

高质量 PDF417 条码应用开发 SDK

1、PDF417Enc.dll 条码应用开发库简单介绍

PDF417Enc.dll 大约 **30k** 左右, 其中还包含版本信息等等辅助数据。**PDF417Enc.dll** 用 **20k** 左右的程序代码, 完成了 **PDF417** 二维条码的编码和绘制功能。**PDF417Enc.dll** 代码编写是非常精练的, 且是非常高效的, 运行速度也是首屈一指的。其他的 **PDF417** 条码应用开发组件, 大多以位图输出的形式给其他应用软件提供接口, 这很不便于用户机动编程, 应用灵活性小。再者, 以位图为接口数据的情况下, 是不利于大批量打印条码的, 原因很简单, 一张 **600dpi** 的单色条形码位图, 数据量就有几百 **k** 大, 而一张 **600dpi** 的彩色条码位图, 则可能有十几 **M** 大, 一张 **1440dpi** 的条形码的彩色位图, 则可能在 **50M** 以上! 如果应用软件需要批量制作 **10000** 张条形码标签, 采用位图为接口数据的话, 总数据量可能高达数百 **G**, 可想象, 要在计算机上生成数百 **G** 的位图数据, 并把这数百 **G** 的位图数据转换成打印数据再传送到打印机, 是一项多么艰巨、费时和消耗资源的任务! **PDF417Enc.dll** 则以矢量矩阵数据为接口, 一张 **2880dpi** 甚至更高分辨率的条码, 不论是彩色还是单色, 数据量都在几十 **k** 左右! 若使用 **PDF417Enc.dll** 输出 **10000** 张 **PDF417** 二维条形码, 总数据量不到 **1G**! 所以, 对于大批量、高精度、高速度打印条码的应用, **PDF417Enc.dll** 是有得天独厚的优势。

2、PDF417Enc.dll 条形码应用开发库的使用特性

PDF417Enc.dll 的应用非常简单, 因为 **PDF417Enc.dll** 只有一个主函数 **DrawPDF417**。但请应用者注意: **PDF417Enc.dll** 是基于 **Unicode** 的!

3、PDF417Enc.dll 的一个函数 DrawPDF417 的声明

VC++ 声明:

```
DWORD DrawPDF417 (HDC DC, WCHAR * Data, RECT * Bounds, RECT * BarMatrix, int BarWidth,  
    int BarHeight, int BarRows, int BarCols, int TextWidth, DWORD Color, DWORD Params);
```

Delphi 声明:

```
function DrawPDF417 (DC :HDC; Data : PWideChar; Bounds :PRect; BarMatrix :PRect;  
    BarWidth, BarHeight, BarRows, BarCols : Integer; Color, Params : DWord) :DWord;  
stdcall; external 'PDF417Enc.dll' name 'DrawPDF417';
```

DrawPDF417 参数的意义说明如下:

DC: 设备环境。要把条码绘制在窗口, 传递窗口的 **DC** 即可, 要打印条码, 传递打印机的 **DC** 即可。有时可传递一个 **0** 参数, 此时不绘制条码, 仅用于快速获得一些特定的数据, 下面会说明。

Bounds: 条码绘制在 **DC** 上的位置和尺寸, 也就是条码的绑定矩形; 该参数为一个 **Rect** 指针, 比如想把条码绘制在 **Rect(10, 10, 100, 100)** 的空间内, 则可照此初始化 **Bounds**。但 **PDF417** 条码的 **Bar** 的粗细是步进的且 **Bar** 的数量一定是 **17** 的整数倍, 所以大多数时候并不能恰好绘制在指定的空间内。在 **DrawPDF417** 函数返回时, **Bounds** 会被更新, 更新后的数据即绘制时实际的位置和尺寸。所以, 在应用时可先调用 **DrawPDF417**——**DC** 设为 **0**——计算出实际绘制的空间, 然后再绘制即可保证准确无误, 也就是先测量后绘制的模式。

BarMatrix: 这是一个 **Rect** 数组的指针, 用于存放 **PDF417** 条码的矩阵数据; 若不需要矩阵数据, 可传递一个 **NULL** 参数。若需要该数据, 在 **BarMatrix** 数组初始化时, 最好分配 **10000** 个 **Rect** 结构的空間, 因为 **PDF417** 条码的 **Bar** 矢量化为矩形后, 矩形的数量可能是巨大的。前面说到, 可调用 **DrawPDF417** 获得 **Bounds**, 也可调用 **DrawPDF417**——**DC** 参数设为 **0**——同时获得 **Bounds** 和 **BarMatrix**, 然后根据 **Bounds** 定位条码位置, 使用 **FillRect** 填充 **BarMatrix** 中的每一个矩形即可快速完成绘制, 而不必二次调用 **DrawPDF417** 进行绘制, 因为已经获得了绘制 **PDF417** 的条件 (参见后文的说明)。

注: 获得了 **Bounds** 和 **BarMatrix** 这两个数据, 屏幕上缩放 **PDF417** 条码, 只要缩放 **Bounds** 和 **BarMatrix** 中

的每一个矩形,而不必在缩放时调用 **DrawPDF417**。在图形重画时,我们也不必每次都调用 **DrawPDF417**,只需应用现成的 **Bounds** 和 **BarMatrix** 这两个数据,重画即可。这保证了极快的缩放速度和重画速度。

BarWidth: **PDF417** 由很多个 **Bar** 组成;这里的 **BarWidth** 指每个 **Bar** 的宽度,而非条码整体的宽度。

BarHeight: 条码每行 **Bar** 的高度。当设定了 **BarHeight** 为 **BarWidth** 的倍率 **Aspect**(见后文说明)大于 0 的话,**BarHeight** 参数就会忽略。

注: **PDF417** 条码的整体尺寸由 **DrawPDF417** 返回后的 **Bounds** 确定。**Bounds** 确定 **PDF417** 的定位和尺寸。

BarRows: **PDF417** 条码的 **Bar** 可分行绘制, **BarRows** 这个参数就是行数; 若为 0, 则自动分行。

BarCols: **PDF417** 条码的 **Bar** 可分列绘制, **BarCols** 这个参数就是列数; 若为 0, 则自动分列。

注: 若人工设定行数或列数,若行数或列数过小,可能产生一个错误,因为行数、列数不够编码 **PDF417** 条码的数据,也就是信息量超过了当前的行数、列数所能编码的最大信息量。

Color: 条码的绘制颜色。

Params: 这是 **DrawPDF417** 中最复杂的参数,也是一个复合型参数。**Params** 是一个 **DWORD** 参数,一个 **DWORD** 有 4 字节共 32 bit,我们把 **Params** 的各 bit 从低到高编号为 0 到 31。

0 bit – 7 bit: 保留未用;

8 bit – 9 bit: 条码旋转方式(取值为 0 – 3: 这里的取值 0 – 3 指 9bit、8bit 为 00、01、10、11, 以下同)。

0: 不旋转(水平样式)

1: 逆时针旋转 90 度

2: 顺时针旋转 90 度

3: 旋转 180 度

10 bit – 11 bit: 保留未用。

12 bit – 15 bit: **PDF417** 的纠错等级(取值为 0 – 9)。为 0 则表示自动,为 1 – 9 依次为 0 级 – 8 级纠错。

注: 纠错级别越高,需要的条码面积更大,所以条码需要的行、列数会更多。所以设置纠错级别时,可能会产生一个错误,此时增大 **BarRows**、**BarCols** 即可解决。

16 bit – 19 bit: **BarHeight** 相对 **BarWidth** 的倍率 **Aspect**(取值为 0 – 15),也就是 $\text{BarHeight} = \text{Aspect} * \text{BarWidth}$,当 $\text{Aspect} > 0$ 时,忽略传递给 **DrawPDF417** 的 **BarHeight** 参数。

20 bit: **PDF417** 是否支持扩展语法。为 1 则支持,为 0 则不支持;

21 bit: **PDF417** 是否截短。为 1 则截短,为 0 则完整绘制;**PDF417** 截短码可缩小 **PDF417** 条码的面积。

22 bit: **PDF417** 是否以最大高度绘制。当用户初始化 **Bounds** 时, $\text{Bounds.Bottom} - \text{Bounds.Top}$ 即为初始化高度。为 1 则尽可能保证整体高度为初始化高度,为 0 则自动设置一个合理的高度;

23 bit: **PDF417** 是否支持 **GLI** 标志。为 1 则支持,为 0 则不支持。**GLI** 标志是一些特殊意义的字符。

24 bit – 31bit: 保留未用。

DrawPDF417 的返回值: **DrawPDF417** 的返回值是一个 **DWORD** 类型,复合了两个数据:低 16 bit 是条码矢量矩阵 **BarMatrix** 中矩形的数量,而高 16 bit,则是 **DrawPDF417** 执行结果(为 0 则表示执行成功,大于 0 则表示出错的错误号,见下面的说明)。利用返回值,用户可知道 **BarMatrix** 中,有多少个矩形需要处理,也可知道执行是否出错,出了什么错。假设 **R** 是 **DrawPDF417** 的返回值,**X**、**Y** 是两个变量,用于存放 **PDF417** 条码矩阵 **BarMatrix** 中矩形的个数和 **DrawPDF417** 的执行结果和,那么:

BarMatrix 中矩形个数: $X = R$ 且 $0x0000FFFF$

我们可以用如下的简单语句自己绘制 **PDF417** 条码:

```
int i;
```

```
for (i = 0; i < X; i++)
```

```
    FillRect(DC, BarMatrix[i], Brush);
```

DrawPDF417 的执行结果: $Y = R$ 右移 16。

若 $Y = 0$: 执行成功

若 **Y = 1**: **PDF417** 条码的行列数不够编码 **PDF417** 条码的现有数据。

若 **Y = 2**: 存在错误的 **GLI** 标志。

若 **Y = 3**: 编码时出现错误的码字。该错误一般不会出现。

4、使用 **PDF417Enc.dll** 易犯的错误

PDF417Enc.dll 是基于 **Unicode** 的, 在 **VC++** 中使用, 所有字符串相关的函数参数, 应使用 **WCAHR**, 而不能是 **char**。而在 **Delphi** 中使用, 所有字符串相关的函数参数, 应使用 **PWideChar**, 而不能是 **PAnsiChar**! 随着 **Windows 98** 的消失, 基于 **AnsiChar** 的应用程序, 已经越来越没有了市场。

5、试用 **PDF417Enc.dll**

在作者编写 **PDF417Enc.dll** 时, 下载了数个 **PDF417** 条码控件、软件和几个 **PDF417** 解码软件进行测试, 发现一个奇怪的问题: 各个 **PDF417** 条码控件 (或软件) 生成的 **PDF417** 条码, 可能某个 **PDF417** 解码软件可正常解码, 而另一个解码软件却无法解码。为此, 我查阅大量的 **PDF417** 标准和规范, 充分完善了 **PDF417Enc.dll** 的代码, 生成的 **PDF417** 条码, 可被我下载的所有 **PDF417** 解码软件快速、准确无误地解码。 **PDF417Enc.dll** 有那些特性呢?

- 1、文本压缩方式、数字压缩方式、字节压缩方式, 根据实际数据的组成, **PDF417Enc.dll** 自动分段选择最优的压缩方式, 保证最大的数据存储量 (很多 **PDF417** 控件采用的是单纯的字节压缩 (即二进制压缩) 方式, 这样编码算法最为简单, 但信息容量最小)。
- 2、完全 **Unicode** 代码, 支持 **Unicode** 信息和非 **Unicode** 信息的混合编码;
- 3、支持多核多线程应用, 支持 **Vista** 和 **Windows 7**, 保证相对长远的应用需求。

注: 因为 **PDF417** 条码的生成与编码算法密切相关, 所以不同的 **PDF417** 控件或软件, 即使是一模一样的数据, 所生成的 **PDF417** 条码也可能是完全不同的。

在准备使用 **PDF417Enc.dll** 时, 请参考作者编写的 **Free2DBarcode** 软件, 这个软件提供了 **PDF417** 条码的多种输出方式: 位图文件输出、**EMF** 矢量文件输出、矢量矩阵数据直接输出到剪贴板、打印输出。尤其, 矢量矩阵数据输出到剪贴板后, 你可把条码的矢量数据, 直接粘贴到 **CorelDraw**、**Illustrator**、**Word**、**Excel**、**WPS** 等常见的应用软件中, 并保持其矢量特性。**Free2DBarcode** 就是基于 **PDF417Enc.dll** 开发的, 是完全免费的, 且支持 **Vista** 和 **Windows 7**。在试用 **PDF417Enc.dll** 时, 你必需创建一个名为 **Free2DBarcode** 的工程 (试用时必须使用该工程名, 否则你的程序不能加载 **PDF417Enc.dll**), 再把 **PDF417Enc.dll** 拷贝到你的工程目录下, 这样就可试用 **PDF417Enc.dll** 了。

PDF417Enc.dll 可在 **VC** 和 **Delphi**、**VB**、易语言、**PB** 等编程语言里使用, 因为 **PDF417Enc.dll** 是标准的 **API** 封装, 而所有编程语言都是支持调用 **API** 的。

6、购买 **PDF417Enc.dll** 授权

PDF417Enc.dll 的价格为人民币 **500** 元/授权。这里所说的每个授权, 是指授权在一个指定的应用程序 (比如 **MyApp.exe**) 中使用, 也就是说, 使用 **PDF417Enc.dll** 的应用程序不论发售了多少份, 都只需购买一个授权。现在不少组件开发商所谓的授权, 是指装机量, 比方 **A** 软件使用 **B** 组件, **A** 软件的作者每卖出一份软件, 应支付 **B** 组件的作者一份授权费。所以, 当你看见 **2000** 元/**50** 个授权 (也就是最少 **50** 个授权起卖) 时, 应区分开来, 因为这是完全不同的授权方式。而 **PDF417Enc.dll** 的授权, 是一个应用程序, 无论发售多少份, 都只需购买一个授权。

注: 未经授权请勿擅自在您的软件中使用 **PDF417Enc.dll**, 除非您的软件只是编写给自己使用的。若您的软件是商业软件或者是共享软件, 请自觉购买 **PDF417Enc.dll** 的授权。

(完)

开发者: 李辉宇

电 话: 135 8886 7730

QQ : 1497 96232

网 站: <http://www.3wcad.com>