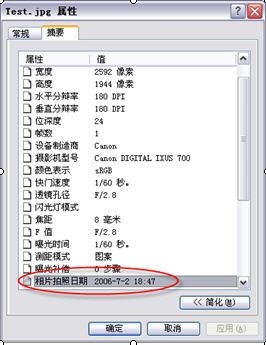
利用C#为数码照片添加拍照日期

<http://www.cnblogs.com/Risen/archive/2008/01/15/1039733.html>

继上一篇”利用C#实现任务栏通知窗口”后继续我们的C#实战演练。现在人们具备一台数码相机已经不是什么新鲜事了，更何况500万像素以上的数码相机更是逐渐成了主流。相比较以前以胶卷为感光介质的普通相机，数码相机可以将所照图像即刻转换成计算机可识别的图像文件格式以便浏览、共享和打印。虽然数码相机在技术和方便性上都远远高于普通相机，但是笔者发现所有已经生成的图像文件以及打印的数码照片上都没有拍照日期，若想在日后拍照时加上该选项也不是容易事，翻遍了数码相机的说明书竟然没有关于怎样在照片上显示拍照日期的帮助。而且，随着数码图像文件的不断增加，面对成本成本的相册要想回顾一下其到底是在何时拍照的将会非常困难，尤其像笔者这样不辞辛苦的记录小儿生长历程的朋友更感觉如此。虽然在桌面电脑上利用我的电脑浏览所拍照的图像文件时，在窗口底部任务栏上或者图像文件属性窗口的摘要页都可以清楚地显示出拍照日期，但是要想将拍照日期绘制到图像上且能够打印到数码照片上却没有工具可以做到。其实，拍照日期已经保存在了图像文件里，我们需要自己动手编程获取拍照日期并在图像的右下角将其绘制出来，然后保存新生成的图像文件并拷贝到数码照片打印店进行打印，我们就可以获得具备拍照日期的数码照片了，如下图：  


一、简介

目前大部分数码相机都将所拍照的图像保存成JPG格式，而像拍照日期这样的信息统称为EXIF信息。EXIF是英文ExchangeableImageFile(可交换图像文件)的缩写，最初由日本电子工业发展协会(JEIDA--JapanElectronicIndustryDevelopmentAssociation)制订，当时JEITA决定为数码相机厂商专门制定一套标准，随着数码相机的发展，其普及趋势越来越明显，于是JEITA对Exif标准进行了升级，目前最新版本为2.2。其实EXIF也是一种图像文件格式，EXIF信息就是由数码相机在拍摄过程中采集一系列的信息，然后把信息放置在JPG文件或者TIFF文件的头部，也就是说EXIF信息是镶嵌在图像文件格式内的一组拍摄参数，主要包括摄影时的光圈、快门、ISO、拍照日期时间等各种与当时摄影条件相关的信息、相机品牌型号、色彩编码，甚至还包括拍摄时录制的声音以及全球定位系统（GPS）等信息。简单的说，它就好像是傻瓜相机的日期打印功能一样，只不过EXIF信息所记录的资讯更为详尽和完备。需要注意的是，用图像处理软件编辑过的数码图像文件有可能会丢失其EXIF信息。

所以，要想在图像上绘制拍照日期首先必须读取EXIF信息，然后将读取出来的拍照日期绘制在图像表面，我们将以500万像素分辨率为2592x1944的JPG图像文件为对象，使用Visual Studio .Net 2005 的C#来编写一工具程序来实现上述功能。

二、 技术背景

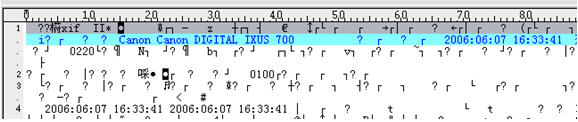
    EXIF信息以键值对的方式保存在数码JPG图像文件的头部，在.Net平台中所有图像文件头部信息统称为元数据，我们可以使用GDI+读取现有的元数据，也可以将新的元数据写入图像文件中。GDI+ 将单独的元数据段存储在 PropertyItem 对象中，读取 Image 对象的 PropertyItems 属性以便从某个文件中检索所有的元数据。PropertyItems 属性返回一个 PropertyItem 对象的数组。PropertyItem 对象具有以下四个属性：Id、Value、Len 和Type。Id用于标识元数据项的标记，下表显示一些Id 的值：

|  |  |
| --- | --- |
| 十六进制值 | 说明 |
| 0x0320  0x010F  0x0110  0x0132  0x829A  0x5090  .... | 图像标题  设备制造商  设备型号  拍照时间  Exif 曝光时间  亮度表  .... |

Value为数组值，这些值的格式由 Type 属性确定。Len属性指向的值的数组长度（以字节表示）。Type属性指向数组中值的数据类型。下表显示由Type 属性值指示的格式：

|  |  |
| --- | --- |
| **数值** | **说明** |
| 1 | 一个 Byte |
| 2 | ASCII 编码的 Byte 对象的数组 |
| 3 | 16 位整数 |
| 4 | 32 位整数 |
| 5 | 包含两个表示有理数的 Byte 对象的数组 |
| 6 | 未使用 |
| 7 | 未定义 |
| 8 | 未使用 |
| 9 | SLong |
| 10 | SRational |

我们所感兴趣的ID值就是0x0132即图片拍照时间，对应的标记为PropertyTagDateTime，而在联机的MSDN中我们发现了更详细的关于EXIF属性的GDI+的描述，PropertyTagDateTime值的类型为PropertyTagTypeASCII，它以ASCII编码的形式保存数据，我们在获取数据后就按照ASCII进行解码，将一些列字节转换为日期/时间的字符串。

在进行下一步之前，我们可以先用文本编辑软件如UltraEdit打开要操作的图片文件实际看看头文件到底是怎样的，如下图：  


我们发现里面的日期格式为：2006:06:07 16:33:41，这个格式既不是标准的日期/时间格式也不是当前系统设置的格式，所以还需要对日期/时间格式进行格式化。

获得了拍照日期/时间后，从指定的图片文件来创建Graphics对象，在该Graphics对象上绘制先前我们获取的拍照日期/时间。

三、程序实现

     启动Visual Studio .Net 2005 创建名为PicStamp的Visual C# 项目，选择Windows 应用程序模版。在默认的窗体上放置一个listBox组件用于保存需要绘制拍照日期的图片文件列表，一个textBox组件用于设置绘制后的图片文件所放置的文件夹，五个Button组件，分别用于向listBox添加图像文件、清空列表框、选择放置绘制后的图片的文件夹、实际绘制操作以及退出示例程序，一个选择文件对话框用于挑选图片文件，一个选择文件夹对话框用于选择图片文件要放置的文件夹，程序运行界面如下：  
  
  
我们自定义一个函数GetExifProperties用于返回图片文件的Exif信息，代码如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.gif//获取图像文件的所有元数据属性，以PropertyItem数组的格式保存  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifpublic static PropertyItem[] GetExifProperties(string fileName)  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif      FileStream stream = new FileStream(fileName, FileMode.Open, FileAccess.Read);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif      //通过指定的数据流来创建Image  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif      System.Drawing.Image image = System.Drawing.Image.FromStream(stream,true,false);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif      return image.PropertyItems;  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

获得所有元数据后，需要挑选出我们所感兴趣的拍照日期/时间属性所对应的值，代码如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.gif//遍历所有元数据，获取拍照日期/时间  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifprivate string GetTakePicDateTime(System.Drawing.Imaging.PropertyItem[] parr)  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     Encoding ascii = Encoding.ASCII ;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //遍历图像文件元数据，检索所有属性  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     foreach (System.Drawing.Imaging.PropertyItem p in parr)  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif     {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif          //如果是PropertyTagDateTime，则返回该属性所对应的值  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif          if (p.Id==0x0132)  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif          {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif               return ascii.GetString(p.Value);  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif          }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif     }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //若没有相关的EXIF信息则返回N/A  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     return "N/A";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

循环处理图片文件列表框中的文件，并重新格式化获取的拍照日期/时间，然后通过Graphics对象将其绘制到数码图像的表面并保存为新文件，代码如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.giffor (int i = 0; i < listBox1.Items.Count; i++)  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     pi = GetExifProperties(listBox1.Items[i].ToString() );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //获取元数据中的拍照日期时间，以字符串形式保存  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     TakePicDateTime = GetTakePicDateTime(pi);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //分析字符串分别保存拍照日期和时间的标准格式  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     SpaceLocation = TakePicDateTime.IndexOf(" ");  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     dt = TakePicDateTime.Substring(0, SpaceLocation);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     dt=dt.Replace(":", "-");  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     tm = TakePicDateTime.Substring(SpaceLocation+1, TakePicDateTime.Length - SpaceLocation-2);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     TakePicDateTime = dt + " " + tm;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //由列表中的文件创建内存位图对象  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     Pic = new Bitmap(listBox1.Items[i].ToString());  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //由位图对象创建Graphics对象的实例  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     g = Graphics.FromImage(Pic);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //在Graphics表面绘制数码照片的日期/时间戳  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     g.DrawString(TakePicDateTime, normalContentFont, new SolidBrush(normalContentColor),     
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gifPic.Width - 700, Pic.Height - 200);  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //将添加日期/时间戳后的图像进行保存  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     Pic.Save(textBox1.Text + Path.GetFileName(listBox1.Items[i].ToString()));  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     //释放内存位图对象  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif     Pic.Dispose();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

四、总结

该程序在Visual Studio .Net 2005 C# + Windows XP SP2下运行成功。通过实际使用该程序可以批量且有效地将数码图片拍照日期/时间绘制到图像表面，我们是以分辨率为2592x1944的JPG图像文件为绘制对象，读者可以根据实际图片尺寸适当调整源码中拍照日期/时间的字体、大小以及位置。本文仅演示了如何读取EXIF信息，读者可以稍加改动就可以修改EXIF信息并加以保存。还有需要注意的是，正像本文开头所提到的，任何图像编辑软件对数码照片的编辑都有可能使EXIF信息丢失，本文示例程序也不例外，经过绘制后的数码图片确实会丢失一些EXIF信息，但是所有关键信息并没有丢失。