的字节数
001e	
0020	第4个平面中的字节数
0022	
0024	第5个平面中的字节数
0026	

B.3.2 颜色表

仅当对象不是单色时，才有颜色表。

DWORD RGB数据 每种颜色一个项目

B.3.3 位数据

在彩色图像中，表中的每个项目对应颜色表中的一个 RGB值。在单色图像中，每个项目的值可以为0或1(黑或白)。

	在颜色表中的像素指针 每一个像素对应一项 如果是单色，则只是0或1

B.4 注意

• 可以用图标或光标编辑器(Icon or Cursor Editor)为应用程序创建图标或者光标。在创建新图标或者新光标时，这些编辑器创建缺省为 32×32像素的16色图标或者32×32像素的单色光标。要创建其他类型的图标和光标，可单击 Developer Studio的Image和New Device Image…菜单命令以打开New Image对话框。如果要创建的图标或者光标还没有被列出，则单 Custom按钮创建自己的图标或者光标。用这个办法可以创建彩色光标。在创建了图标或者光标之后，可以删除缺省的图标或者光标，做法是：在编辑器的组合框内选中缺省图标或者光标，然后单击Studio的Image和Delete Device Image菜单命令。

• 当前还没有Windows API或者MFC类函数可将文件中的数据直接转变为图标或者光标。不过可以创建一个内存文件，以此提供给相关的图标或者光标装载函数使用。

B.5 位图文件

位图(.bmp)文件的结构类似于图标或光标文件中的位图部分。通过最抽象的观点看，每个位图文件由下列四部分构成。

- 文件头
- 位图头
- 颜色表
- 图片数据

B.5.1 文件头

文件头结构(BITMAPFILEHEADER的结构类型)包括一个文件类型标记(“BM”)以及文件大小、布局信息。

0000	“BM”
0002	文件字节大小
0004	“
0006	0000
0008	0000
0010	图片数据在该文件中的偏移量
0012	“

B.5.2 位图头

文件头的后面是位图信息头 (BITMANINFOHEADER的结构类型)。该结构指定了维数、压缩类型以及图像的颜色格式。

0000	位图头字节大小(40)
0002	
0004	位图宽度像素值
0008	位图高度像素值
0010	平面数，必须为1
0012	每个像素的位数 (参见B.5.5)
0014	压缩类型
	0=无
0016	压缩后的图像大小
0018	水平像素/米
001c	垂直像素/米
0020	颜色表中使用的颜色(参见B.5.5)
0024	颜色表中的重要颜色

B.5.3 颜色表

根据可用颜色数目的不同 (参见B.5.5)，颜色表的大小也不同。使用 24位色时，将省略颜色表，因为图片数据的每个项目将显式地解码 RGB值。

DWORD RGB 数据 每一种颜色对应一项

B.5.4 图片数据

图片数据是一个像素值表(可能是压缩的)。根据所使用的颜色方案的不同，解压缩的像素值将进行不同的解释。

- 如果是单色，数据为二进制位序列，每一个像素对应一位， 0=黑色、1=白色。
- 如果是16色或256色，数据为颜色表中4或8位的指针序列，每一个像素对应一个指针。
- 如果是高彩色或真彩色，数据为24位RGB值序列，每一个RGB值定义了一种像素颜色。

B.5.5 注意

位图头包含了一个“ 每个像素的位数 (bits per pixel) ”值，该值确定颜色表的大小以及图像数据代表什么，该值可以是1、4、8和24，含义如下：

- 如果该值为1，这就是单色位图。没有颜色表，图像数据只是一系列的 0和1，前者代表黑色，后者代表白色。
- 如果该值为4或者8，这就是16或者256色位图。16色位图的颜色表是16双字长，256色位图的颜色表则是 256双字长。但如果位图头中“ #颜色表中使用的颜色 (of colors used in Color Table) ”这一项的值不是0，那么该值将确定颜色表内应该具有多少项。
- 如果该值是24，这就是真彩色位图。因为图像数据本身包含了一系列完整的 RGB数值定

义位图内每个像素的颜色，所以位图内就没有颜色表。

- 有关位图和其他文件格式的详细信息，请参阅 http://www.wotsit.org站点。

B.6 对话框模板

对话框模板包括一系列对话框信息，后跟多个部分的控制信息，其中每个控件包含一个控制信息。对话框信息存放在 DLGTEMPLATE结构中，后跟可选的 FONTINFO结构。控制信息存放在DLGITEMTEMPLATE结构中。模板具有下列的一般结构。

- 对话框信息
- 头
- 菜单
- 类
- 标题
- 字体
- 控制信息
- 控件
- : : :
- 控件n

B.6.1 对话框信息

尽管对话框信息大多都包含在一个 DLGTEMPLATE结构中，但它对查看包括菜单、类、标题和字体信息的对话框信息来说是很有用的。

1. 头

对话框的风格、位置和大小都存放在 DLGTEMPLATE结构的头部分中。模板的这部分还包括对话框中的控件数目。

0000	风格
0002	
0004	扩展风格
0006	
0008	控件数
0010	对话框单元(DU)的x位置
0012	DU的y位置
0014	DU的宽度
0016	DU的高度

2. 菜单

尽管菜单信息仅仅只是 DLGTEMPLATE结构的一个字段，但这个字段对真正的 C结构来说还是比较复杂的。有三种可互换的形式：

如果没有则为0
或
-1

菜单资源的ordinal(参见B.7.1)
或
菜单资源的unicode名
0

3. 类

类信息使用一个相似的变换形式存放。并且，它也有三种可替换的形式：

如果使用默认值则为0
或
-1
类名的ordinal(参见B.7.1)
或
类名的unicode名
0

4. 标题

如果对话框有标题栏，该字段则必须包括此标题。如果没有标题栏，该字段则为空。

如果没有则为0
或
unicode标题
0

5. 字体

字体信息存放在一个单独的 FONTINFO结构中。仅当在对话框的风格参数中设置了DS_FONT后，才有此结构。

指针大小 unicode字体名

B.7 控制信息

控制信息是一个顺序的结构，每个控件一个。每个结构具有下列形式。

- 头
- 类
- 标题
- 创建数据

头、类和标题信息存放在一个 DLGITEMTEMPLATE结构中。创建数据对每个控件类来说是唯一的。

1. 头

DLGITEMTEMPLATE结构的几个字段构成 DLGTEMPLATE结构的头信息。这些字段指定风格、位置、大小和控件的 ID。

0000	风格
0002	
0004	扩展风格

0006	
0008	对话框单元(DU)的x位置
0010	DU的y位置
0012	DU的宽度
0014	DU的高度
0016	控件ID

2. 类

该字段指定控件的类，可以作为 ordinal，也可以作为可变长度的字符串。可以使用的格式为：

-1
类名的ordinal

其中：

80 = “ BUTTON ”
81 = “ EDIT ”
82 = “ STATIC ”
83 = “ LISTBOX ”
84 = “ SCROLLBAR ”
85 = “ COMBOBOX ”

或

类名的unicode ：：：
0

3. 标题

控件可以使用可变长度的名称或一个 ordinal标识。可使用下列形式：

如果没有则为0

—1
资源的ordinal

可以是字符串、图标或位图

或

unicode标题 ：：：
0

4. 创建数据

如果无，则为0，否则为数据的字节数
对每个控件都不同，创建控件时通过 CreateEX()传递。

注意

• Ordinal是一个内部指针系统，用于资源编辑器以最小空间访问资源 (字符串、位图和图

标等)。由于这些指针在内部分配，所以如果要自行对它们进行解释将会遇到麻烦。

• 用上面的结构可以动态地创建、编辑或者组合对话框模板。为了装载已有的模板供编辑，使用如下代码：

```
HRSRC hResource = ::FindResource( AfxGetInstanceHandle(),
    lpszTemplateName, RT_DIALOG );
HGLOBAL hTemplate = LoadResource( AfxGetInstanceHandle(),
    hResource );
DLGTEMPLATE *pTemplate = ( DLGTEMPLATE* )LockResource( hTemplate );
```

pTmplate指针现在指向对话框模板，它可以由 memcpy()函数来操作，然后由下面修改的模板创建一个对话框，使用如下代码：

```
CreateDlgIndirect( pTemplate, pWndOwner, AfxGetInstanceHandle() );
// also run extra creation data
ExecuteDlgInit( MAKEINTRESOURCE( IDD ) );
```

确保使用以下代码完成清除工作：

```
UnlockResource( hTemplate );
FreeResource( hTemplate );
```