**Растр**– это матрица (массив) ячеек (пикселов).

**Растровый формат** используется для хранения растровых данных. Файлы такого типа особенно хорошо подходят для хранения изображений реального мира, например, оцифрованных фотографий. Растровые файлы содержат битовую карту изображения и се спецификацию. Наиболее распространенные растровые форматы: BMP, TIFF, GIF, PCX, JPEG.

Здесь мы рассмотрим работу с одним из самых популярных форматов, который обязан своей распространенностью операционной системе Windows – **формат bmp.**

В операционной системе Windows используются два формата растровых (битовых) изображений –**аппаратно-зависимый DDB** (devicedependent bitmap) и **аппаратно-независимый DIB** (device-independent bitmap).

Для описания растровых изображений в ОС Windows используется структура BITMAP:

Изображение выглядит как текст

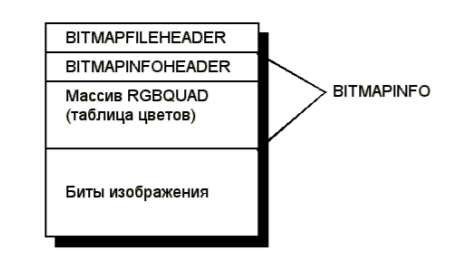
Автоматически созданное описание

Битовые изображения в формате **DDB** являются аппаратно-зависимыми. Поэтому структура изображения в оперативной памяти зависит от особенностей аппаратуры.

Если бы в Windows можно было работать только с изображениями DDB, было бы необходимо иметь отдельные наборы изображений для каждого типа видеоконтроллера и каждого видеорежима, что, очевидно, крайне неудобно.

**Аппаратно-независимое битовое изображение DIB** содержит описание цвета пикселов изображения, которое не зависит от особенностей устройства отображения. Операционная система Windows после соответствующего преобразования может отобразить такое изображение на любом устройстве вывода. Несмотря на некоторое замедление процесса вывода по сравнению с выводом изображений DDB, универсальность изображений DIB делает их весьма привлекательными для хранения изображений.

Изображения DIB, в отличие от изображений DDB, являются аппаратно-независимыми, поэтому без дополнительного преобразования их нельзя отображать на экране с помощью функций BitBlt и StretchBlt. В операционной системе Windows битовые изображения хранятся в файлах с расширением имени bmp, при этом используется аппаратно-независимый формат DIB.



* Файл, содержащий битовое изображение, начинается со структуры BITMAPFILEHEADER. Эта структура описывает тип файла и его размер, а также смещение области битов изображения.

Typedef structtagBITMAPFILEHEADER {

WORDb fType; //признак bmp-файла, 42 4D (коды букв ВМ)

DWORD bfSize; //размер файла

WORD bfReserved1; //первое резервное поле, всегда ноль

WORD bfReserved2; //второе резервное поле – тоже ноль

DWORD bfOffBits; //смещение от начала файла до первого байта графических данных

} BITMAPFILEHEADER, \*PBITMAPFILEHEADER;

* Сразу после структуры BITMAPFILEHEADER в файле следует структура BITMAPINFO, которая содержит описание изображения и таблицу цветов.

typedef struct tagBITMAPINFO {

BITMAPINFOHEADERbmiHeader; RGBQUAD bmiColors[1];

} BITMAPINFO, \*PBITMAPINFO;

* Описание изображения (размеры изображения, метод компрессии, размер таблицы цветов и т. д.) находится в структуре BITMAPINFOHEADER.

typedef struct tagBITMAPINFOHEADER{

DWORD biSize; //размер BITMAPINFOHEADER в байтах.

LONG biWidth; //ширина картинки в пикселах

LONG biHeight; //высота картинки в пикселах

WORD biPlanes; //количество битовых плоскостей (=1)

WORD biBitCount; //количество бит на пиксел (определяет максимальное число цветов в bitmap) DWORD biCompression; //способ сжатия (0 – данные не сжимаются).

DWORD biSizeImage; //размер изображения в байтах (без заголовков)

LONG biXPelsPerMeter; //число пикселов на метр по горизонтали

LONG biYPelsPerMeter; //число пикселов на метр по вертикали

DWORD biClrUsed; //количество элементов палитры хранящихся после заголовка

DWORD biClrImportant; //количество разных цветов действительно используемых в рисунке

} BITMAPINFOHEADER, \*PBITMAPINFOHEADER;

* В некоторых случаях (не всегда) в файле может присутствовать палитра–таблица цветов (как массив структур RGBQUAD), присутствующих в изображении.

Общий алгоритм работы загрузки изображений:

1. Получение передаваемого параметрного изображения.

2. Выделение памяти для воспроизведения переданного изображения.

3. Передача изображения в окно.

Общий алгоритм работы записи выделенной области изображения:

1. Получение координат левой верхней и правой точек для захвата области выделения

2. Получение информации о хранимой информации в пределах двух точек.

3. Формирование файла для собрания.

4. Сохранение файла в директорию.