Aplicatie pentru manipularea fisierelor de tip Bitmap

*Introducere*

***Cerinte:*** Studiu teoretic si aplicativ asupra formatului grafic BMP cu paleta pe 16 culori. Studiul trebuie sa raspunda la:

1. De ce si pentru ce este utilizat acest format de fisiere de imagine.
2. Ce facilitati ofera acesta fata de altele de acelasi tip.
3. Care este structura fisierului.
4. Studiu de caz/exemplu.
5. Proiectare si implementare aplicatie pentru manipulare fisier.

***Obs:*** Pasajele preluate din literatura de specialitate vor fi insotite de referinta bibliografica prin specificarea unui numar intre „ [] ” si plasarea detaliului la sfarsitul referatului in zona numita bibliografie.

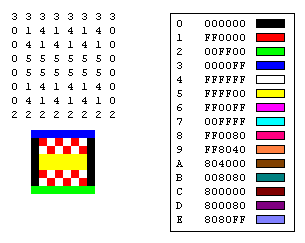
1. De ce si pentru ce este utilizat acest format de fisiere de imagine:

Formatul de fișier **BMP** (en. **B**it**m**a**p** - hartă de biți), cunoscut de asemenea și ca *fișier imagine Bitmap* sau formatul de fișier **DIB** (en. **Device Independent Bitmap** - „bitmap” independent de dispozitiv) sau pur și simplu **Bitmap**, este un format de fișier imagine de tip rastru folosit pentru a stoca imagini digitale independent de dispozitivul de afișare (cum ar fi un adaptor grafic) în special pe sistemele de operare Microsoft Windows și OS/2.

Formatul de fișier BMP este capabil de a stoca imagini digitale bidimensionale de lățime, înălțime și rezoluție arbitrare, monocrome sau color, de variate profunzimi ale culorii iar opțional cu comprimare de date, canale alfa și profile de culoare.[1]

Formatul de fisier BMP cu paleta pe 16 culori sau formatul de fisier BMP pe 4 biti poate reprezenta pe cate un pixel dat 16 culori diferinte (2^4 = 16).

Fisierele imagine de tip Bitmap contin de obicei unul sau mai multe blocuri care stocheaza informatia cum ar fi numarul de biti pe pixel, numarul de pixel de pe fiecare linie, numarul de linii din matrice. Fisierele respective pot contine de altfel si o matrice de culori (numita si paleta de culori). O matrice de culori mapeaza corespondente numerice in fisier pentru culoarea respectiva. Urmatoarea ilustratie arata o imagine marita alaturata de bitmap-ul si matricea de culori. Fiecare pixel este reprezentat de un numar pe 4 biti, deci avem 2^4 = 16 culori in matricea de culori. Fiecare culoare din tabel este reprezentata de un numar pe 24 de biti: 8 biti pentru rosu, 8 biti pentru verde, 8 biti pentru albastru. Numerele sunt afisate in format hexazecimal.

[2]

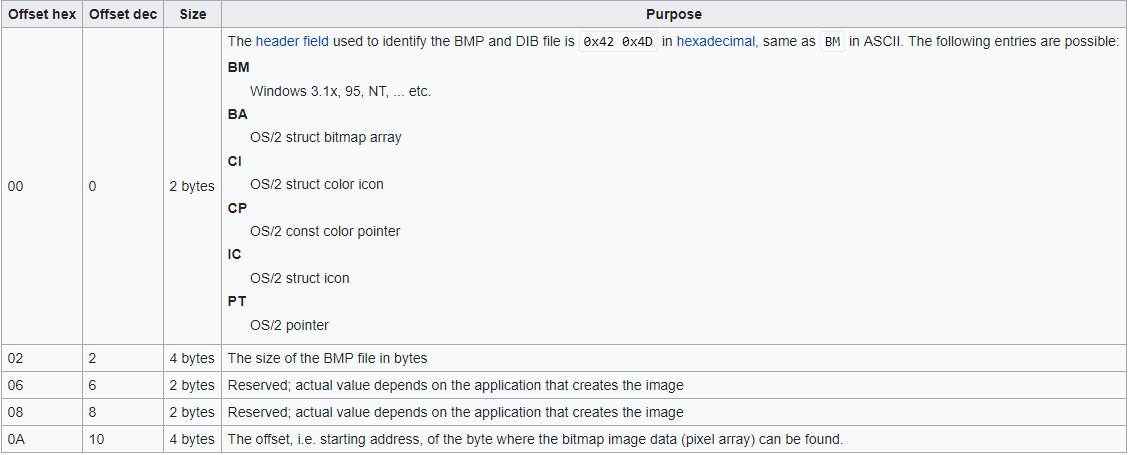
1. Ce facilitati ofera acesta fata de altele de acelasi tip:

BMP este formatul reprezentativ al sistemului de operare Windows. Acesta nu permite in mod normal compresia imaginilor. Fisierele de imagine BMP sunt clare si precise, insa fiind depente de pixel, nu scaleaza bine din punct de vedere al marimii. Pe internet, detaliul acestora vine impreuna cu neajunsul dimensiunii, astfel ca acestea sunt aproape inexistente pe web.[3]

1. Care este structura fisierului:

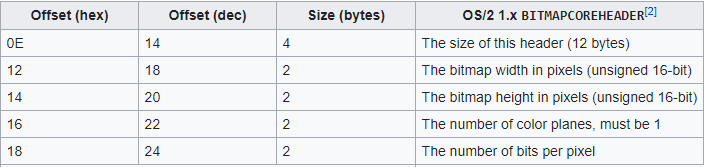
**Bitmap file header**

Acest bloc de octeti este situat la inceputul fisierului si este utilizat la identificarea fisierului. O aplictatie tipica citeste initial acest bloc pentru a se asigura ca fisierul este intr-adevar de format BMP si ca nu este deteriorat. Primii 2 octeti ai formatului BMP sunt caracterul „B”, urmat de caracterul „M” in format ASCII. Toate valorile intregi sunt stocate in format little-endian (cel mai putin semnificativ byte fiind primul).

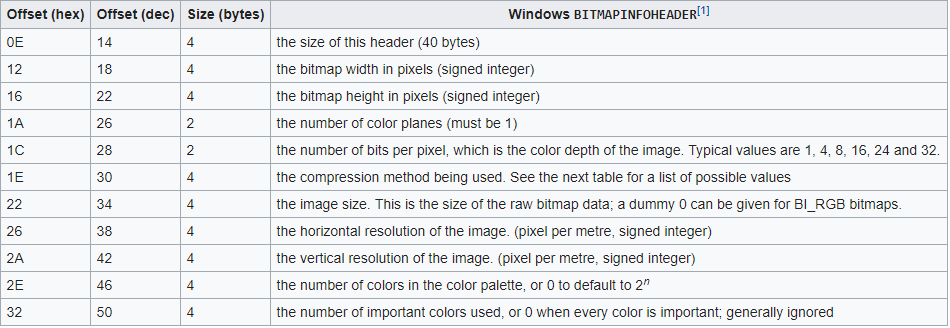


### DIB header (header informatii bitmap)

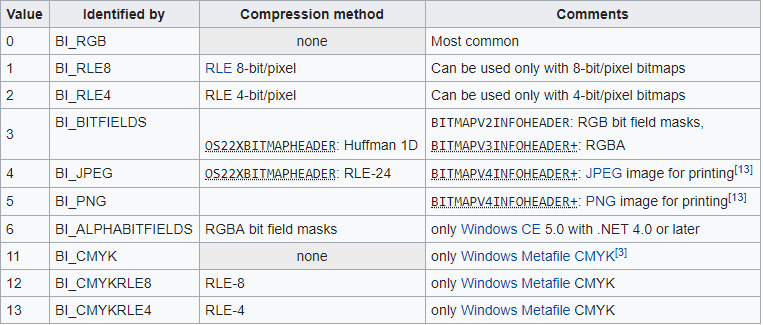
### 



In situatia noastra, pentru formatul BMP cu paleta pe 16 culori (4 biti), trebuie sa avem obligatoriu valoarea 4 in campul 18h.



Un OS/2 2.x OS22XBITMAPHEADER sau BITMAPCOREHEADER2 contine 24 de octeti in plus. Metoda de compresie (offset 30) poate fi:



### Color table

### Matricea de culori apare in fisierul BMP direct dupa BMP file header, DIB header (si dupa 3 masti de rosu, verde si albastru optionale daca headerul BITMAPINFOHEADER cu optiunea BI\_BITFIELDS este utilizata).

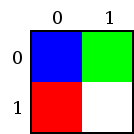
### Numarul intrarile din paleta este fie 2n sau un numar mai mic specificat in header. In majoritatea cazurilor, fiecare sectiune din matricea de culori ocupa 4 octeti, in ordinea albastru, verde, rosu, 0x00.

### Matricea de culori este un block de octeti (un tabel) care listeaza culorile folosite de catre imagine. Fiecare pixel este descris de un numar de biti (1,4 sau 8) care reprezinta indexul unei singure culori descrisa de catre tabel.[4]

### [5]

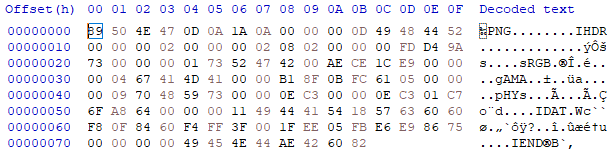
1. Studiu de caz/exemplu

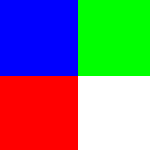
Fie un fisier de tip BMP pe 4 biti, continand o matrice cu 2x2 pixeli dupa figura urmatoare:

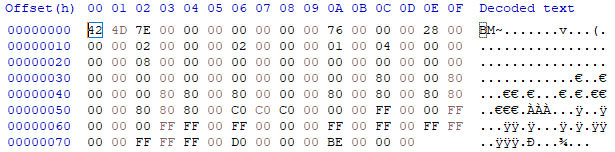


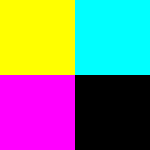
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Dimensiune | Valoare hexa | Semnificatie | Descriere |
| BMP Header | | | | |
| 0h | 2 | 42 4D | „BM” | Specific BMP |
| 2h | 4 | 7E 00 00 00 | 126 octeti (118+8) | Dimensiune fisier BMP |
| 6h | 2 | 00 00 | Nefolosit | Specific aplicatie |
| 8h | 2 | 00 00 | Nefolosit | Specific aplicatie |
| Ah | 4 | 76 00 00 00 | 118 octeti (78+40) | Offset unde se poate gasi harta de pixeli |
| DIB Header | | | | |
| Eh | 4 | 28 00 00 00 | 40 octeti | Numar octeti din headerul DIB |
| 12h | 4 | 02 00 00 00 | 2 pixeli (de la stanga la dreapta) | Latime bitmap in pixeli |
| 16h | 4 | 02 00 00 00 | 2 pixeli (de jos in sus) | Inaltime bitmap in pixeli |
| 1Ah | 2 | 01 00 | 1 plan | Numar de planuri de culoare utilizate |
| 1Ch | 2 | 04 00 | 4 biti | Numar de biti per pixel |
| 1Eh | 4 | 00 00 00 00 | 0 (tip compresie) | 0 inseamna ca nu exista nicio compresie |
| 22h | 4 | 08 00 00 00 | 8 octeti | Dimensiune bruta de date bitmap (include indentarea) |
| 26h | 4 | 00 00 00 00 | 0 pixeli/metru orizontala | Rezolutie printare imagine |
| 2Ah | 4 | 00 00 00 00 | 0 pixeli/metru verticala |
| 2Eh | 4 | 00 00 00 00 | 0 culori | Numbar de culori din paleta |
| 32h | 4 | 00 00 00 00 | 0 culori importante | 0 inseamna ca toate culorile sunt importante |
| 36h | 40h | 40h\* 00...00 | Alte campuri | Masti, alpha, etc. |
|  |  |  |  |  |
| 76h | ½ | 9 | Culoarea 10 | Rosu |
| 76h | ½ | F | Culoarea 16 | Alb |
| 77h | 3 | 00 00 00 | 0 0 0 | Indentare |
| 7Ah | ½ | C | Culoarea 13 | Albastru |
| 7Ah | ½ | A | Culoarea 11 | Verde |
| 7Bh | 3 | 00 00 00 | 0 0 0 | Indentare |

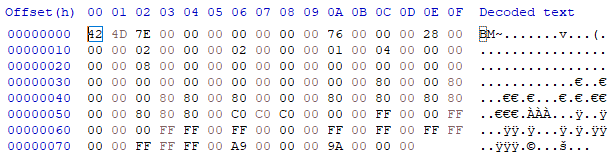
[4]

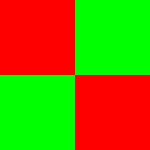
1. 

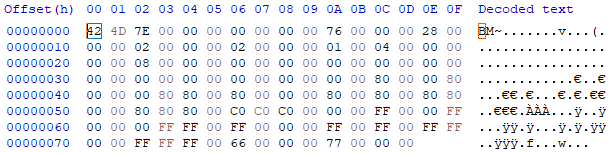


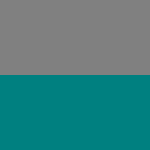
1. 

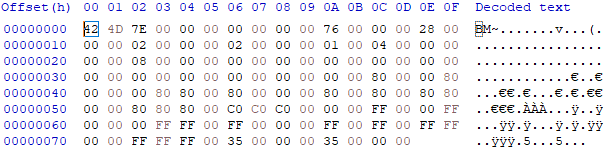


1. 



1. 



1. 



1. Proiectare si implementare aplicatie pentru manipulare fisier

using System;

using System.IO;

using System.Globalization;

using System.Windows.Forms;

namespace spg

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void richTextBox1\_TextChanged(object sender, EventArgs e)

{

}

private void richTextBox2\_TextChanged(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bisize\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biwidth\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biheight\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biplanes\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bibitcount\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bicompression\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bisizeimage\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bixpelspermeter\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biypelspermeter\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biclrused\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_biclrimportant\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bfreserved1\_TextChanged(object sender, EventArgs e)

{

}

private void tx\_bfreserved2\_TextChanged(object sender, EventArgs e)

{

}

private void txt\_bfoffbits\_TextChanged(object sender, EventArgs e)

{

}

public struct BITMAPFILEHEADER

{

public int bfType;

public int bfSize;

public short bfReserved1, bfReserved2;

public int bfOffBits;

}

public struct BITMAPINFOHEADER

{

public int biSize;

public int biWidth, biHeight;

public short biPlanes, biBitCount;

public int biCompression, biSizeImage;

public int biXPelsPerMeter, biYPelsPerMeter;

public int biClrUsed, biClrImportant;

}

public struct COLORTABLE

{

public int rgbBlue;

public int rgbGreen;

public int rgbRed;

public int rgbReserved;

}

BITMAPFILEHEADER bfh = new BITMAPFILEHEADER();

BITMAPINFOHEADER bih = new BITMAPINFOHEADER();

COLORTABLE ct = new COLORTABLE();

byte[] buffer = new byte[100];

private void button1\_Click(object sender, EventArgs e)

{

BinaryReader br = null;

byte[] file = null;

bfh.bfType = 2;

bfh.bfSize = 4;

bfh.bfReserved1 = 2;

bfh.bfReserved2 = 2;

bfh.bfOffBits = 4;

bih.biSize = 4;

bih.biWidth = 4;

bih.biHeight = 4;

bih.biPlanes = 2;

bih.biBitCount = 2;

bih.biCompression = 4;

bih.biSizeImage = 4;

bih.biXPelsPerMeter = 4;

bih.biYPelsPerMeter = 4;

bih.biClrUsed = 4;

bih.biClrImportant = 4;

try

{

//--------------------Citim fisierul bmp--------------------

br = new BinaryReader(File.OpenRead("C:\\untitled.bmp"));

//--------------------Citim in file datele--------------------

file = br.ReadBytes((int)br.BaseStream.Length);

//-------Copiem datele din fisierul BMP in fisierul richTextBox1------

for (int i = 0; i < file.Length; i++)

richTextBox1.Text += Convert.ToString(file[i], 16).PadLeft(2, '0') + " ";

///Adaugam cate un caracter blank la fiecare octet pentru parsare.///

//------------Impartim fisierul in structuri dupa campuri------------

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BITMAPFILEHEADER

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

for (int i = 0; i < bfh.bfType; i++)

txt\_bftype.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul bfType

for (int i = bfh.bfType; i < bfh.bfType + bfh.bfSize; i++)

txt\_bfsize.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul bfType

for (int i = bfh.bfType + bfh.bfSize; i < bfh.bfType + bfh.bfSize + bfh.bfReserved1; i++)

txt\_bfreserved1.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul bfReserved1

for (int i = bfh.bfType + bfh.bfSize + bfh.bfReserved1; i < bfh.bfType + bfh.bfSize + bfh.bfReserved1 + bfh.bfReserved2; i++)

txt\_bfreserved2.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul bfReserved2

for (int i = bfh.bfType + bfh.bfSize + bfh.bfReserved1 + bfh.bfReserved2; i < bfh.bfType + bfh.bfSize + bfh.bfReserved1 + bfh.bfReserved2 + bfh.bfOffBits; i++)

txt\_bfoffbits.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul bfOffBits

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BITMAPFILEHEADER

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int suma = bfh.bfType + bfh.bfSize + bfh.bfReserved1 + bfh.bfReserved2 + bfh.bfOffBits;

for (int i = suma; i < suma + bih.biSize; i++)

txt\_bisize.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biSize

for (int i = suma + bih.biSize; i < suma + bih.biSize + bih.biWidth; i++)

txt\_biwidth.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biWidth

for (int i = suma + bih.biSize + bih.biWidth; i < suma + bih.biSize + bih.biWidth + bih.biHeight; i++)

txt\_biheight.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biHeight

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes; i++)

txt\_biplanes.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biPlanes

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount; i++)

txt\_bibitcount.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biBitCount

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression; i++)

txt\_bicompression.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biCompression

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage; i++)

txt\_bisizeimage.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biSizeImage

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter; i++)

txt\_bixpelspermeter.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biXPelsPerMeter

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter; i++)

txt\_biypelspermeter.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biYPelsPerMeter

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter + bih.biClrUsed; i++)

txt\_biclrused.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biClrUsed

for (int i = suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter + bih.biClrUsed; i < suma + bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter + bih.biClrUsed + bih.biClrImportant; i++)

txt\_biclrimportant.Text += Convert.ToString(file[i], 16).PadLeft(2, '0'); //campul biClrImportant

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

COLORTABLE

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

suma += bih.biSize + bih.biWidth + bih.biHeight + bih.biPlanes + bih.biBitCount + bih.biCompression + bih.biSizeImage + bih.biXPelsPerMeter + bih.biYPelsPerMeter + bih.biClrUsed + bih.biClrImportant+16\*4; //paleta pe 16 culori, 4 octeti de culoare

for (int i = suma; i < file.Length; i++)

richTextBox2.Text += Convert.ToString(file[i], 16).PadLeft(2, '0');

}

catch (Exception exc)

{

//Afisam message box cu exceptia.

MessageBox.Show("Fisierul nu a putut fi deschis sau citit.\n" + "Exception: " + exc.Message);

}

finally

{

if (br != null)

{

//Inchidem

br.Close();

}

}

}

private BinaryWriter dataOut;

private string numeFisier = "c:\\exemplu.bmp";

private void button2\_Click(object sender, EventArgs e)

{

byte[] fileOut = new byte[richTextBox1.Text.Length];

string[] s;

int j = 0;

NumberStyles stil = NumberStyles.HexNumber;

//Parsam richTextBox1 in stringuri separate mai sus de blank, apoi convertim in integer hexa si salvam string-ul rezultat in alt fisier "exemplu.bmp".

try

{

s = richTextBox1.Text.Split(' ');

foreach (string octet in s)

{

byte.TryParse(octet, stil, null as IFormatProvider, out fileOut[j]);

j++;

}

dataOut = new BinaryWriter(new FileStream(numeFisier, FileMode.Create));

dataOut.Close();

dataOut = new BinaryWriter(new FileStream(numeFisier, FileMode.Append));

dataOut.Write(fileOut);

}

catch (Exception exc)

{

//Afisam message box cu exceptia.

MessageBox.Show("Fisierul nu a putut fi salvat.\n" + "Exception: " + exc.Message);

}

finally

{

//Inchidem

dataOut.Close();

}

}

}

}

Bibliografie

[1] – <https://ro.wikipedia.org/wiki/BMP_(format_fișier)>

[2] – <https://docs.microsoft.com/ro-ro/windows/desktop/gdiplus/-gdiplus-types-of-bitmaps-about>

[3] – <https://www.guidingtech.com/7729/difference-between-jpg-png-gif-bmp-image-formats/>

[4] – <https://en.wikipedia.org/wiki/BMP_file_format>

[5] – <https://commons.wikimedia.org/wiki/File:Windows_16_color_palette.svg>