НАЗВАНИЕ

MiniLibX - управление изображениями

СИНОПСИС

```
void      *mlx_new_image ( void *mlx_ptr, int width, int height );

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

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

unsigned int      mlx_get_color_value ( void *mlx_ptr, int color );

void      *mlx_xpm_to_image ( void *mlx_ptr, char **xpm_data, int *width, int *height );

void      *mlx_xpm_file_to_image ( void *mlx_ptr, char *filename, int *width, int *height );

int       mlx_destroy_image ( void *mlx_ptr, void *img_ptr );
```

ОПИСАНИЕ

mlx_new_image() создает новое изображение в памяти. Она возвращает идентификатор void*, необходимый для дальнейшей работы с этим изображением. Для этого требуется только размер изображения, которое нужно создать, используя параметры ширины и высоты и идентификатор соединения mlx_ptr.

Пользователь может рисовать внутри изображения и может в любое время сбросить изображение внутри указанного окна, чтобы отобразить его на экране.

Это делается с помощью mlx_put_image_to_window(). Здесь необходимы три идентификатора для подключения к дисплею, используемому окну и изображению (соответственно, mlx_ptr, win_ptr и img_ptr). Координаты (x, y) определяют, где изображение должно быть размещено в окне.

mlx_get_data_addr() возвращает информацию о созданном изображении, позволяя пользователю изменить его позже. Параметр img_ptr указывает изображение для использования. Три следующих параметра должны быть адресами трех разных допустимых целых чисел. bits_per_pixel будет заполнен количеством битов, необходимых для представления цвета пикселя (также называемого глубиной изображения). size_line - количество байтов, используемых для хранения одной строки изображения в памяти. Эта информация необходима для перемещения от одной строки к другой на изображении. endian говорит вам, нужно ли сохранять цвет пикселя в изображении с прямым порядком байтов (endian == 0) или с прямым порядком байтов (endian == 1).

mlx_get_data_addr возвращает адрес char*, который представляет начало области памяти, в которой хранится изображение. Исходя из этого адреса, первые биты bits_per_pixel представляют цвет первого пикселя в первой строке изображения. Вторая группа битов bits_per_pixel

представляет второй пиксель первой строки и так далее. Добавьте size_line к адресу, чтобы получить начало второй строки. Таким образом, вы можете достичь любых пикселей изображения.

mlx_destroy_image уничтожает данное изображение (img_ptr).

ХРАНЕНИЕ ЦВЕТА ВНУТРИ ИЗОБРАЖЕНИЙ

В зависимости от дисплея количество битов, используемых для хранения цвета пикселя, может изменяться. Пользователь обычно представляет цвет в режиме RGB, используя один байт для каждого компонента (см. Руководство по mlx_pixel_put). Это должно быть переведено, чтобы соответствовать требованию bit_per_pixel для изображения, и сделать цвет понятным для XServer.

Это цель функции mlx_get_color_value(). Она принимает стандартный параметр цвета RGB и возвращает значение типа unsigned int.

Наименьшие значащие биты этого значения bits_per_pixel могут быть сохранены в изображении. Имейте в виду, что позиция младших битов зависит от порядкового номера локального компьютера. Если порядковый номер образа (фактически, порядковый номер компьютера X-Server) отличается от локального порядкового номера, то значение должно быть преобразовано перед использованием.

XPM ИЗОБРАЖЕНИЯ

Функции mlx_xpm_to_image () и mlx_xpm_file_to_image () будут создавать новое изображение таким же образом. Они заполняют его, используя указанные xpm_data или имя файла, в зависимости от того, какая функция используется. Обратите внимание, что MiniLibX не использует стандартную библиотеку Xpm для работы с изображениями xpm. Возможно, вы не сможете прочитать все типы изображений xpm. Однако обрабатывает прозрачность.

ВОЗВРАЩАЕМЫЕ ЗНАЧЕНИЯ

Три функции, которые создают изображения, mlx_new_image (), mlx_xpm_to_image () и mlx_xpm_file_to_image (), вернет NULL, если произойдет ошибка. В противном случае они возвращают ненулевой указатель в качестве идентификатора изображения