# Creating your own Nuklear based project

Huge thanks go to the *ainvar* and *iarwain* for their work bringing the Nuklear User Interface Library to Orx.

You can easily generate a Nuklear project using `init` from the github version of Orx.

This article assumes that you have downloaded the latest version of Orx from github and have built Orx.

## How to Initialise a Nuklear project

There are two ways to create a Nuklear project. Firstly, using Interactive Mode:

```
init
```

Then follow all the prompts and choose `yes` when asked if you want to include Nuklear:

```
[Extension] nuklear: Nuklear support
(https://github.com/immediate-mode-ui/nuklear)? (no)
```

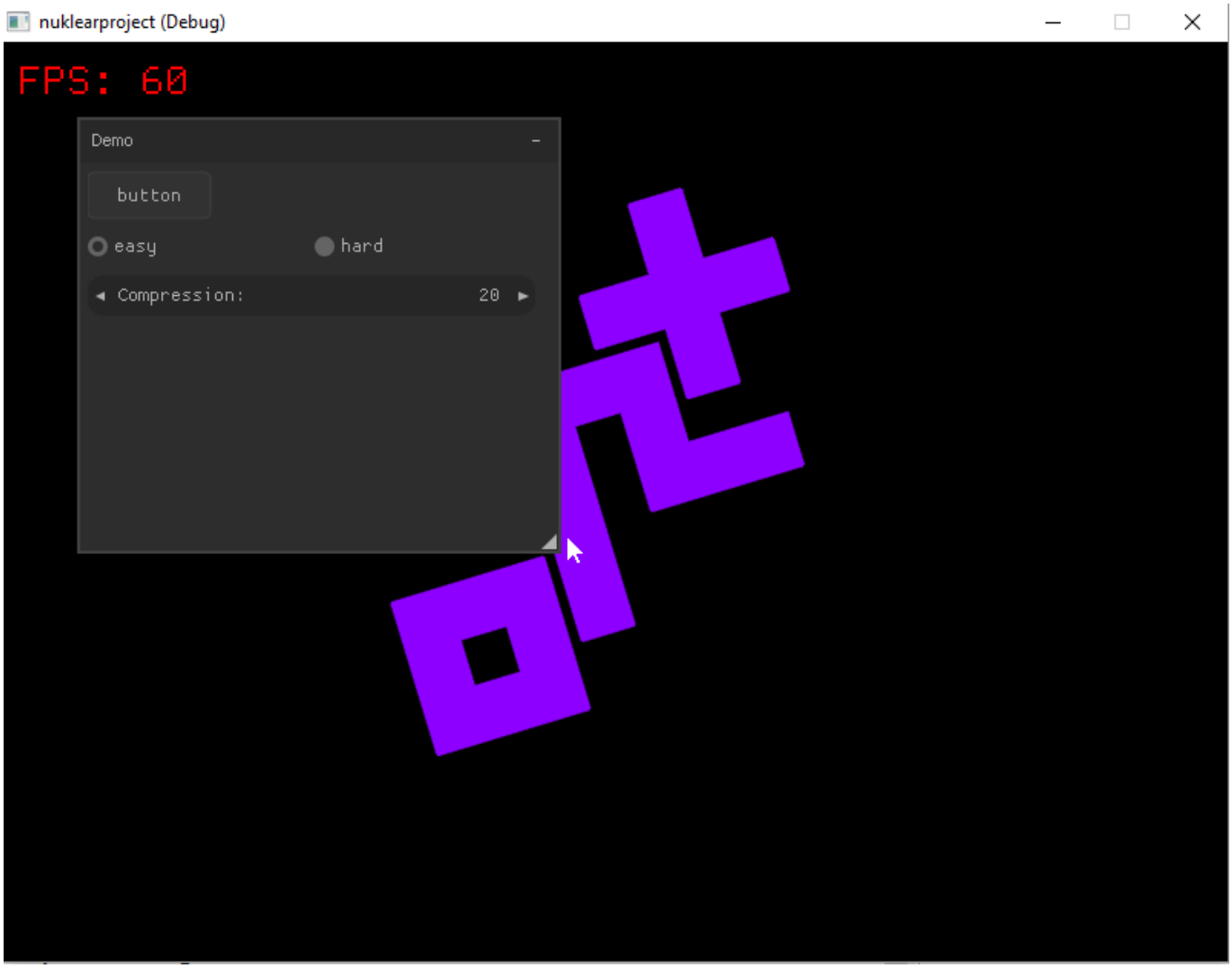
The second way is to specify everything on the commandline:

```
init myGameFolder/MyGame +nuklear
```

## Working with your project

Load your new Nuklear-based project using your favourite IDE. You'll find your build in the `build` folder.

Compile and run. You should get a nice screen with demo gui controls.




And just for some extra flair, add a little up/down movement to the logo in your main config file:

```
[Object]
Graphic         = @
Texture         = logo.png
Pivot           = center
AngularVelocity = 18
FXList          = FadeIn # ColorCycle # MoveAround

[MoveAround]
SlotList        = @
Type            = position
StartTime       = 0.0
EndTime         = 4
Curve           = sine
StartValue      = (0,0)
EndValue        = (0, -300)
Loop            = true
```

## Example Nuklear code

In the Run() function contains some demo code, for example:

```
if(nk_begin(&sstNuklear.stContext, "Demo", nk_rect(50, 50, 200, 200),
NK_WINDOW_BORDER | NK_WINDOW_MOVABLE | NK_WINDOW_SCALABLE |
NK_WINDOW_MINIMIZABLE | NK_WINDOW_TITLE))
    {
        enum {EASY, HARD};
        static orxS32 Op = EASY;
        static orxS32 Property = 20;

        nk_layout_row_static(&sstNuklear.stContext, 30, 80, 1);
        if (nk_button_label(&sstNuklear.stContext, "button"))
        {
            orxLOG("Nuklear button pressed.");
        }
        nk_layout_row_dynamic(&sstNuklear.stContext, 30, 2);
        if(nk_option_label(&sstNuklear.stContext, "easy", Op == EASY))
        {
            Op = EASY;
        }
        if(nk_option_label(&sstNuklear.stContext, "hard", Op == HARD))
        {
            Op = HARD;
        }
        nk_layout_row_dynamic(&sstNuklear.stContext, 25, 1);
        nk_property_int(&sstNuklear.stContext, "Compression:", 0, &Property,
100, 10, 1);
    }
    nk_end(&sstNuklear.stContext);
```

## orx/Scroll based projects

You can also create an orx/Scroll-based Nuklear project. See: Creating your own orx/Scroll project using 'init'

## Learning Nuklear

To get started learning how to use the UI Library, check the official repo page: https://github.com/Immediate-Mode-UI/Nuklear

The documentation is available here: https://immediate-mode-ui.github.io/Nuklear/doc/nuklear.html

You can also find information in this previous guide for Orx and Nuklear here: https://www.danjodev.com/2020/01/nuklear-use-with-orx-engine.html (this was created before the

final integration with Orx).

Some tutorials that could be helpful:

- https://www.thecodingfox.com/nuklear-usage-guide-lwjgl
- https://dexp.in/articles/nuklear-intro/