NAME

MiniLibX - Handle events

SYNOPSYS

int
mlx_loop (*void *mlx_ptr*);

int
mlx_key_hook (*void *win_ptr, int (*funct_ptr)(), void *param*);

int
mlx_mouse_hook (*void *win_ptr, int (*funct_ptr)(), void *param*);

int
mlx_expose_hook (*void *win_ptr, int (*funct_ptr)(), void *param*);

int
mlx_loop_hook (*void *mlx_ptr, int (*funct_ptr)(), void *param*);

EVENTS

Both X-Window and MacOSX graphical systems are bi-directionnal. On one hand, the program sends orders to the screen to display pixels, images, and so on. On the other hand, it can get information from the keyboard and mouse associated to the screen. To do so, the program receives "events" from the keyboard or the mouse.

DESCRIPTION

To receive events, you must use **mlx_loop** (). This function never returns. It is an infinite loop that waits for an event, and then calls a user-defined function associated with this event. A single parameter is needed, the connection identifier *mlx_ptr* (see the **mlx manual).**

You can assign different functions to the three following events:
- A key is pressed
- The mouse button is pressed
- A part of the window should be re-drawn (this is called an "expose" event, and it is your program's job to handle it).

Each window can define a different function for the same event.

The three functions **mlx_key_hook** (), **mlx_mouse_hook** () and **mlx_expose_hook** () work exactly the same way. *funct_ptr* is a pointer to the function you want to be called when an event occurs. This assignment is specific to the window defined by the *win_ptr* identifier. The *param* adress will be passed to the function everytime it is called, and should be used to store the parameters it might need.

The syntax for the **mlx_loop_hook** () function is identical to the previous ones, but the given function will be called when no event occurs.

When it catches an event, the MiniLibX calls the corresponding function with fixed parameters:

```
expose_hook(void *param);
key_hook(int keycode,void *param);
mouse_hook(int button,int x,int y,void *param);
loop_hook(void *param);
```

These function names are arbitrary. They here are used to distinguish parameters according to the event.

These functions are NOT part of the MiniLibX.

param is the address specified in the mlx_*_hook calls. This address is never used nor modified by the MiniLibX. On key and mouse events, additional information is passed: *keycode* tells you which key is pressed (X11 : look for the include file "keysymdef.h", MacOS : create a small software and find out by yourself), (*x* , *y*) are the coordinates of the mouse click in the window, and *button* tells you which mouse button was pressed.

GOING FURTHER WITH EVENTS

The MiniLibX provides a much generic access to all type of events. The *mlx.h* include define **mlx_hook()** in the same manner mlx_*_hook functions work. The event and mask values will be taken from the X11 include file "X.h" (even for MacOSX, for compatibility purposes)

See source code of mlx_int_param_event.c to find out how the MiniLibX will call your own function for a specific event.

SEE ALSO

mlx(3), mlx_new_window(3), mlx_pixel_put(3), mlx_new_image(3)

AUTHOR