Cours sur la MiniLibX

La MiniLibX est une bibliothèque graphique simple qui permet de créer des fenêtres, afficher des images, et gérer les événements clavier/souris. Elle est couramment utilisée dans les projets d'apprentissage pour introduire à la programmation graphique.

1. Initialisation et fenêtres

1.1. mlx_init()

Prototype : void *mlx_init(void);

Description : Initialise la connexion au système graphique (X11 sous Linux).

Retour : Un pointeur à utiliser pour les fonctions suivantes, ou NULL en cas d'échec.

Exemple :

```
    void *mlx = mlx_init();
    if (!mlx)
        return (1);
```

1.2. mlx_new_window()

Prototype : void *mlx_new_window(void *mlx_ptr, int width, int height, char *title);

Description : Crée une nouvelle fenêtre.

Arguments :

- mlx_ptr : Pointeur retourné par mlx_init.
- width et height : Dimensions de la fenêtre en pixels.
- title : Titre de la fenêtre.

Retour : Un pointeur pour la fenêtre ou NULL en cas d'échec.

Exemple :

```
    void *win = mlx_new_window(mlx, 800, 600, "Ma Fenêtre");
```

```
    if (!win)
        return (1);
```

2. Gestion des événements

2.1. mlx_loop()

Prototype : int mlx_loop(void *mlx_ptr);

Description : Démarre une boucle d'événements pour gérer les interactions (clavier, souris, etc.).

Arguments :

- mlx_ptr : Pointeur retourné par mlx_init.

Exemple :

```
    mlx_loop(mlx);
```

2.2. mlx_hook()

Prototype : int mlx_hook(void *win_ptr, int event, int mask, int (*funct)(), void *param);

Description : Associe une fonction callback à un événement spécifique.

Arguments :

- win_ptr : Pointeur retourné par mlx_new_window.
- event : Code de l'événement (ex. : 2 pour les touches clavier).
- mask : Masque des événements (souvent ignoré).
- funct : Pointeur vers une fonction callback.
- param : Pointeur passé à la callback.

Exemple : Fermer la fenêtre avec la touche ESC.

```
    int close_window(int key, void *param)
    {
        if (key == 65307) // Code de la touche ESC
            exit(0);
```

```
        return (0);
    }
    mlx_hook(win, 2, 0, close_window, NULL);
```

3. Images et pixels

3.1. mlx_new_image()

Prototype : void *mlx_new_image(void *mlx_ptr, int width, int height);

Description : Crée une nouvelle image en mémoire.

Arguments :

- mlx_ptr : Pointeur retourné par mlx_init.
- width et height : Dimensions de l'image.

3.2. mlx_get_data_addr()

Prototype : char *mlx_get_data_addr(void *img_ptr, int *bits_per_pixel, int *size_line, int *endian);

Description : Retourne un pointeur vers les données de l'image.

Arguments :

- img_ptr : Image créée par mlx_new_image.
- bits_per_pixel : Nombre de bits par pixel.
- size_line : Nombre d'octets par ligne d'image.
- endian : Ordre des octets (endianness).

3.3. mlx_put_image_to_window()

Prototype : int mlx_put_image_to_window(void *mlx_ptr, void *win_ptr, void *img_ptr, int x, int y);

Description : Affiche une image dans la fenêtre.

Arguments :

- mlx_ptr : Pointeur retourné par mlx_init.
- win_ptr : Pointeur retourné par mlx_new_window.

- img_ptr : Image créée par mlx_new_image.

- x et y : Coordonnées où afficher l'image.