Grafický formát BMP

Obsah

1. Úvodní informace o grafickém formátu BMP

Grafický formát **BMP** (*BitMaP)* patří v současnosti mezi nejpoužívanější grafické formáty, což je z technologického pohledu docela paradoxní, protože je poměrně složitý na zpracování a přitom nabízí pouze minimum užitečných vlastností. Oba dva dříve popisované formáty, tj. jak **GIF**, tak i **PNG**, umožňují ukládání obrázku v mnohem více variantách a přitom používají lepší komprimační schéma. Postup použitý při komprimaci v BMP je tak špatný, že se používá pouze u minima obrázků a mnoho aplikací s komprimovanými obrázky ani neumí korektně pracovat.

V čem však tkví taková rozšířenost BMP, pro kterou nemluví technologické parametry? Je to z prozaického důvodu: tento formát byl navržen firmami IBM a Microsoft (každá firma navrhla jinou variantu) jako základní rastrový obrazový formát pro jejich operační systémy (OS/2 a Microsoft Windows). Tím pádem je načítání i ukládání obrázků v tomto formátu podporováno přímo v aplikačním rozhraní daného operačního systému a tvůrci programů mohou toto rozhraní využít bez toho, aby daný formát detailně znali. Obzvlášť zajímavé je ukládání interních obrázků aplikací v BMP, především dnes, kdy se rozdíl mezi rychlostí zpracování dat v CPU a rychlostí přenosu či ukládání dat na datová média stále zvyšuje – jinými slovy, výhodnější je provádět komprimaci náročnou na CPU než zatěžovat přenosové linky nebo datová média.

Z předchozích odstavců mohli čtenáři správně vydedukovat, že BMP nepatří zrovna mezi formáty úsporné. Je tomu z několika důvodů:

1. Neúspornost začíná už u datové hlavičky celého souboru, která obsahuje několik rezervovaných, tj. z praktického hlediska zbytečných položek, které ani v minulosti a s velkou pravděpodobností ani v budoucnosti nebudou využity.
2. Další zdroj neúspornosti můžeme vidět na příkladu obrázků obsahujících barvovou paletu. Pro každé místo v barvové paletě jsou rezervovány čtyři byty místo dostačujících bytů tří. Poslední byte však není možné (podle platné specifikace) použít pro jiné účely, tedy ani pro alfa-kanál (průhlednost). U některých obrázků se tak stává, že barvová paleta má větší objem než vlastní obrazová data (například ikony). Podpora alfa kanálu je zavedena až od Windows XP a mnoho dalších programů ji neimplementuje.
3. Třetí a zdaleka nejzávažnější důvod, proč formát BMP řadím mezi formáty neúsporné, spočívá v použitém komprimačním schématu (metodě). Zvolený postup komprimace je tak "dobře" navržen, že pro mnoho obrázků dochází spíše k nárůstu velikosti souboru místo jejího

snižování! Kromě toho je komprimace povolena pouze pro obrázky s barvovou paletou a nikoli pro plnobarevné (truecolor) obrázky. Všechny obrazové řádky musí mít velikost (bytovou) dělitelnou čtyřmi.

Zajímavý je také způsob ukládání obrazových řádků do souborů. Ty se neukládají směrem shora dolů, jak je to přirozené pro programátory, procesory, operační paměti i pro použité zobrazovací prostředky (grafické karty, tiskárny), ale přesně naopak – zespoda nahoru. Vzhledem k tomu, že i vykreslování obrázků pomocí WinAPI se provádí od spodního okraje, se vše výrazně komplikuje.

Důvod, proč je směr vykreslování a ukládání u bitmap (obrázků typu BMP) opačný, spočívá v tom, že tuto orientaci původně používal operační systém OS/2, který je však v tomto ohledu alespoň konzistentní – souřadnice rostou směrem od levého spodního rohu směrem doprava a nahoru. Ve WinAPI se však používá "klasické" a přirozenější orientace souřadnic, které rostou od levého horního rohu směrem doprava a dolů, a tak se opačný smysl vykreslování bitmap do tohoto systému vůbec nehodí a u několika API funkcí způsobuje zmatky v chápání směru.

2. Zevrubný popis rastrového souboru typu BMP

Rastrové (obrazové) soubory typu BMP jsou uloženy v takzvaném formátu nezávislém na zařízení (*Device Independent Bitmap*), ostatně místo zkratky **BMP** se v minulosti používala také zkratka **DIB** (viz další odstavec). Nejedná se o nic jiného než o rastrový obrázek uložený způsobem, který není závislý na interních metodách práce s barvou nebo uspořádáním pixelů. Prakticky všechny dnes používané obrazové formáty jsou nezávislé na zařízení, mezi formáty závislé patří interní formát (či spíše pseudoformát) využívaný u digitálních fotoaparátů – jde o "surový" (**RAW**) formát odpovídající datům přečteným ze světlocitlivé matice čipu CCD.

Formát **DIB** byl mezi uživatelskou i programátorskou veřejnost zaveden spolu s operačním systémem Microsoft Windows verze 2.0. Od Microsoft Windows verze 3.0 se používá částečně vylepšená verze označená zkratkou **BMP**. Rastrové soubory uložené v předchozí verzi měly koncovku *.raw*, soubory uložené ve verzi stávající pak koncovku *.bmp* (Microsoft Windows patří mezi systémy, které typ souboru rozeznávají podle koncovky, nikoli podle metadat uložených v hlavičce, což mnohdy vede k bezpečnostním problémům).

Formát BMP je navržen tak, že umožňuje ukládání rastrových dat ve čtyřech formátech:

- **1 bit na pixel** – dvoubarevné obrázky (používá se barevná paleta, nemusí se tedy jednat pouze o černobílé grafiky, ale o libovolnou kombinaci dvou barev)
- **4 bity na pixel** – 16ti barevné obrázky (taktéž se používá barevná paleta o délce 64 bytů, v minulosti nejpoužívanější typ, zejména na grafických kartách EGA a VGA)
- **8 bitů na pixel** – 256ti barevné obrázky (opět se používá barevná paleta, tentokrát o délce 1024 bytů)
- **24 bitů na pixel** – TrueColor obrázky (16 milionů barev, barevná paleta se nepoužívá, protože každý pixel je reprezentován přímo svou barvou).

3. Testovací obrázky

Pro ukázku struktur různých typů BMP souborů byly vytvořeny dva testovací obrázky. Na prvním obrázku samozřejmě nemohlo být zobrazeno nic jiného než počítačovými grafiky oblíbená Lenna. Jedná se o obrázek s rozlišením 256×256 pixelů ve formátu plných barev (truecolor), ve skutečnosti je však použito "pouze" 31 840 barvových odstínů. Na druhém testovacím obrázku je zobrazena známá Mandelbrotova množina. Jde o obrázek s rozlišením 320×240 pixelů, který obsahuje pouze 16 základních barev standardní palety grafických karet EGA a VGA.

4. Interní struktura souboru typu BMP

Každý korektní soubor typu BMP obsahuje následující datové struktury, jejichž název odpovídá C-čkovským strukturám definovaným ve WinAPI:

Název struktury	Význam
BITMAPFILEHEADER	hlavička BMP souboru
BITMAPINFOHEADER	informační hlavička o obrázku
RGBQUAD[]	tabulka barev (barvová paleta)
BITS	pole bitů obsahujících vlastní rastrová data (pixely)

Význam těchto datových struktur je následující:

1. **BITMAPFILEHEADER**: jedná se o datovou strukturu, která obsahuje základní informace o souboru typu BMP. Velikost této struktury je konstantní a má hodnotu 14 bytů.
2. **BITMAPINFOHEADER**: tato datová struktura obsahuje základní metainformace o uloženém obraze. Velikost této struktury je opět konstantní, zde jde o 40 bytů.
3. **RGBQUAD[]**: pole obsahující barvovou paletu ve formě složek RGB. Typická délka barvové palety, tj. počet barev, je 2, 16 a 256.
4. **BITS**: v této datové struktuře jsou uložena vlastní obrazová data. Konkrétní formát těchto dat závisí na použité komprimační metodě (i na tom, zda je vůbec použita) a na celkovém počtu barev v obrázku.

Vícebytové sekvence jsou uloženy ve formátu little-endian, tj. byte s nejvyšší váhou je uložený jako poslední. To platí jak pro dvoubytové, tak i pro čtyřbytové sekvence (a také pro barvovou paletu).
V následujících kapitolách si výše uvedené datové struktury popíšeme podrobněji.

5. Hlavička souboru BMP

Hlavička souboru typu BMP, tj. datová struktura **BITMAPFILEHEADER**, má délku čtrnácti bytů a obsahuje údaje o typu, velikosti a celkovém uspořádání dat v souboru. Hlavička má následující formát:

Název položky	Délka položky	Význam
bfType	2 byty	Identifikátor formátu BMP. Aktuální verze formátu BMP zde obsahuje ASCII kód znaků "BM", tj. 0x42 a 0x4d hexadecimálně.
bfSize	4 byty	Celková velikost souboru s obrazovými údaji. Některé aplikace tuto položku ignorují a dosazují zde nulu.

bfReserved1	2 byty	Tento údaj je rezervovaný pro pozdější použití. V současné verzi formátu BMP zde musí být uložena nulová hodnota.
bfReserved2	2 byty	I tento údaj je rezervovaný pro pozdější použití. V současné verzi formátu BMP zde musí být uložena nulová hodnota.
bfOffBits	4 byty	Posun struktury *BITMAPFILEHEADER* od začátku vlastních obrazových dat.

První testovací obrázek (Lenna) obsahuje tuto hlavičku:

Offset	Byte či sekvence	Význam sekvence
0	42 4d	znaky 'BM' – identifikace souboru typu BMP
2	36 00 03 00	celková velikost souboru je rovna 196662 bytům (0x00030036)
6	00 00	rezervovaná položka číslo 1
8	00 00	rezervovaná položka číslo 2
10	36 00 00 00	posun obrazových dat od začátku této struktury je roven 54 bytům (0x00000036)

Druhý testovací obrázek (Mandelbrotova množina) má hlavičku mírně odlišnou:

Offset	Byte či sekvence	Význam sekvence
0	42 4d	znaky 'BM' – identifikace souboru typu BMP
2	76 96 00 00	celková velikost souboru je rovna 38518 bytům (0x00009676)
6	00 00	rezervovaná položka číslo 1
8	00 00	rezervovaná položka číslo 2
10	76 00 00 00	posun obrazových dat od začátku této struktury je roven 118 bytům (0x00000076)

6. Metainformace o uloženém rastrovém obrazu

Datová struktura **BITMAPINFOHEADER** obsahuje základní metainformace o rastrovém obraze. Mezi tyto informace patří především jeho rozměry, tj. výška a šířka, dále pak identifikace použité komprimační metody a specifikace formátu rastrových dat. Celková velikost této struktury je vždy rovna 40 bytům, čemuž ostatně odpovídá i údaj o 54 bytech uvedený v předchozí tabulce (54=14+40). Datová struktura **BITMAPINFOHEADER** má tuto organizaci:

Název položky	Délka položky	Význam
biSize	4 byty	Tato položka specifikuje celkovou velikost datové struktury *BITMAPINFOHEADER*
biWidth	4 byty	Tato položka udává šířku obrazu zadávanou v pixelech
biHeight	4 byty	Tato položka udává výšku obrazu zadávanou taktéž v pixelech
biPlanes	2 byty	V této položce je zadaný počet bitových rovin pro výstupní zařízení. V BMP, jakožto formátu nezávislém na zařízení, je zde vždy hodnota 1. Položka existuje z historických důvodů.
biBitCount	2 byty	V této položce je specifikovaný celkový počet bitů na pixel. Podle počtu barev zde mohou být hodnoty 1, 4, 8 nebo 24 (to odpovídá postupně 2, 16, 256ti barvám popř. plnobarevnému režimu).
biCompression	4 byty	Udává typ komprimační metody obrazových dat. Musí být nastavené na jednu z hodnot: 0 (BI_RGB),

		1 (BI_RLE8) nebo 2 (BI_RLE4).
biSizeImage	4 byty	Tato položka udává velikost obrazu v bytech. Pokud je bitmapa nekomprimovaná, může zde být nulová hodnota, protože ji je možno vypočítat z rozměrů obrázků a počtu bitů na pixel.
biXPelsPerMeter	4 byty	Udává horizontální rozlišení výstupního zařízení v pixelech na metr. Tato hodnota může být použita například pro výběr obrazu ze skupiny obrazů, který nejlépe odpovídá rozlišení daného výstupního zařízení. Většina aplikací však nemá potřebné informace o výstupním zařízení, a proto do této položky vkládá hodnotu 0.
biYPelsPerMeter	4 byty	Udává vertikální rozlišení výstupního zařízení v pixelech na metr. Opět, jako u předchozí položky, zde většina programů zapisuje hodnotu 0.
biClrUsed	4 byty	Udává celkový počet barev, které jsou použité v dané bitmapě. Jestliže je tato hodnota nastavena na nulu (což provádí většina aplikací), znamená to, že bitmapa používá maximální počet barev. Tento počet lze jednoduše zjistit z položky biBitCount. Nenulová hodnota může být použita například při optimalizacích zobrazování.
biClrImportant	4 byty	Udává počet barev, které jsou důležité pro vykreslení bitmapy. Pokud je tato hodnota nulová (téměř vždy), jsou všechny barvy důležité. Tento údaj je používán při zobrazování na zařízeních, které mají omezený počet současně zobrazitelných barev (například starší grafické karty se 16ti resp. 256ti barevnými režimy). Ovladač displeje může upravit systémovou paletu tak, aby zobrazil daný obrázek co nejvěrněji. Také je vhodné upravit paletu metodou seřazení jednotlivých barev podle důležitosti.

Opět se podíváme na to, jakým způsobem je datová struktura **BITMAPINFOHEADER** použita v demonstračním obrázku (Lenna):

Offset	Byte či sekvence	Význam sekvence
14	28 00 00 00	délka celé struktury je rovna 40 bytům
18	00 01 00 00	šířka obrázku je 256 pixelů
22	00 01 00 00	výška obrázku je taktéž 256 pixelů
26	01 00	počet bitových rovin=1
28	18 00	počet bitů na pixel=24 (truecolor)
30	00 00 00 00	metoda komprimace=žádná
34	00 00 00 00	velikost obrazu je zjištěna z jeho rozměrů
38	00 00 00 00	horizontální počet pixelů na metr není zadán
42	00 00 00 00	vertikální počet pixelů na metr taktéž není zadán
46	00 00 00 00	je použit maximální počet barev (aplikace se nenamáhala s jejich zjištěním, plný počet barev nelze v tomto obrázku zobrazit)
50	00 00 00 00	důležité jsou všechny barvy, tato položka je u plnobarevných obrázků bezvýznamná

Druhý demonstrační obrázek (Mandelbrotova množina) je poněkud odlišný:

Offset	Byte či sekvence	Význam sekvence
14	28 00 00 00	délka celé struktury je rovna 40 bytům
18	40 01 00 00	šířka obrázku je 320 pixelů
22	f0 00 00 00	výška obrázku je 240 pixelů
26	01 00	počet bitových rovin=1
28	04 00	počet bitů na pixel=4 (16 barev)
30	00 00 00 00	metoda komprimace=žádná
34	00 00 00 00	velikost obrazu je zjištěna z jeho rozměrů
38	00 00 00 00	horizontální počet pixelů na metr není zadán
42	00 00 00 00	vertikální počet pixelů na metr taktéž není zadán
46	10 00 00 00	je použito 16 barev
50	00 00 00 00	důležité jsou všechny barvy

7. Barvová paleta

V předchozích kapitolách jsme si řekli, že barvová paleta je v obrázcích typu BMP uložena ve formátu RGB, ovšem tak, že každá barva zabírá čtyři byty místo potřebných třech bytů. První demonstrační obrázek barvovou paletu neobsahuje, druhý ano, proto se můžeme podívat na její strukturu:

Sekvence bytů	Význam sekvence
00 00 00 00	barva číslo 0
aa 00 00 00	barva číslo 1
00 aa 00 00	barva číslo 2
aa aa 00 00	barva číslo 3
00 00 aa 00	barva číslo 4
aa 00 aa 00	barva číslo 5
00 55 aa 00	barva číslo 6
aa aa aa 00	barva číslo 7
55 55 55 00	barva číslo 8
ff 55 55 00	barva číslo 9
55 ff 55 00	barva číslo 10
ff ff 55 00	barva číslo 11
55 55 ff 00	barva číslo 12
ff 55 ff 00	barva číslo 13
55 ff ff 00	barva číslo 14
ff ff ff 00	barva číslo 15

Vidíme, že vždy čtvrtý byte je nastavený na nulovou hodnotu a první trojice bytů udává hodnotu tří barvových složek, ovšem uložených opačně než bychom očekávali – místo RGB jde o posloupnost BGR! Na tuto skutečnost je zapotřebí dávat pozor jak při ukládání do BMP, tak i při zobrazování bitmap pomocí funkcí WinAPI.

8. Obrazová data uložená v souborech typu BMP

Rastrová data (pixely) jsou v grafickém souboru typu BMP uložena buď za barvovou paletou (v případě obrázků s 1bpp, 4bpp a 8bpp), nebo přímo za informační hlavičkou (obrázky s 16bpp, 24bpp a 32bpp). Data reprezentují po sobě jdoucí obrazové řádky (*scanlines*). Každý obrazový řádek sestává z hodnot reprezentujících jednotlivé pixely v pořadí zleva doprava. Počet bytů popisujících obrazový řádek je závislý na šířce obrázku a na barvovém formátu, tj. počtu bitů na pixel. Obrazový řádek musí být vždy zarovnaný na 4 byty (32 bitů), v případě nutnosti se doplňuje na 4 byty nulami – na toto zarovnání je zapotřebí myslet při načítání i ukládání.

Obrazové řádky se v rastrovém obrázku ukládají směrem zdola nahoru, což je odlišný způsob od většiny ostatních obrazových formátů (výjimku tvoří například grafický formát **TGA**, v jehož hlavičce je možné také nastavit opačné ukládání). To znamená, že první byte v poli **BITS** reprezentuje pixel (resp. pixely či jejich část) v levém dolním rohu bitmapy. Poslední byte v tomto poli potom reprezentuje barvu pixelů v pravém horním rohu.

Položka struktury **BITMAPINFOHEADER** nazvaná **biBitCount** (viz předchozí část tohoto seriálu) určuje počet bitů, které jsou zapotřebí pro reprezentaci barvy každého pixelu. Tato položka současně určuje maximální počet barev v bitmapě. Význam jednotlivých hodnot této položky je následující:

1 - Bitmapa je monochromatická, přičemž barevná paleta obsahuje dvě položky. Každý bit pole bitmapy reprezentuje jeden pixel. Jestliže není bit nastavený (je nulový), pixel bude zobrazený barvou první položky barvové palety. Jestliže je bit nastavený (je jedničkový), pixel bude zobrazený barvou druhé položky barvové palety.

4 - Bitmapa má maximálně 16 barev, přičemž barvová paleta obsahuje maximálně 16 položek. Každý pixel bitmapy je reprezentován čtyřmi bity. To znamená, že v jednom byte je uložen index do tabulky barev pro dva pixely. Pokud například první byte v bitmapě má hodnotu 0x2a, byte reprezentuje dva pixely. První pixel bude mít barvu třetí položky barvové palety (hodnota horních čtyřech bitů je 2), druhý pixel bude mít barvu jedenácté položky barvové palety (hodnota spodních čtyřech bitů je 10).

8 - Bitmapa má maximálně 256 barev, přičemž barvová paleta obsahuje maximálně 256 položek. Každý pixel bitmapy je reprezentovaný jedním bytem, který udává index do tabulky barev, podobně jako v předchozích případech.

16 - Bitmapa je uložena Hi-color režimu, na každou barvovou složku připadá 5 bitů, výjimku tvoří zelená barvová složka, která může být uložena na 6 bitů. Barvová paleta není použita.

24 - Bitmapa je uložena v TrueColor režimu, tzn. obsahuje maximálně 2^{24} barev (16 777 216 barev). Barevná paleta není uložena, protože je pixel reprezentován přímo svými RGB hodnotami. Každý pixel je uložen ve třech bytech, kde každý byte reprezentuje intenzity červené, zelené a modré barvové složky.

32 - Bitmapa je uložena v TrueColor režimu, tzn. obsahuje maximálně 2^{24} barev (16 777 216 barev). Podobně jako v předchozím typu, i zde není použita barvová paleta, zato je použita bitová maska na hodnoty jednotlivých barvových složek.

9. Komprimace obrazových dat v souborech typu BMP

Operační systémy Windows od verze 3.0 podporují jednoduchou RLE (Run Length Encoding) komprimaci bitových map. RLE komprimaci lze využít pouze u bitmap s čtyřmi resp. osmi bity na pixel. U jednobitových a TrueColor bitmap jsou obrázky vždy uloženy v nekomprimované podobě, což však, jak uvidíme dále, nemusí být na škodu, protože komprimační poměry jsou malé.

Jestliže je položka *biCompression* struktury **BITMAPINFOHEADER** nastavena na hodnotu *BI_RLE8*, je bitmapa komprimovaná RLE metodou pro 256ti barvové bitmapy.

Informace jsou v bitmapě organizovány do skupiny po dvou bytech. První byte určuje počet po sobě jdoucích pixelů, které budou mít index barvy určený v druhém bytu. První byte ve skupině může být nastavený na nulu, což znamená tzv. *únik*: například konec řádku, konec bitmapy nebo delta. Typ úniku je zapsán ve druhém bytu skupiny. Význam jednotlivých únikových hodnot je následující:

hodnota druhého byte ve skupině	význam úniku
0x00	indikuje konec řádku
0x01	indikuje konec bitmapy
0x02	delta – dva byty následující za touto hodnotou obsahují bezznaménkové (unsigned char) hodnoty, které znamenají horizontální a vertikální posunutí následujících pixelů od aktuální pozice
0x03-0xff	druhý byte určuje počet bytů, které následují. Tyto byty přímo určují pixely v nezkomprimovaném tvaru – co byte to jeden pixel.

Následuje ukázka 256ti barevné bitmapy (všechny hodnoty jsou zapsané v hexadecimálním tvaru):

Komprimovaná data	Nekomprimovaná data
05 10	10 10 10 10 10
03 aa	aa aa aa
00 03 12 34 56	12 34 56
02 bb	bb bb
00 02 0a 14	posun o 10 pixelů vpravo a 20 pixelů dolů
02 cc	cc cc
00 00	konec řádku
01 dd	dd
00 01	konec bitmapy

Jestliže je položka *biCompression* struktury *BITMAPINFOHEADER* nastavena na hodnotu *BI_RLE4*, je bitmapa komprimovaná RLE metodou pro šestnáctibarevné bitmapy.

Informace jsou v bitmapě organizovány do skupiny dvou bytů. První byte určuje počet po sobě jdoucích pixelů, které budou mít index barvy určený v druhém bytu. Druhý byte obsahuje dva indexy barev, uložené v nejvyšších resp. nejnižších čtyřech bitech daného bytu (masky pro získání těchto čtyřech bitů jsou 0x0f a 0xf0). První pixel ve skupině je zobrazený barvou s indexem získaným maskou 0xf0, druhý pixel barvou s indexem získaným maskou 0x0f. Třetí pixel má barvu stejnou jako první pixel atd. Barvy jsou postupně přiřazeny všem pixelům ve skupině.

První byte ve skupině může být nastavený na nulu, což znamená takzvaný únik: například konec řádku, konec bitmapy nebo delta. Typ úniku je zapsán v druhém bytu skupiny. Význam jednotlivých únikových hodnot:

hodnota druhého byte ve skupině	význam úniku
0x00	indikuje konec řádku
0x01	indikuje konec bitmapy
0x02	delta – dva byty následující za touto hodnotou obsahují bezznaménkové (unsigned char) hodnoty, které znamenají horizontální a vertikální posunutí následujících pixelů od aktuální pozice
0x03-0xff	druhý byte určuje počet indexů barev, které následují v nezkomprimovaném tvaru.

Ukázka 16ti barevné bitmapy (všechny hodnoty jsou zapsané v hexadecimálním tvaru):

Komprimovaná data	Nekomprimovaná data
05 01	0 1 0 1 0
03 12	1 2 1
00 06 12 34 56	1 2 3 4 5 6
04 ab	a b a b
00 02 0a 14	posun o 10 pixelů vpravo a 20 pixelů dolů
02 cc	c c
00 00	konec řádku
09 ef	e f e f e f e f e
00 01	konec bitmapy