NUS Library Documentation

Version 1

Developed and Written by Kim Pampusch

Table Of Contents

1 Introduction

NUS library is video game development library written in c.

2 Minimum Specs

Requires os: Linux or Windows
Requires video driver supporting Vulkan 1.0.0 or later

3 Development

3.0 Development Environments

3.0.0 Linux

3.0.0.0 Debian

Copy the following into a terminal

```
cd ~
git clone https://github.com/PyCee/NUS_library.git
cd NUS_library/
make install
make recompile
```

3.0.0.1 Other

idk

3.0.1 Windows

Idk

3.1 Unit Tests

Once the environment is setup, the developer may run the unit tests located in the directories in `NUS_library/unit_tests/` to test library functionality.

4 Error Checking

A function that supports error checking will return a NUS_result

```
typedef enum NUS_result{
  NUS_FAILURE = 0,
  NUS_SUCCESS = 1
} NUS_result;
```

4.0 Results

4.0.0 Failure

NUS_FAILURE means the called function failed in such a way that the program must terminate.

4.0.1 Success

NUS_SUCCESS represents a complete success with no cause for worry

5 Window Management

6 User Input

User Input is handled by a series of callbacks. A single NUS_system_events should be created by the application and populated with application defined callbacks.
Types of user input can be categorized into one of the following:

```
typedef enum NUS_event_type{
  NUS_EVENT_MIN_VALUE = 1,
  NUS_EVENT_CLOSE_WINDOW = 2,
  NUS_EVENT_KEY_PRESS = 3,
  NUS_EVENT_KEY_RELEASE = 4,
  NUS_EVENT_MOUSE_BUTTON_PRESS = 5,
  NUS_EVENT_MOUSE_BUTTON_RELEASE = 6,
  NUS_EVENT_MOUSE_MOTION = 7,
  NUS_EVENT_MOUSE_SCROLL = 8,
  NUS_EVENT_MAX_VALUE = 9,
} NUS_event_type;
```

Each event type has a several subtypes which specify an exact event to respond to
Ex: a specific key to a NUS_EVENT_KEY_PRESS

`NUS_result nus_event_handler_build(NUS_event_handler *p_event_handler)`

PARAMETERS

p_event_handler - pointer to an uninitialized, allocated NUS_event_handler

DESCRIPTION
Initializes p_event_handler

`void nus_event_handler_free(NUS_event_handler *p_event_handler)`
PARAMETERS
p_event_handler - pointer to an initialized NUS_event_handler
DESCRIPTION
Frees allocated memory of p_event_handler

`void nus_event_handler_set(NUS_event_handler *p_event_handler)`
PARAMETERS
p_event_handler - pointer to an initialized NUS_event_handler
DESCRIPTION
Tells the library what NUS_event_handler is responsible for the various callbacks

---

`void nus_system_events_handle(NUS_window window)`
PARAMETERS
window - events from this window will be received and handled
DESCRIPTION
calls callbacks for events that have not been handled

`#define nus_event_handler_function_append(event_handler, event_type, group_index, function)`
PARAMETERS
event-handler - an initialized event handler that will receive the callback
event_type - NUS_event_type that specifies what type of event the callback will respond to
group_index - an event subtype that specifies what specific event the callback will respond to
function - callback function
SPECIFICS
if event_type is NUS_EVENT_MOUSE_MOTION, function should be of type
`void (*function)(float rel_x, float rel_y)`
Where `rel_x` and `rel_y` are the change in the mouse coordinates
otherwise, function should be of type

```
        void (*function)(void)
    DESCRIPTION
        sets up a user input based callback
```

6.0 Close Window

Represented by NUS_event_type NUS_EVENT_CLOSE_WINDOW
Callbacks of this type are called when the user clicks on the close window button of the gui, typically in the top left or right of the window, characterized by an ‘x’.
The only valid subtype is the value 0, as no real subtype exists.

5.0.0 Example

```
nus_event_handler_append(event_handler, NUS_EVENT_CLOSE_WINDOW, 0,
                                        close_window_callback);
```

6.1 Keyboard

6.1.0 Key Codes

Key code subtypes refer to which key the callback is bound to, which are:

Key Code	**Physical Representation**	**Key Name**
NUS_KEY_ESC	ESC	Escape
NUS_KEY_1	1	One
NUS_KEY_2	2	Two
NUS_KEY_3	3	Three
NUS_KEY_4	4	Four
NUS_KEY_5	5	Five
NUS_KEY_6	6	Six
NUS_KEY_7	7	Seven
NUS_KEY_8	8	Eight

NUS_KEY_9	9	Nine
NUS_KEY_0	0	Zero
NUS_KEY_MINUS	-	Minus
NUS_KEY_EQUALS	=	Equals
NUS_KEY_BACKSPACE	BACKSPACE	Backspace
NUS_KEY_TAB	TAB	Tab
NUS_KEY_Q	q	Q
NUS_KEY_W	w	W
NUS_KEY_E	e	E
NUS_KEY_R	r	R
NUS_KEY_T	t	T
NUS_KEY_Y	y	Y
NUS_KEY_U	u	U
NUS_KEY_I	i	I
NUS_KEY_O	o	O
NUS_KEY_P	p	P
NUS_KEY_LBRACKET	[	Left Bracket
NUS_KEY_RBRACKET	]	Right Bracket
NUS_KEY_ENTER	ENTER	Enter
NUS_KEY_LCTRL	CTRL	Left Control
NUS_KEY_A	a	A
NUS_KEY_S	s	S
NUS_KEY_D	d	D
NUS_KEY_F	f	F
NUS_KEY_G	g	G

NUS_KEY_H	h	H
NUS_KEY_J	j	J
NUS_KEY_K	k	K
NUS_KEY_L	l	L
NUS_KEY_SEMICOLON	;	Semi-colon
NUS_KEY_APOSTROPHE	‘	Apostrophe
NUS_KEY_LSHIFT	SHIFT	Left Shift
NUS_KEY_BACKSLASH	\	Backslash
NUS_KEY_Z	z	Z
NUS_KEY_X	x	X
NUS_KEY_C	c	C
NUS_KEY_V	v	V
NUS_KEY_B	b	B
NUS_KEY_N	n	N
NUS_KEY_M	m	M
NUS_KEY_COMMA	,	Comma
NUS_KEY_PERIOD	.	Period
NUS_KEY_RSHIFT	SHIFT	Right Shift
NUS_KEY_KP_MULTIPLY	*	Star
NUS_KEY_LALT	ALT	Left Alt
NUS_KEY_SPACE		Spacebar
NUS_KEY_NUM_LOCK	NUM LOCK	Num Lock
NUS_KEY_KP_7	7	Keypad Seven
NUS_KEY_KP_8	8	Keypad Eight
NUS_KEY_KP_9	9	Keypad Nine

NUS_KEY_KP_MINUS	-	Keypad Minus
NUS_KEY_KP_4	4	Keypad Four
NUS_KEY_KP_5	5	Keypad Five
NUS_KEY_KP_6	6	Keypad Six
NUS_KEY_KP_PLUS	+	Keypad Plus
NUS_KEY_KP_1	1	Keypad One
NUS_KEY_KP_2	2	Keypad Two
NUS_KEY_KP_3	3	Keypad Three
NUS_KEY_KP_0	0	Keypad Zero
NUS_KEY_KP_PERIOD	.	Keypad Period
NUS_KEY_KP_ENTER	ENTER	Keypad Enter
NUS_KEY_RCTRL	CTRL	Right Control
NUS_KEY_RALT	ALT	Right Alt
NUS_KEY_ARROW_UP	↑	Up Arrow
NUS_KEY_ARROW_LEFT	←	Left Arrow
NUS_KEY_ARROW_RIGHT	→	Right Arrow
NUS_KEY_ARROW_DOWN	↓	Down Arrow

6.1.1 Example

```
nus_event_handler_append(event_handler, NUS_EVENT_KEY_PRESS,
                                        NUS_KEY_W, w_press_callback);
nus_event_handler_append(event_handler, NUS_EVENT_KEY_RELEASE,
                                        NUS_KEY_R, r_release_callback);
```

6.2 Mouse

6.2.0 Motion

Mouse motion requires a 0 in place of a subtype. No real subtype exists.

6.2.1 Button

A mouse button has the subtypes:
`NUS_MOUSE_BUTTON_LEFT, NUS_MOUSE_BUTTON_RIGHT, and NUS_MOUSE_BUTTON_MIDDLE`

6.2.2 Scroll

A scroll event has the subtypes:
`NUS_SCROLL_UP, NUS_SCROLL_DOWN, NUS_SCROLL_LEFT, and NUS_SCROLL_RIGHT`

6.2.3 Example

```
nus_event_handler_append(event_handler, NUS_EVENT_MOUSE_MOTION, 0,
                         motion_callback);
nus_event_handler_append(event_handler, NUS_EVENT_MOUSE_SCROLL,
                         NUS_SCROLL_UP, scroll_up_callback);
nus_event_handler_append(event_handler,
                         NUS_EVENT_MOUSE_BUTTON_PRESS,
                         NUS_MOUSE_BUTTON_RIGHT,
                         right_button_press_callback);
```

7 Vulkan Instance

8 GPU Information

8.0 Queue Info

9 Presentation Surface

9.0 Present