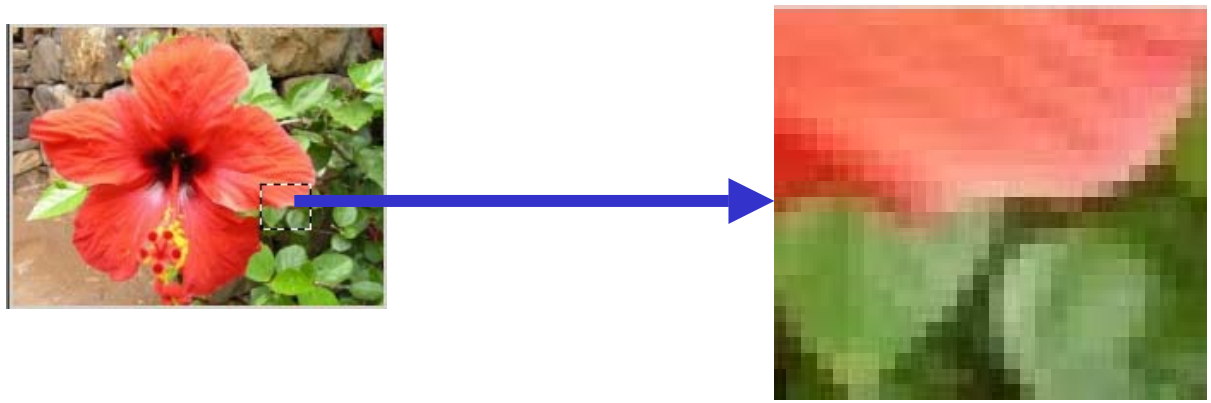
Formaty plików graficznych - wprowadzenie

Obraz graficzny jest dwuwymiarową tablicą pikseli, zwana czasem rastrem.



Kolor piksela może być reprezentowany w następujący sposób:

- **Dla obrazów monochromatycznych – 1 bit (czarny/biały)**
- **Dla obrazów „true color" dla każdego piksela określane są 3 składowe: czerwona (R), zielona (G), i niebieska (B). Gdy wartość każdej składowej reprezentowana jest przez 1 bajt, to każda barwa może być reprezentowana w 256 odcieniach jasności, co daje 16777216 (ponad 16 mln)**
- **W przypadku obrazu opartego na palecie, wartości poszczególnych pikseli są indeksami tablicy RGB, zwanej paletą kolorów. Liczba przypadających na jeden piksel bitów zależy od liczby kolorów w palecie. Typowa paleta może zawierać 16 (4 bity na piksel) lub 256 kolorów (8 bitów na piksel).**

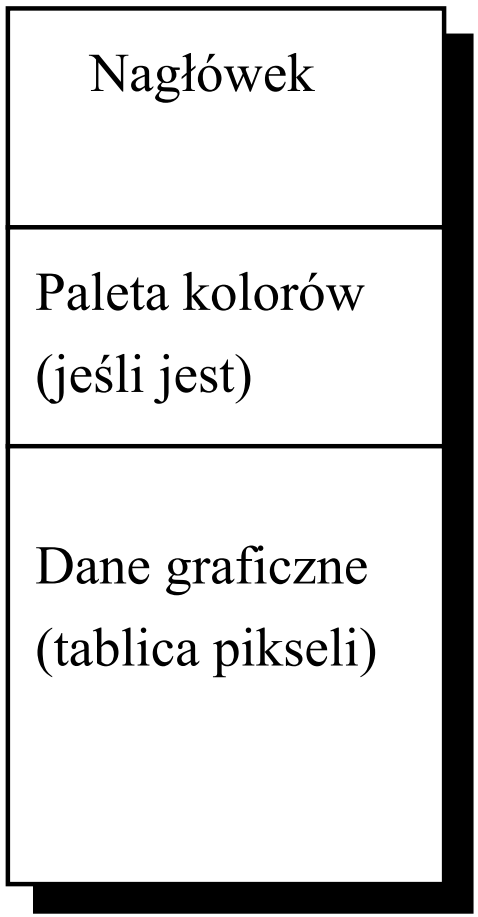
Formaty plików graficznych - wprowadzenie

Minimum informacji konieczne do wyświetlenia obrazu na ekranie:

1. **Rozmiar rysunku (szerokość i wysokość)**
2. **Liczba bitów na piksel (głębokość)**
3. **Rodzaj grafiki (monochromatyczna / „true color” / paleta kolorów)**
4. **Paletę kolorów, jeśli obraz jest opisany na palecie**
5. **Dane graficzne (tablica pikseli)**

Formaty plików graficznych - wprowadzenie

Typowa struktura pliku graficznego:



Formaty plików graficznych - wprowadzenie

Różnice w formatach plików graficznych:

- **Kolejność umieszczonych w nagłówku informacji może być różna dla różnych formatów plików.**
- **Niektóre formaty, przystosowane do konkretnej karty graficznej, nie muszą zawierać palety, a jedynie tablice pikseli.**
- **Linie obrazu mogą być ułożone w kolejności od góry do dołu lub od dołu do góry.**
- **Jeżeli wartości pikseli są składowymi RGB, to kolejność zapisu tych składowych może być różna.**
- **Wartości pikseli mogą być zapisane w postaci pakietów lub planów obrazu. W przypadku pakietów bity, odpowiadające kolejnym pikselom, są zapisywane jeden za drugim. Jeżeli obraz składa się z planów, bity poszczególnych pikseli są zapisywane na tej samej pozycji w kolejnych planach. Najmniej znaczące bity pikseli obrazu znajdują się w jednym planie, a następnym planie są bity bardziej znaczące, itd.**
- **Dane graficzne mogą być skompresowane.**

Formaty plików graficznych – mapy bitowe

Dostępne w systemie Windows mapy bitowe:

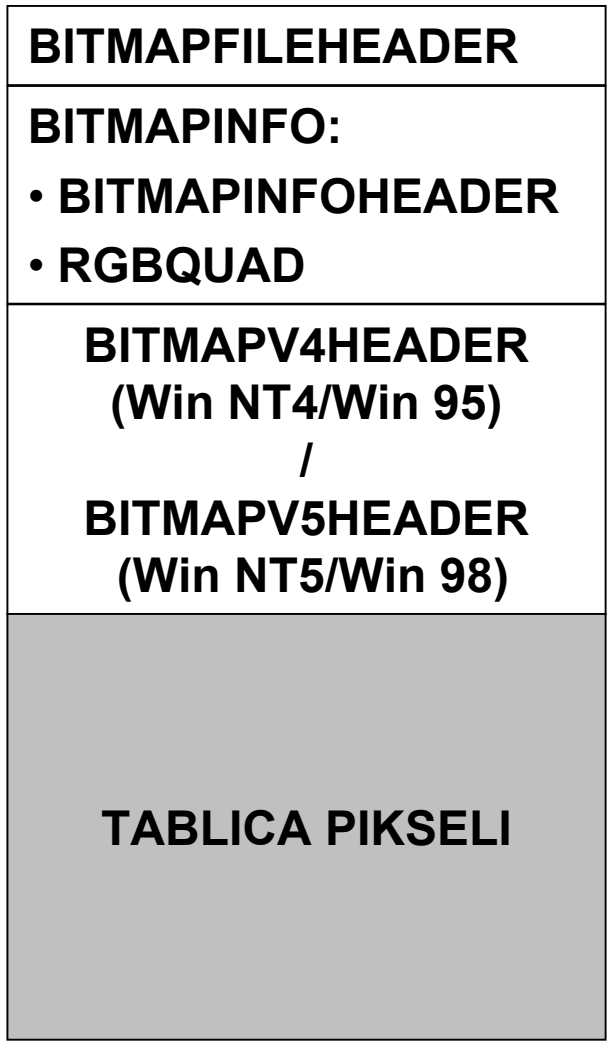
- **Mapy bitowe niezależne od sprzętu (DIB – Device-Independent Bitmap – od wersji 3.0), dostosowane do wyświetlania na dowolnym urządzeniu rastrowym.**
- **Mapy bitowe zależne od sprzętu (DDB – Device-Dependent Bitmap), dostosowane do wyświetlenia na konkretnym urządzeniu wyjściowym.**

Metoda wyświetlania plików graficznych w Windows:

- **Konwersja dowolnego typu graficznego na mapę bitową DIB**
- **Przekształcenie mapy DIB w mapę DDB**
- **Przesłanie mapy DDB na konkretne urządzenie wyświetlające**

Formaty plików graficznych – mapy bitowe

Struktura pliku typu BMP:



Formaty plików graficznych – mapy bitowe

Struktura nagłówka BITMAPFILEHEADER:

```
typedef struct tagBITMAPFILEHEADER { // bmfh
        WORD    bfType;
        DWORD   bfSize;
        WORD    bfReserved1;
        WORD    bfReserved2;
        DWORD   bfOffBits;
} BITMAPFILEHEADER;
```

Pole	Wielkość	Opis
bfType	WORD	Dwa bajty "BM" (od słów: mapa bitowa)
bfSize	DWORD	Całkowita wielkość pliku
bfReserved1	WORD	Wartość 0
bfReserved2	WORD	Wartość 0
bfOffBits	DWORD	Odległość od początku pliku do miejsca, w którym zaczynają się bity mapy bitowej.

Formaty plików graficznych – mapy bitowe

Struktura nagłówka BITMAPINFO:

```
typedef struct tagBITMAPINFO { // bmi
   BITMAPINFOHEADER bmiHeader;
   RGBQUAD          bmiColors[1];
} BITMAPINFO;
```

Struktura nagłówka BITMAPINFOHEADER:

```
typedef struct tagBITMAPINFOHEADER{ // bmih
   DWORD  biSize;
   LONG   biWidth;
   LONG   biHeight;
   WORD   biPlanes;
   WORD   biBitCount;
   DWORD  biCompression;
   DWORD  biSizeImage;
   LONG   biXPelsPerMeter;
   LONG   biYPelsPerMeter;
   DWORD  biClrUsed;
   DWORD  biClrImportant;
} BITMAPINFOHEADER;
```

Formaty plików graficznych – mapy bitowe

Struktura nagłówka BITMAPINFOHEADER (interpretacja pól):

Pole	**Wielkość**	**Opis**
biSize	DWORD	Wielkość tej struktury w bajtach
biWidth	LONG	Szerokość mapy bitowej w pikselach
biHeight	LONG	Wysokość mapy bitowej w pikselach
biPlanes	WORD	Wartość 1
biBitCount	WORD	Liczba bitów koloru przypadająca na jeden piksel (1,4,8 lub 24)
biCompression	DWORD	Metoda kompresji (0, gdy nie ma kompresji)
biSizeImage	DWORD	Wielkość mapy bitowej w bajtach (potrzebne jedynie w przypadku map skompresowanych)
biXPelsPerMeter	LONG	Rozdzielczość pozioma w pikselach na metr
biYPelsPerMeter	LONG	Rozdzielczość pionowa w pikselach na metr
biClrUsed	WORD	Liczba kolorów użyta w obrazie
biClrImportant	DWORD	Liczba istotnych kolorów użyta w obrazie

Formaty plików graficznych – mapy bitowe

Struktura nagłówka RGBQUAD:

```
typedef struct tagRGBQUAD { // rgbq
    BYTE    rgbBlue;
    BYTE    rgbGreen;
    BYTE    rgbRed;
    BYTE    rgbReserved;
} RGBQUAD;
```

Pole	Wielkość	Opis
rgbBlue	BYTE	Intensywność barwy niebieskiej
rgbGreen	BYTE	Intensywność barwy zielonej
rgbRed	BYTE	Intensywność barwy czerwonej
rgbReserved	BYTE	Wartość 0

Formaty plików graficznych – mapy bitowe

Liczbę struktur RGBQUAD wyznacza zwykle pole biBitCount:

- **Dla koloru 1 bitowego: 2 struktury**
- **Dla kolorów 4 bitowych: 16 struktur**
- **Dla kolorów 16 bitowych: 256 struktur**

Gdy jednak pole biClrUsed ma wartość różną od zera to określa ono liczbę kolorów zdefiniowanych w palecie.

Po tablicy barw w pliku znajduje się obraz mapy bitowej zakodowany w następujący sposób:

- **Tablica zaczyna się od dolnego wiersza pikseli**
- **Piksel zajmuje 1, 4, 8 lub 24 bity**
  - **W przypadku monochromatycznych map bitowych, w których 1 bit koloru przypada na piksel, pierwszy piksel w każdym wierszu odpowiada najbardziej znaczącemu bitowi pierwszego bajtu w każdym wierszu. Jeśli wartość bitu wynosi 0, kolor piksela należy odczytać z pierwszej struktury RGBQUAD znajdującej się w tablicy barw. Jeżeli wartość bitu wynosi 1, to kolor jest zakodowany w drugiej strukturze RGBQUAD.**

Formaty plików graficznych – mapy bitowe

- **W przypadku 16-kolorowych map bitowych z 4 bitami koloru na piksel, pierwszy piksel w każdym wierszu odpowiada czterem najbardziej znaczącym bitom pierwszego bajtu każdego wiersza. W celu odkodowania koloru piksela używamy pobranej 4-bitowej wartości jako indeksu do tablicy barw. Tablica barw liczy 16 elementów.**
- **W 256-kolorowych mapach bitowych każdy bajt odpowiada jednemu pikselowi. Kolor piksela odczytujemy używając wartości 8-bitowej jako indeksu do tablicy barw, która liczy tym razem 256 elementów.**
- **Jeśli obraz składa się z pikseli kodowanych na 24 bitach koloru, to każdy zbiór kolejnych trzech bajtów odpowiada wartości RGB piksela. W tym przypadku nie ma tablicy barw, chyba że pole biClrUsed struktury BITMAPINFOHEADER jest różne od 0.**

W każdym przypadku, każdy wiersz danych mapy bitowej zawiera wielokrotność 4 bajtów. Jeżeli nie pasuje to do rzeczywistych wymiarów obrazu, to wiersz jest uzupełniany, aby ten warunek został spełniony.

Formaty plików graficznych – wyświetlanie map bitowych

Etapy wyświetlania plików graficznych:

- **Wczytanie zawartości pliku do pamięci komputera**
- **Przekształcenie wczytanych danych w format DIB**
- **Przekształcenie wczytanych danych w format DDB**
- **Przesłanie obrazu do okna programu**

Formaty plików graficznych – wyświetlanie map bitowych

Metoda wczytania pliku DIB:

- **Odczytanie z pliku struktury BITMAPFILEHEADER (Pole struktury o nazwie bfSize zawiera rozmiar pliku).**
- **Na podstawie danych ze struktury BITMAPFILEHEADER należy zaalokować w pamięci komputera bufor o odpowiednim rozmiarze i przekopiować do niego zawartość pliku.**
- **Ustawić uprzednio zdefiniowane wskaźniki na początek pól BITMAPFILEHEADER, BITMAPINFOHEADER, BITMAPINFO (wskaźniki zastosowane zostaną przez funkcje przetwarzające bitmapę).**
- **Z zastosowaniem funkcji CreateDIBitmap() przekształcić dane z pliku DIB w bitmapę DDB**
- **Zamknąć plik dyskowy z danymi**

Formaty plików graficznych – wyświetlanie map bitowych

Funkcja CreateDIBitmap():

```
HBITMAP CreateDIBitmap
(
HDC hdc,                        // uchwyt do kontekstu urządzenia
CONST BITMAPINFOHEADER *lpbmih,   // wskaźnik do struktury
                    // BITMAPINFOHEADER przetwarzanej DIB
DWORD fdwInit,      // flaga inicjalizacji
CONST VOID *lpbInit,// wskaźnik do początku tablicy pikseli
CONST BITMAPINFO *lpbmi, // wskaźnik do struktury BITMAPINFO
                         // przetwarzanej DIB
UINT fuUsage             // tryb konwersji kolorów
);
```

Formaty plików graficznych – wyświetlanie map bitowych

Uchwyt do bitmapy:

```
//...
static HBITMAP mHBmp=0;
//...

if(mHBmp)
{ DeleteObject(mHBmp);
  mHBmp=0;
}

HBITMAP CreateDIBitmap(//...
                       );
```

Formaty plików graficznych – wyświetlanie map bitowych

Uchwyt do kontekstu urządzenia:

```
//...
HDC mHDC;   // uchwyt do kontekstu urządzenia
//...

mHDC=GetDC(hwnd);

HBITMAP CreateDIBitmap(//...
                            mHDC,
                        //...
                       );
ReleaseDC(hwnd,mHDC);
```

Formaty plików graficznych – wyświetlanie map bitowych

Flaga inicjalizacji:

```
//...

HBITMAP CreateDIBitmap(//...
                            CBM_INIT,
                        //...
                       );
```

Formaty plików graficznych – wyświetlanie map bitowych

Tryb konwersji kolorów:

```
//...

HBITMAP CreateDIBitmap(//...
                            DIB_PAL_COLORS, // DIB_RGB_COLORS
                          //...
                         );
```

Formaty plików graficznych – wyświetlanie map bitowych

Metoda wyświetlenia bitmapy DDB:

- **Spowodować odświeżenie okna (funkcja InvalidateRect())**
- **Utworzyć kontekst urządzenia pamięciowego zgodny z kontekstem okna (funkcja CreateCompatibleDC())**
- **Przyłączyć do utworzonego kontekstu bitmapę DDB (makrorozwinięcie SelectBitmap())**
- **Przesłać bitmapę pomiędzy kontekstami (funkcja BitBlt())**
- **Odłączyć bitmapę od pamięciowego kontekstu urządzenia**
- **Usunąć kontekst urządzenia pamięciowego**

Formaty plików graficznych – wyświetlanie map bitowych

Unieważnienie zawartości okna – wymuszenie wyrysowania na nowo zawartości okna:

```
//...
InvalidateRect(hwnd,NULL,TRUE);

//...
```

Utworzenie kontekstu urządzenia pamięciowego zgodnego z kontekstem okna:

```
//...
HDCPaint=BeginPaint(hwnd,&PaintStruct);

mMemDC=CreateCompatibleDC(HDCPaint);

//...
```

Formaty plików graficznych – wyświetlanie map bitowych

Przyłączenie bitmapy DDB od pamięciowego kontekstu urządzenia:

```
//...

mHBitmapOld=SelectBitmap(mMemDC,mHBmp);

//...
```

Formaty plików graficznych – wyświetlanie map bitowych

Przesłanie bitmapy pomiędzy kontekstami

```
//...
BOOL BitBlt
(
HDC hdcDest, // uchwyt kontekstu urządzenia docelowego
int nXDest,  // współrzędna x lewego górnego rogu obrazu
             // docelowego
int nYDest, // współrzędna y lewego górnego rogu obrazu
             // docelowego
int nWidth,  // szerokość obrazu docelowego
int nHeight, // wysokość obrazu docelowego
HDC hdcSrc,  // uchwyt kontekstu urządzenia źródłowego
int nXSrc,  // współrzędna x lewego górnego rogu obrazu
             // źródłowego
int nYSrc, // współrzędna y lewego górnego rogu obrazu
            // źródłowego
DWORD dwRop  // kod operacji rastrowej np.:SRCCOPY
);
//...
```

Formaty plików graficznych – wyświetlanie map bitowych

Odłączenie bitmapy od pamięciowego kontekstu urządzenia:

```
//...
SelectBitmap(mMemDC,mHBitmapOld);

//...
```

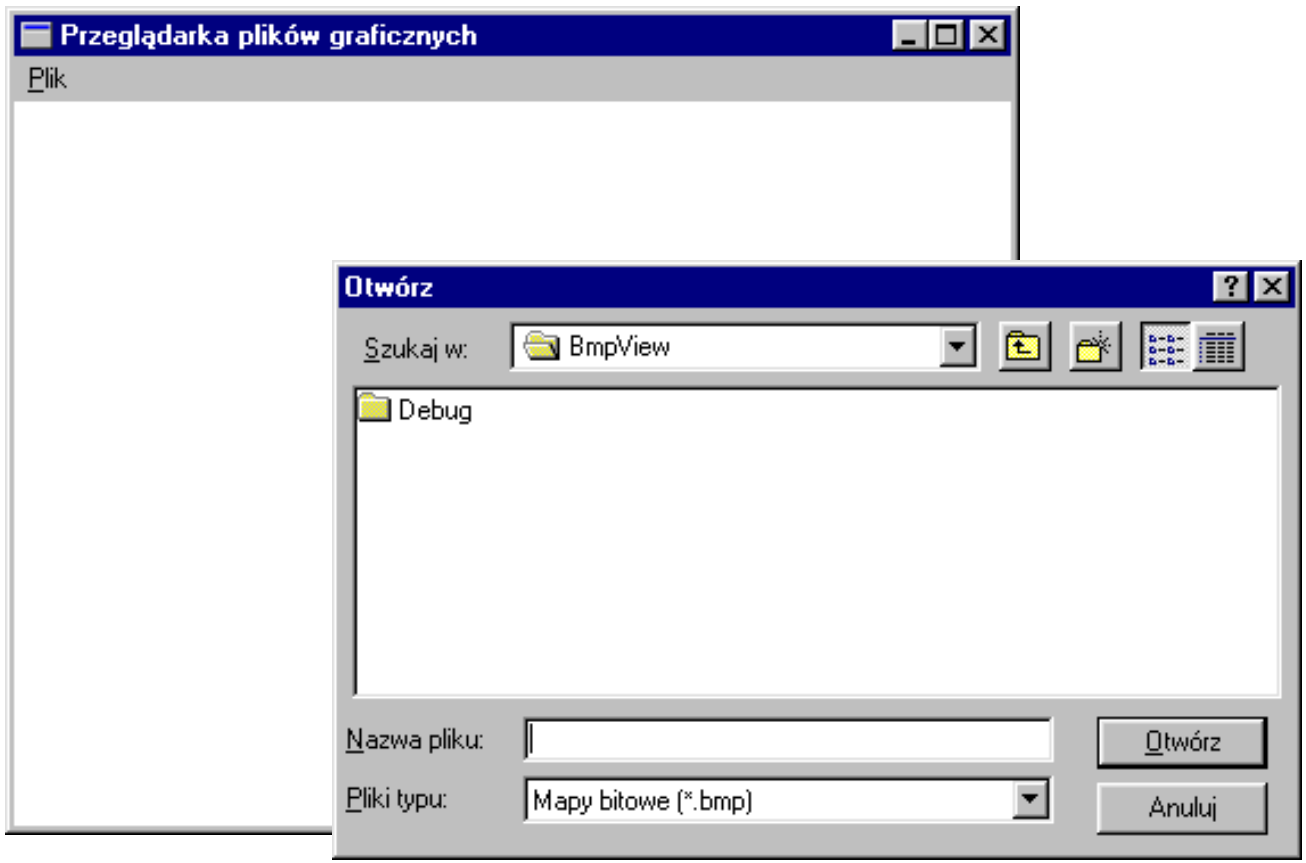
Usunięcie pamięciowego kontekstu urządzenia:

```
//...

DeleteDC(mMemDC);

//...
```

Formaty plików graficznych – program BmpView



Formaty plików graficznych – program BmpView

Pliki nagłówkowe:

```
#include <windowsx.h> // makro SelectBitmap()
#include <stdio.h>    // operacje na plikach
#include <stdlib.h>   // alokacja pamięci//...
```

Formaty plików graficznych – program BmpView

Nowe zmienne lokalne:

```
LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg,
                         WPARAM wParam,LPARAM lParam)
{HDC HDCPaint;      // uchwyt obszaru kontekstu
                    // urządzenia klienta
PAINTSTRUCT PaintStruct; // informacja o malowaniu
// Zestaw danych do obsługi standardowego okna "Otwórz plik";
char Filter[]="Mapy bitowe (*.bmp)\0*.bmp\0";
// filtr wyboru plików
char FileName[256]={'\0'};  // tablica na nazwę
                            // otworzonego pliku
OPENFILENAME OFName; // struktura obsługująca
                     // okno "Otwórz plik"
FILE *plik;           // uchwyt pliku
```

Formaty plików graficznych – program BmpView

Nowe zmienne lokalne (cd):

```
// Zestaw danych, które mogą być przydtne do przetwarzania
// i wczytywania pliku DIB
static BITMAPFILEHEADER *BmpFilHeadPtr;
static BITMAPINFOHEADER *BmpInfHeadPtr;
static BITMAPINFO       *BmpInfPtr;
static unsigned char    *BmpDataPtr;
HDC mHDC;   // uchwyt do kontekstu urządzenia
HDC mMemDC; // uchwyt do pamięciowego kontekstu urządzenia
static int mBmp_Width;
static int mBmp_Height;

// Uchwyty do bitmap DDB:
HBITMAP mHBitmapOld;
static HBITMAP mHBmp=0;
```

Formaty plików graficznych – program BmpView

Obsługa standardowego okna Otwórz:

```
case WM_COMMAND:
        switch(LOWORD(wParam))
        { case ID_PLIK_OTWORZ:// Załaduj do pamięci bitmapę
          memset(&OFName,0,sizeof(OPENFILENAME));
          OFName.lStructSize=sizeof(OPENFILENAME);
          OFName.hwndOwner=hwnd;
          OFName.lpstrFilter=Filter;
          OFName.lpstrFile=FileName;
          OFName.nMaxFile=sizeof(FileName);
          FName.Flags=OFN_FILEMUSTEXIST|OFN_HIDEREADONLY;
          OFName.lpstrDefExt="bmp";
          if(GetOpenFileName(&OFName)==FALSE)return 0;
          // Okno komunikatu:
        MessageBox(hwnd,"Za chwilę będzie ładowana do pamięci
        bitmapa","Przeglądarka...",MB_OK|MB_ICONINFORMATION);
        if(mHBmp){    DeleteObject(mHBmp);mHBmp=0;
                  }
        //...
```

Formaty plików graficznych – program BmpView

Struktura obsługi komunikatów:

```
switch(iMsg)
{//...
        case WM_COMMAND:
                switch(LOWORD(wParam))
                {case ID_PLIK_OTWORZ:
                        // Załaduj do pamięci bitmapę, utwórz DDB
                        return 0;
                case ID_PLIK_KONIEC: //...
                        return 0;
                }return 0;
        case WM_DESTROY:    //...
                return 0;
        case WM_PAINT:
                // Wyświetlaj bitmapę, jeśli została załadowana
                return 0;
        default:return DefWindowProc(hwnd,iMsg,wParam,lParam);
        }}
```