# bmp File Info

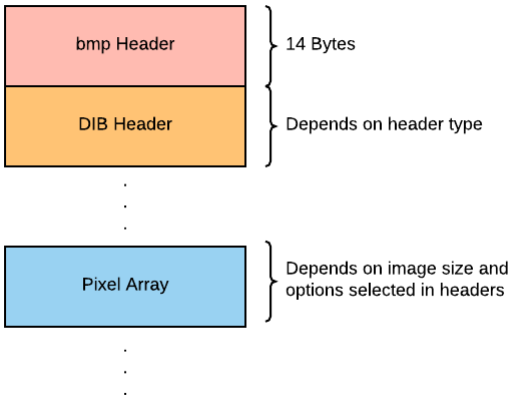
## Background:

Note: This section provides a digest of the wikipedia entry on the bmp file format which can be found here:

https://en.wikipedia.org/wiki/BMP_file_format (https://en.wikipedia.org/wiki/BMP_file_format)

The bmp image format has been in use for several decades and is a portable method of representing digital image data on virtually any modern computing system. For this lab you will build a command line tool for applying filters to simple bitmap (bmp) images. Though bmp is capable of numerous features that build upon the original specification, we will concentrate on very basic images that are comprised of two-dimensional arrays of pixels - with each pixel represented by three bytes of information. These three bytes represent the Red, Green, and Blue (RGB) color intensity values for each pixel. A value of zero indicates that the color is absent, a value of 255 indicates the maximum intensity for that color for the given pixel. By combining these three "primary color" values nearly 17 million unique colors can be described and rendered.

bmp files contain more than simply the RGB values for each pixel in an image. In order to recognize and interpret a bmp file properly, certain things must be known, such as the width and height of the image being represented. Such information is contained in a set of informative bytes located at the beginning of a bmp file. We refer to these bytes a "headers". Ultimately a valid bmp file must contain AT LEAST three things - the bmp header, a "Device-Independent Bitmap" header (DIB header for short), and the pixel array that encodes the image.

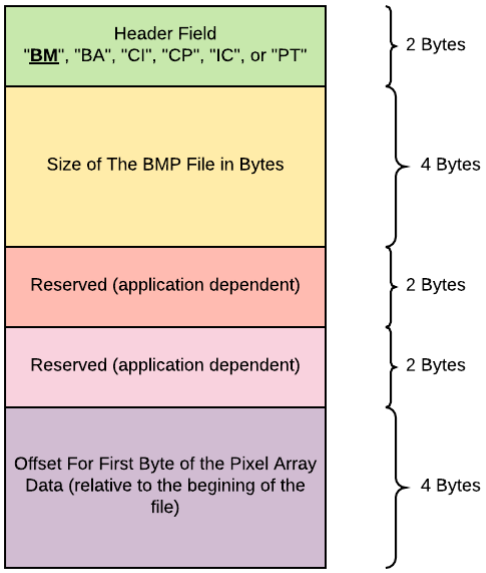



Let's look at each of these three items...

## Bitmap Header:

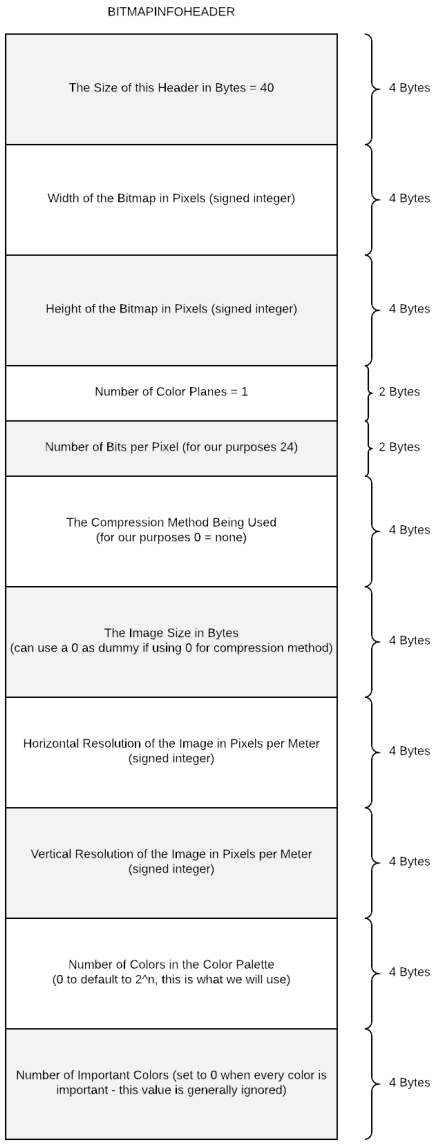
The bitmap header consists of 14 bytes divided into five fields as shown below. The header field specifies the type of bitmap the file contains. For our purposes these two bytes will always contain

the ascii characters "BM". The next two fields that we are concerned with are the size of the file in bytes, and the offset of the first byte in the pixel array. In this lab we will not concern ourselves with the contents of the reserved fields.




## DIB Header:

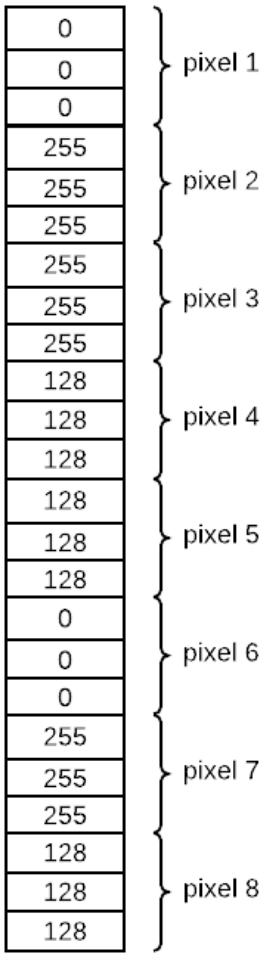
In addition to the bitmap file header, a valid bmp must contain a "Device-Independent Bitmap" header (DIB header for short). This header is utilized by an application to properly interpret the bytes of the file that encode the image. Though there are several different DIB header formats to choose from, for this lab we will limit ourselves to the BITMAPINFOHEADER. It's fields, their purposes, and our usage of them is outlined in the following image...

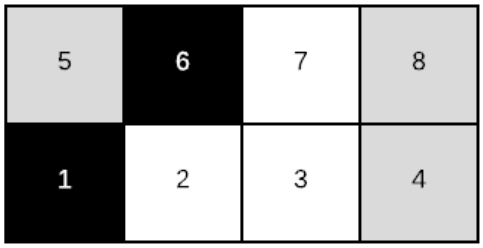



# Pixel Array:

The final piece of the bmp file format that you will need to be able to work with for this lab is the pixel array. This is where the information describing the pixels in the image are located. As noted earlier, the images we will be working with represent single pixels with trios of bytes that store numeric values between 0 and 255 that indicate the intensity of the red, green, and blue components of the pixel's color. Though we often refer to these values as RGB - please note that in the bmp format they are actually stored in the order Blue, Green, Red - probably just to keep life interesting...

Pixels are stored row by row, **starting in the lower left corner** of the image reading **left to right** and then **bottom to top** as shown in the following figure.

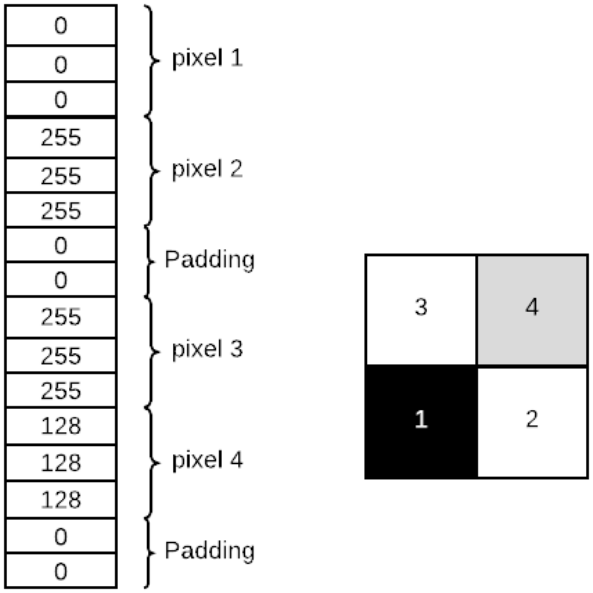



## Pixel Row Padding:

An important point to note about the pixel array is that each row stored in the file is required to occupy a multiple of 4 bytes. If the number of bytes needed to represent the pixels of a row does not add up to a multiple of 4 bytes then padding (extra bytes that are not part of the image) must be inserted. Thus, if an image were 8 pixels wide (as shown above), 3 X 8 = 24 bytes, which IS a multiple of 4 so no padding would be needed.

On the other hand if an image were 2 pixels wide then 3 X 2 = 6 bytes, which is NOT a multiple of 4, so padding is needed. In this case two extra bytes would need to inserted (as shown in the image below), bringing the number to 8. The value stored in these bytes isn't important and can be set to zero.




You can read more about the bmp standard here: **https://en.wikipedia.org/wiki/BMP_file_format (https://en.wikipedia.org/wiki/BMP_file_format)**