Vážení zákazníci,

dovolujeme si Vás upozornit, že na tuto ukázku knihy se vztahují autorská práva, tzv. copyright.

To znamená, že ukázka má sloužit výhradně pro osobní potřebu potenciálního kupujícího (aby čtenář viděl, jakým způsobem je titul zpracován a mohl se také podle tohoto, jako jednoho z parametrů, rozhodnout, zda titul koupí či ne).

Z toho vyplývá, že není dovoleno tuto ukázku jakýmkoliv způsobem dále šířit, veřejně či neveřejně např. umisťováním na datová média, na jiné internetové stránky (ani prostřednictvím odkazů) apod.

redakce nakladatelství BEN – technická literatura
redakce@ben.cz

3.1 Získání rychlého náhledu obrázku

Třída *Bitmap* obsahuje členskou funkci *GetThumbnailImage*, kterou jsem také použil v našem ukázkovém projektu při vykreslování náhledu obrázku do uživatelsky kresleného prvku *Static*. Některé grafické soubory mohou v sobě obsahovat vedle vlastního obrázku ještě jeho kopii v malém rozlišení, tedy jeho náhled. Členská funkce *GetThumbnailImage* nám vrátí ukazatel (typu *Image** – *Image* jak již víme je předek třídy *Bitmap*) na obrázek vytvořený právě z tohoto malého náhledu. Samozřejmě pokud tento náhled není v obrázku obsažen, načte celý velký obrázek. V prvním případě získáme náhled rychleji, což může být významné především v případě vykreslování nějaké galerie mnoha náhledů.

Avšak toto rychlé získání má svůj háček. V některých případech v závislosti na požadované velikosti náhledu (první dva parametry funkce) a na existenci náhledu v souboru dojde při prvním překreslení k znatelně zhoršené kvalitě nakresleného obrázku. Při druhém a dalším překreslení je pak již výsledek v plné kvalitě. Proto jsem také záměrně na konci funkce *OnOtevrit* umístil volání *RedrawWindow* (vyvolá překreslení okna s náhledem) dvakrát bezprostředně za sebou. Podívejme se, jak bude vypadat okno se stejnou fotografií při svém prvním překreslení, pokud funkci *RedrawWindow* zavoláme jen jednou:

Obrázek 7 Ukázka kvality prvního rychlého náhledu

Samozřejmě i při této variantě programu dosáhneme vysoké kvality náhledu tím, že takto zobrazené okno překryjeme jiným oknem a poté opět zobrazíme, čímž dosáhneme kýženého druhého překreslení okna nyní již v plné kvalitě.

Důvodem je pravděpodobně to, že při prvním volání z důvodu maximální rychlosti dokončení funkce tato vrátí obrázek v malém rozlišení (mělo by to být v rozměru do 120 × 120 bodů), který při roztažení do našeho náhledu je větší než 120 bodů samozřejmě ztratí na kvalitě. Při dalším zavolání z téhož obrázku (instance třídy Bitmap) pak již funkce vrátí obrázek rozměrů, které zadáme v 1. a 2. parametru.

3.2 Úpravy obrázku

Dosud jsme se zabývali načtením, zobrazením a uložením obrázku do souboru. Nyní se budeme věnovat možnostem úpravy obrázku. Knihovna GDI+ nám pro tento účel dává nástroj, který je výkonný a rychlý a současně jednoduchý na použití. Umožňuje nám totiž získat pole bytů představující RGB složky obrázku a to nezávisle na grafickém formátu souboru, ze kterého byl obrázek načten.

Zmíněným nástrojem je třída *BitmapData* a dvě členské funkce třídy *Bitmap*: *LockBits* a *UnlockBits*.

Třída *BitmapData* je „čistě datová" a najdeme ji v hlavičkovém souboru *gdiplusimaging.h*. Vypadá takto:

```
class BitmapData
{
public:
    UINT Width;
    UINT Height;
    INT Stride;
    PixelFormat PixelFormat;
    VOID* Scan0;
    UINT_PTR Reserved;
};
```

Prvek *Scan0* představuje ukazatel na zmíněné pole bytů představujících data bitmapy. Pokud víme kolik bytů připadá na jeden pixel a jak jsou v nich uložené jednotlivé barevné složky, je čtení a změna těchto dat jen otázkou matematiky. Formát těchto dat si můžeme sami zvolit příslušným parametrem členské funkce *LockBits*. A jak jsme již naznačili, snadno takto získáme například pole RGB hodnot i pro obrázek načtený ze souboru ve formátu JPEG, který přímo žádné takovéto bytové pole neobsahuje. Bez pomocí GDI+ bychom museli psát nějaký vlastní parser JPEG formátu (což rozhodně není jednoduchá záležitost) nebo použít nějakou knihovnu třetí strany.

Možné hodnoty parametru *PixelFormat* jsou definovány v hlavičkovém souboru gdipluspixelformats.h. Typ *PixelFormat* je celé číslo, jak vidíme z jeho definice:

```
typedef INT PixelFormat;
```

Ve většině případů budeme požadovat data ve formátu RGB nebo ARGB, to znamená 24 resp. 32 bitů na pixel. Formát RGB obsahuje „pouze" tři základní barevné složky (červená, zelená a modrá), zatímco ARGB obsahuje navíc byt pro *alfa kanál* definující stupeň průhlednosti daného obrazového bodu. Pro formát RGB použijeme hodnotu *PixelFormat24bppRGB* a pro ARGB pak *PixelFormat32bppARGB*.

Ukažme si opět použití v praxi. Ukázkový projekt najdete v doprovodném materiálu pod názvem *DataBitmapy*. Zdrojový kód je tentokrát opět ve stylu „vše v jednom" s využitím

vícenásobné dědičnosti a tříd ATL knihovny *CAtlExeModule* a *CWindowImpl*. Aplikace umožňuje volbou z hlavní nabídky okna otevřít grafický soubor, který se následně zobrazuje v klientské oblasti okna. Jako ukázka úprav dat obrázku jsou realizovány tři funkce: převod na 256 stupňů šedé, inverze barev – negativ a „vykostkování“ levé horní čtvrtiny plochy obrázku. Na *obrázku 8* vidíme aplikované efekty „negativ“ a „vykostkování“.

Obrázek 8 Ukázka filtrů: negativ a vykostkování části obrázku

Jak uvidíme ve zdrojovém kódu, velikost „kostky“, do které jsou barevně zprůměrovány body ležící v její oblasti, lze nastavit parametrem.

Uveďme si nyní zdrojový kód s vloženými komentáři.

Hlavičkový soubor *stdafx.h*:

```
#pragma once

#define STRICT

#define WINVER 0x0500
#define _WIN32_WINNT 0x0500
#define _WIN32_WINDOWS 0x0410
#define _WIN32_IE 0x0500

#define _ATL_APARTMENT_THREADED
#define _ATL_NO_AUTOMATIC_NAMESPACE
#define _ATL_CSTRING_EXPLICIT_CONSTRUCTORS
```

```
#define _ATL_NO_COM_SUPPORT
#define _INC_WINDOWSX

#include <atlbase.h>
#include <atlcom.h>
#include <atlstr.h>
#include <atlwin.h>
using namespace ATL;

#include <shlobj.h>

#include <gdiplus.h>
using namespace Gdiplus;

#pragma comment (lib, "comctl32.lib")
#pragma comment (lib, "gdiplus.lib")
```

Skript prostředků *DataBitmapy.rc*:

```
// Microsoft Visual C++ generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
//////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "winres.h"

//////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

//////////////////////////////////////////////////////
// Czech resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_CSY)
#ifdef _WIN32
LANGUAGE LANG_CZECH, SUBLANG_DEFAULT
#pragma code_page(1250)
#endif //_WIN32

#ifdef APSTUDIO_INVOKED
//////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE
BEGIN
    "#include ""winres.h""\r\n"
```

```
    "\0"
END

3 TEXTINCLUDE
BEGIN
    "\r\n"
END

#endif    // APSTUDIO_INVOKED

/////////////////////////////////////////////////////////
//
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 1,0,0,1
 PRODUCTVERSION 1,0,0,1
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x4L
 FILETYPE 0x1L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040504b0"
        BEGIN
            VALUE "Comments", "Úpravy dat bitmapy"
            VALUE "CompanyName", "Radek Chalupa"
            VALUE "FileDescription", "Úpravy dat bitmapy"
            VALUE "FileVersion", "1.0.0.1"
            VALUE "InternalName", "DataBitmapy.exe"
            VALUE "LegalCopyright",
             "Copyright © 2000-2006 Radek Chalupa"
            VALUE "OriginalFilename", "DataBitmapy.exe"
            VALUE "ProductName", "Úpravy dat bitmapy"
            VALUE "ProductVersion", "1.0.0.1"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x405, 1200
    END
END

/////////////////////////////////////////////////////////
//
// Icon
//
// Icon with lowest ID value placed first
```

```
// to ensure application icon
// remains consistent on all systems.
IDI_HLAVNI          ICON           "res\\hlavni.ico"

//////////////////////////////////////////////////////
//
// Menu
//

IDR_HLAVNI MENU
BEGIN
    POPUP "&Soubor"
    BEGIN
        MENUITEM "&Otevřít",                    ID_OTEVRIT
        MENUITEM "&Uložit",                     ID_ULOZIT
        MENUITEM "&Zavřít",                     ID_ZAVRIT
        MENUITEM SEPARATOR
        MENUITEM "&Konec",                      ID_KONEC
    END
    POPUP "Úpr&avy"
    BEGIN
        MENUITEM "N&egativ",                    ID_NEGATIV
        MENUITEM "&Stupně šedé",                ID_STUPNE_SEDE
        MENUITEM "&Kostky",                     ID_KOSTKY
    END
END

//////////////////////////////////////////////////////
//
// String Table
//

STRINGTABLE
BEGIN
    IDS_PROJNAME            "DataBitmapy"
END

#endif    // Czech resources
//////////////////////////////////////////////////////


#ifndef APSTUDIO_INVOKED
//////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//

//////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
```

Zdrojový soubor *DataBitmapy.cpp* **realizující zapouzdření aplikace i okna v jedné třídě:**

```
/*****************************************************
Soubor: DataBitmapy.cpp
(C) 2006 Radek Chalupa - www.radekchalupa.cz
*****************************************************/
```