SAE Image de Quentin Béatrix

A.0

Expliquez ces valeurs.

Une image BMP est toujours composée de :

- Le header BMP
- Le header DIB
- Les pixels

Le header BMP est codée avec 14 octets comporte :

- 0-1 : le type du fichier qui peut être traduit avec le code ASCII
- 2-5 : la taille du fichier en octets
- 6-9 : un champ réservé pour autre chose (dans le futur surtout)
- 10-13 : l'offset des pixels, c'est à dire quand commence les pixels

Dans notre cas, le type du fichier est 42 4D traduit avec le code ASCII qui est égale à BM.

La taille du fichier est de 00 0C 73 99 (en little-endian), égale à 816 025 octets.

L'offset des pixels commence à 00 00 00 1A (en little-endian), c'est à dire à l'octets 26.

Le header DIB commence toujours par la taille de celui-ci en 4 octets : 00 00 00 0C, c'est à dire 12 octets.

Le header DIB de 12 octets comporte :

- 0-3 : la taille du header DIB
- 4-5 : la largeur en pixels
- 6-7 : la longueur en pixels
- 8-9 : le nombre de plans de couleurs (presque toujours 1)
- 10-11 : le nombre de bits utilisé pour chaque pixel

Dans notre cas, la largeur en pixels est de 02 80 (en little-endian), égale à 640 pixels.

La longueur en pixels est de 01 A9 (en little-endian), égale à 425 pixels.

Le nombre de pixels est de 00 18 (en little-endian), égale à 24, donc 3 octets pour représenter la couleur en RGB mais BGR en little-endian.

Proposez une solution pour la corriger et faire en sorte qu'elle ne s'affiche plus.

La taille du fichier indiquée n'est pas la bonne, elle est égale à 816 025 alors que le fichier fait en réalité 816 026 octets.

Je propose donc je juste de changer la valeur de la taille du fichier.

A.1

Avec Okteta saisir de l'hexadécimal pour créer une image bmp de type Windows 2.0:

L'image était assez simple à faire, il fallait alterner le rouge (00 00 FF) et le blanc (FF FF FF) de gauche à droite et de bas en haut.

A.2

Modification d'image

Il suffit de modifier l'image avec les couleurs en se référant à la page indiqué.

Cyan : 00 FF FF
Magenta : FF 00 FF
Bleu céruléen : E8 9D 0F
Blanc : FF FF FF
Rouge : 00 00 FF
Bleu : FF 00 00
Vert : 00 FF 00

A.3

Un Fichier BMP d'un type plus récent.

Avec la commande : >convert Image0.bmp bmp3:Image1.bmp

Quelle est le poids alors de notre image?

Si on regarde l'Image0, on voit qu'elle prend que 74 octets et que si le header DIB de 12 octets est remplacé par un header DIB de 40 octets, on a alors une différence de 28 octets que l'on ajoute, ce qui nous donne : 74 + 28 = 102 octets.

Combien y-a-t-il de bits par pixel?

Il y a toujours 24 bits par pixels car le RGB utilise 3 octets (3 * 8 = 24) donc de 0 à 255.

Quelle est la taille des données pixels?

Il y a toujours 48 octets car si il y a 3 octets par pixels et qu'il y a 4 lignes et 4 colonnes de pixels, ça nous donne : 3 * 4 * 4 = 48 octets.

Y a-t-il une compression utilisée?

Non, il y a pas de compression utilisé car les 4 octets de compression sont égale à 0.

Le codage des pixels a-t-il changé?

Non, elle n'a pas changé, si on regarde bien les 2 offsets des pixels, elles sont identiques donc le codage des pixels est le même.

A.4

Un Fichier BMP avec index de couleurs.

Avec la commande : >convert Image1.bmp -colors 2 bmp3:Image2.bmp

Combien y-a-t-il de bits par pixel?

"-colors 2" signifie qu'il n'y a que 2 couleurs uniquement.

On a 2 couleurs, donc on a uniquement besoin d'un seul bit.

Quelle est la taille des données pixels?

Il y a 4 pixels par ligne donc 4 bits par ligne mais la ligne est arrondie à l'octet près, donc on a un octet mais BMP impose de faire une ligne avec un multiple de 4 octets donc : 4 lignes * 4 octets = 16 octets

Y a-t-il une compression utilisée?

Non, il y a pas de compression utilisé car les 4 octets de compression sont égale à 0.

Comment sont codées les couleurs de la palette?

Elles sont codées après le header DIB en utilisant 4 octets cette fois-ci : Bleu - vert - Rouge - Alpha (Un octet réservé (souvent égale à 0)).

Quel est le nombre de couleurs dans la palette?

Il y a uniquement 2 couleurs dans la palette car le nombre de couleur utilisé est indiqué et il est égale à 00 00 00 02 = 2.

Le codage des pixels a-t-il changé?

oui, maintenant il suffit que d'un seul bit pour coder un pixel, par exemple 50 00 00 00 = 0101 0000 0000 0000 0000 0000 0000 0000, le rouge a été codé en premier donc il est égale au 0 et donc le blanc est égale au 1. Donc ca apparait de cette façon :
Rouge - Blanc - Rouge - Blanc
Les 0 après sont uniquement là pour completer le multiple de 4 octets.

Changez la couleur rouge des pixels en bleu pour obtenir l'image ci-dessous que vous nommerez ImageBleue.bmp.

Il suffit de changer la couleur de la palette pour pouvoir transformer tous les rouge en bleu.

Donc on passe de 00 00 FF 00 pour le rouge à FF 00 00 00 pour le bleu.

Inversez le damier : les blancs à la place des bleus et les bleus à la place des blancs, pour obtenir l'image ci-dessous.

Il suffit d'intervertir les A0 et le 50 pour pouvoir inverser le damier.

Modifiez le fichier en mode index de couleurs avec okteta de façon à obtenir ceci. Enregistrez cette image sous ce nom Image3.bmp:

Il suffit de jouer avec les 1 et les 0 et mettre le résultat en hexadécimal.

Passez le fichier de l'ancien logo du Département d'Informatique en mode index de couleurs:

Avec la commande : > convert ImageExemple.bmp -colors 16 bmp3:ImageExempleIndexBMP3_16.bmp

A quelle adresse peut-on trouver le nombre de couleurs qu'il y a dans la palette?

Le nombre de couleurs dans la palette se trouve à l'adresse 0x0000002E.

A quelle adresse dans la palette peut-on trouver la couleur à dominante "Blanc" utilisée par cette image?

La couleur à dominante "Blanc" utilisée par cette image se situe à l'adresse 0x00000066.
Elle est égale à FE FE FD en BGR.

Où commence le tableau de pixel?

Le tableau (ou offset) de pixel commence à l'adresse 0x00000076.

En modifiant l'Hex, placez quelques pixels bleus tout en bas de l'image.

Il suffit de changer les premiers pixels en remplacant par "EE" les "CC".

Que se passe-t-il si l'on diminue le nombre de couleurs dans la palette? Que se passe t-il d'un point de vue visuel? Et dans l'hexa?

Avec l'aide de la commande : convert images/ImageExemple.bmp -colors 4 bmp3:ImageExempleIndexBMP3_4.bmp

D'un point de vue visuel, il ne semble rester uniquement 4 couleurs possibles dans l'image.

Du point de vue de l'hexadécimal, il ne reste que 4 couleurs dans la palette de couleur, le reste de la palette de couleur est remplacer par des 0.

Par contre, les couleurs sont toujours codées de la même façon, avec 4 bits, mais n'ayant que 4 valeurs possibles, les pixels ne peuvent pas dépasser 3.

A.5

Utilisation des négatifs

Changez dans l'entête du fichier la valeur de la hauteur de l'image. Elle est à l'origine de valeur 4 pixels, changez pour la valeur négative de -4 pixels. Que ce passe-t-il?

L'image se retourne. C'est du C2, donc il suffit d'inverser les bits et de rajouter +1 : 04 00 00 00 = FC FF FF FF

Profitez de cette information pour obtenir facilement à partir de ImageExempleIndexBMP3_16.bmp cette image là:

C'est la même opération mais avec des valeurs différentes.

A.6

Quel est le poids du fichier? Pourquoi? Que c'est-il passé?

Le poids du fichier est de 1120 octets, il a donc grossit.

Il a grossit même si on l'a compressé car c'est une très petite image donc avec peut de pixels consécutif de la même couleur donc il n'y a pas grand chose à compresser. Et pour tous les 00, c'est la palette de couleur qui est définie comme ca, c'est le maximum de couleur qu'on peut avoir, mais comme on ne l'a pas réduit, le fichier deviens gros.

Trouvez dans l'entête l'offset que donne l'adresse de début des pixels.

L'adresse de début de pixel est à l'adresse 0x00000436.

Décodez le code des pixels. (C'est-a-dire essayez de retrouver dans l'hexadécimal le codage des pixels et expliquez-le)

00 00 = fin de ligne
00 01 = fin de l'image

Et pour le reste, si on 01 puis 00, c'est qu'on met une fois la couleur 00 (qu'on a définie dans la palette de couleur), si on 03 01 on met 3 pixels de couleur 01.

A.7

Avec la commande : >convert Image3.bmp -colors 2 -compress RLE bmp3:Image5.bmp

Quel est le poids du fichier Image5.bmp? Pourquoi est-il moins grand que celui de l'image Image4.bmp?

Le poids du fichier Image5.bmp est de 44E = 1102 octets.

Il est moin grand car il y a des pixels de même couleurs consécutifs donc le RLE peut compresser.

Décodez le code des pixels.

04 01 00 00 04 00 00 00 04 00 00 00 01 01 01 00 01 01 00 00 00 01

Première ligne : 04 01 00 00 = Il y a 4 pixels de couleur 01 donc blanc, puis un saut de ligne avec 00
Deuxième ligne : 04 00 00 00 = Il y a 4 pixels de couleur 00 donc rouge, puis un saut de ligne avec 00
Troisième ligne : 04 00 00 00 = Il y a 4 pixels de couleur 00 donc rouge, puis un saut de ligne avec 00
Dernière ligne : 01 01 01 00 01 01 00 00 00 01 = Il y a 1 pixel de couleur 01 donc blanc puis 1 rouge puis 1 blanc puis un rouge, et pour finir un saut de ligne avec 00 et une fin d'image avec 01

A.8

Modifiez le fichier Image5.bmp afin d'obtenir cette image que vous nommerez Image6.bmp .

A.9

Modifiez le fichier Image6.bmp afin d'obtenir cette image que vous nommerez Image7.bmp .

A.10

Modifiez le fichier Image7.bmp afin d'obtenir cette image que vous nommerez Image8.bmp .

Modifiez le fichier à la main et en ramenant le nombre de couleur encodés dans la table d'index à 4.

Il faut juste enlever les 0 et changer l'adresse de l'offset des pixels.

B.1

Avec ces éléments créez un programme python qui transpose une image:

Code dans le fichier "SAEImagePython.py".

B.2

Créez un programme python qui inverse une image dans un miroir:

Code dans le fichier "SAEImagePython.py".

B.3

Créez un programme python qui passe cette image (Logo IUT) là en niveaux de gris:

Code dans le fichier "SAEImagePython.py".

B.4

Créez un programme python qui passe cette image (Logo IUT) là en noir et blanc:

Code dans le fichier "SAEImagePython.py".

B.5

Créez un programme python qui cache le logo en noir et blanc dans l'image hall-mod_0.bmp :

Je ne n'ai pas compris la question.