Projet 1: Librairie de gestion d'images

1.0

Generated by Doxygen 1.8.18

Chapter 1

Projet 1: Librairie de gestion d'images

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 PNM_t Struct Reference

Data Fields

- int **magicNumber**
- int **columns**
- int **rows**
- int **maxValuePixel**
- int ∗∗ **matrix**

4.1.1 Detailed Description

pnm.c

Ce fichier contient les définitions de types et les fonctions de manipulation d'images PNM.

Author

: Dumoulin Peissone S193957

Date

: 19/02/21 @projet: INFO0030 Projet 1 Définition du type opaque PNM

The documentation for this struct was generated from the following file:

- /home/peps/Desktop/Univ/Projet de programmation/Projets/Projet 1/Projet1/pnm.c

Chapter 5

File Documentation

5.1 /home/peps/Desktop/Univ/Projet de programmation/Projets/Projet 1/Projet1/pnm.h File Reference

Librairie pour gérer les fichiers d'extension pnm (.pbm, .pgm, .ppm)

Typedefs

- typedef struct PNM_t **PNM**

Functions

- int load_pnm (PNM **image, char *filename)
  Charge une image PNM depuis un fichier.
- int **write_pnm** (PNM *image, char *filename)
- PNM * create_pnm (void)
  Crée et alloue dynamiquement une variable de type opaque PNM.*
- void destroy_pnm (PNM *image)
  *Libère la mémoire allouée par *create_pnm.*
- int get_magicNumber (PNM *image)
  Accesseur en lecture pour le champ magicNumber de image.*
- int get_columns (PNM *image)
  Accesseur en lecture pour le champ columns de image.*
- int get_rows (PNM *image)
  Accesseur en lecture pour le champ rows de image.*
- int get_maxValuePixel (PNM *image)
  Accesseur en lecture pour le champ maxValuePixel de image.*
- int ** get_matrix (PNM *image)
  Accesseur en lecture pour le champ matrix de image.*
- PNM * set_magicNumber (PNM *image, int magicNumber)
  *Accesseur en écriture pour le champ magicNumber de *image.*
- PNM * set_columns (PNM *image, int columns)
  *Accesseur en écriture pour le champ columns de *image.*
- PNM * set_rows (PNM *image, int rows)

Accesseur en écriture pour le champ rows de ∗image.

- PNM ∗ set_maxValuePixel (PNM ∗image, int maxValuePixel)

  Accesseur en écriture pour le champ maxValuePixel de ∗image.
- PNM ∗ set_matrix (PNM ∗image, int ∗∗matrix)

  Accesseur en écriture pour le champ matrix de ∗image.
- int ∗∗ create_matrix (PNM ∗image)

  Crée et alloue dynamiquement la matrice de ∗image.
- int load_matrix (PNM ∗image, FILE ∗fp)

  Lecture dans un fichier et remplissage de la matrice de ∗image.
- int write_matrix (PNM ∗image, FILE ∗fp)

  Ecriture de la matrice de ∗image dans un fichier.
- void destroy_matrix_rows (PNM ∗image)

  Libère la mémoire allouée par ∗∗create_matrix.
- void destroy_matrix_columns (PNM ∗image)

  Libère la mémoire allouée par ∗∗create_matrix.
- void destroy_all (PNM ∗image)

  Libère toute la mémoire allouée.
- int manage_comments (FILE ∗fp)

  Permet de gérer une ligne pour savoir si on doit l'ignorer (celles commençant par '#')
- int short_options (int argc, char ∗∗argv, char ∗∗format, char ∗∗input, char ∗∗output)

  Gère les options courtes passées en argument du programme.
- int manage_format_input (PNM ∗image, char ∗format, char ∗input)

  Gère si le format correspond au format de l'input.
- int verify_output (PNM ∗image, char ∗output)

  Vérifie si l'output contient des caractères spéciaux interdits.

5.1.1 Detailed Description

Librairie pour gérer les fichiers d'extension pnm (.pbm, .pgm, .ppm)

pnm.h

Ce fichier contient les déclarations de types et les prototypes des fonctions pour la manipulation d'images PNM.

Author

: Dumoulin Peissone S193957

Date

: 22/02/21 @projet: INFO0030 Projet 1

Author

Peissone Dumoulin - Université de Liège

Version

1.0

Date

19/02/2021

Déclaration du type opaque PNM

5.1.2 Function Documentation

5.1.2.1 create_matrix()

```
int** create_matrix (
            PNM * image )
```

Crée et alloue dynamiquement la matrice de ∗image.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: matrice allouée

Returns

: image->matrix Succès NULL Erreur lors de l'allocation dynamique

5.1.2.2 create_pnm()

```
PNM* create_pnm ( )
```

Crée et alloue dynamiquement une variable de type opaque PNM∗.

Parameters

/	

Precondition

: /

Postcondition

: ∗image alloué

Returns

: image Succès NULL Erreur lors de l'allocation dynamique

5.1.2.3 destroy_all()

```
void destroy_all (
            PNM * image )
```

Libère toute la mémoire allouée.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: la matrice colonne, la matrice ligne ainsi que ∗image sont libérés

Returns

: /

5.1.2.4 destroy_matrix_columns()

```
void destroy_matrix_columns (
            PNM * image )
```

Libère la mémoire allouée par ∗∗create_matrix.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: la matrice colonne est libérée

Returns

: /

5.1.2.5 destroy_matrix_rows()

```
void destroy_matrix_rows (
            PNM * image )
```

Libère la mémoire allouée par ∗∗create_matrix.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: la matrice ligne est libérée

Returns

: /

5.1.2.6 destroy_pnm()

```
void destroy_pnm (
            PNM * image )
```

Libère la mémoire allouée par ∗create_pnm.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: ∗image libéré

Returns

: /

5.1.2.7 get_columns()

```
int get_columns (
            PNM * image )
```

Accesseur en lecture pour le champ columns de image∗.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: accès en lecture au champ columns de ∗image

Returns

: image->columns Succès

5.1.2.8 get_magicNumber()

```
int get_magicNumber (
            PNM * image )
```

Accesseur en lecture pour le champ magicNumber de image∗.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: accès en lecture au champ magicNumber de ∗image

Returns

: image->magicNumber Succès

5.1.2.9 get_matrix()

```
int** get_matrix (
            PNM * image )
```

Accesseur en lecture pour le champ matrix de image∗.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: accès en lecture au champ matrix de ∗image

Returns

: image->matrix Succès

5.1.2.10 get_maxValuePixel()

```
int get_maxValuePixel (
            PNM * image )
```

Accesseur en lecture pour le champ maxValuePixel de image∗.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: accès en lecture au champ maxValuePixel de ∗image

Returns

: image->getMaxValuePixel Succès

5.1.2.11 get_rows()

```
int get_rows (
            PNM * image )
```

Accesseur en lecture pour le champ rows de image∗.

Parameters

image	un pointeur sur PNM

Precondition

: image != NULL

Postcondition

: accès en lecture au champ rows de ∗image

Returns

: image->rows Succès

5.1.2.12 load_matrix()

```
int load_matrix (
            PNM * image,
            FILE * fp )
```

Lecture dans un fichier et remplissage de la matrice de ∗image.

Parameters

image	un pointeur sur PNM
fp	un pointeur sur FILE

Precondition

: image != NULL, fp != NULL

Postcondition

: matrice chargée

Returns

: 0 Succès -3 Le contenu du fichier en input est mal formé (magicNumber)

5.1.2.13 load_pnm()

```
int load_pnm (
            PNM ** image,
            char * filename )
```

Charge une image PNM depuis un fichier.

Parameters

image	l'adresse d'un pointeur sur PNM à laquelle écrire l'adresse de l'image chargée.
filename	le chemin vers le fichier contenant l'image.

Precondition

: image != NULL, filename != NULL

Postcondition

: image pointe vers l'image chargée depuis le fichier.

Returns

: 0 Succès -1 Erreur à l'allocation de mémoire -2 Nom du fichier malformé -3 Contenu du fichier malformé

5.1.2.14 manage_comments()

```
int manage_comments (
            FILE * fp )
```

Permet de gérer une ligne pour savoir si on doit l'ignorer (celles commençant par '#')

Parameters

fp	un pointeur sur FILE

Precondition

: fp != NULL

Postcondition

: la ligne est correctement ignorée

Returns

: 0 Succès -1 Echec

5.1.2.15 manage_format_input()

```
int manage_format_input (
            PNM * image,
            char * format,
            char * input )
```

Gère si le format correspond au format de l'input.

Parameters

image	un pointeur sur PNM
format	le format du fichier
input	le nom du fichier en entrée

Precondition

: image != NULL, format != NULL, input != NULL

Postcondition

: format du fichier géré correctement

Returns

: 0 Succès -1 Mauvais format passé en argument

5.1.2.16 set_columns()

```
PNM* set_columns (
            PNM * image,
            int columns )
```

Accesseur en écriture pour le champ columns de ∗image.

Parameters

image	un pointeur sur PNM
columns	nombre de pixels de hauteur

Precondition

: image != NULL

Postcondition

: accès en écriture au champ columns de ∗image

Returns

: image Succès

5.1.2.17 set_magicNumber()

```
PNM* set_magicNumber (
            PNM * image,
            int magicNumber )
```

Accesseur en écriture pour le champ magicNumber de ∗image.

Parameters

image	un pointeur sur PNM
magicNumber	nombre qui caractérise le type de fichier (1 pour pbm, 2 pour pgm, 3 pour ppm)

Precondition

: image != NULL

Postcondition

: accès en écriture au champ magicNumber de ∗image

Returns

: image Succès

5.1.2.18 set_matrix()

```
PNM* set_matrix (
            PNM * image,
            int ** matrix )
```

Accesseur en écriture pour le champ matrix de ∗image.

Parameters

image	un pointeur sur PNM
matrix	matrice contenant la valeur de chaque pixel de l'image

Precondition

: image != NULL

Postcondition

: accès en écriture au champ matrix de ∗image

Returns

: image Succès

5.1.2.19 set_maxValuePixel()

```
PNM* set_maxValuePixel (
            PNM * image,
            int maxValuePixel )
```

Accesseur en écriture pour le champ maxValuePixel de ∗image.

Parameters

image	un pointeur sur PNM
maxValuePixel	valeur maximale que peut prendre un pixel

Precondition

: image != NULL

Postcondition

: accès en écriture au champ maxValuePixel de ∗image

Returns

: image Succès

5.1.2.20 set_rows()

```
PNM* set_rows (
            PNM * image,
            int rows )
```

Accesseur en écriture pour le champ rows de ∗image.

Parameters

image	un pointeur sur PNM
rows	nombre de pixels de largeur

Precondition

: image != NULL

Postcondition

: accès en écriture au champ rows de ∗image

Returns

: image Succès

5.1.2.21 short_options()

```
int short_options (
            int argc,
            char ** argv,
            char ** format,
            char ** input,
            char ** output )
```

Gère les options courtes passées en argument du programme.

Parameters

argc	le nombre d'arguments
argv	un tableau d'arguments
format	le format du fichier
input	le nom du fichier en entrée
output	le nom du fichier en sortie

Precondition

: argv != NULL

Postcondition

: options courtes gérées correctement

Returns

: 0 Succès -1 Option inconnue -2 Argument manquant

5.1.2.22 verify_output()

```
int verify_output (
            PNM * image,
            char * output )
```

Vérifie si l'output contient des caractères spéciaux interdits.

Parameters

image	un pointeur sur PNM
output	le nom du fichier en sortie

Precondition

: image != NULL, output != NULL

Postcondition

: format du fichier géré correctement

Returns

: 0 Succès -1 Caractère invalide dans le nom du fichier

5.1.2.23 write_matrix()

```
int write_matrix (
            PNM * image,
            FILE * fp )
```

Ecriture de la matrice de ∗image dans un fichier.

Parameters

image	un pointeur sur PNM
fp	un pointeur sur FILE

Precondition

: image != NULL, fp != NULL

Postcondition

: matrice

Returns

: 0 Succès -2 La matrice n'a pas pu être écrite dans le fichier

Index