42 Docs

Loops

TABLE OF CONTENTS

Introduction

Now that you finally understand the basics of the MiniLibX library, we will start with drawing a tiny animation in the window. For this we will be using two new functions, namely `mlx_loop` and `mlx_loop_hook`.

Loops are a feature of MiniLibX where it will continue to call your hook registered in `mlx_loop_hook` to render new frames, which you obviously have to pass to the window yourself.

Hooking into loops

To initiate a loop, we call the `mlx_loop` function with the `mlx` instance as only parameter, take a look:

```
#include <mlx.h>

int	main(void)
{
	void *mlx;

	mlx = mlx_init();
	mlx_loop(mlx);
}
```

This will do nothing of course as we have no loop hook registered, therefore we will not be able to write anything to our frame.

To do this, you will have to create a new window and use the mutations that we described in the Getting Started chapter. We assume that your knowledge of that is proficient and that you will be able to pass your parameters accordingly. An example whiteboarded version of it could look as follows:

```
#include <mlx.h>

int   render_next_frame(void *YourStruct);

int   main(void)
{
      void *mlx;

      mlx = mlx_init();
      mlx_loop_hook(mlx, render_next_frame, YourStruct);
      mlx_loop(mlx);
}
```

Now for each frame it requires, it will call the function `render_next_frame` with the parameter `YourStruct`. This will persist through multiple calls if it is a pointer, so use that to your advantage!

Test your skills!

Now that you understand how to register your own rendering function, we suggest that you create the following renderers:

- Render a moving rainbow that shifts through all colors (screen turns red, becomes green and then becomes blue, then starts all over again).
- Create a circle that you can move accross your screen using your WASD keys.