NGN_INPUT.H

Lectura del teclado

Listado de teclas disponibles:

```
key_1 key_2 key_3 key_4 key_5 key_6 key_7 key_8 key_9 key_0
key_Q key_W key_E key_R key_T key_Y key_U key_I key_O key_P
key_A key_S key_D key_F key_G key_H key_J key_K key_L
key_Z key_X key_C key_V key_B key_N key_M
key_SPACE key_ESC key_RETURN key_TAB key_BACK_SPACE
key_ARROW_UP key_ARROW_DOWN key_ARROW_LEFT key_ARROW_RIGHT
key_LEFT_CONTROL key_RIGHT_CONTROL
key_LEFT_SHIFT key_RIGHT_SHIFT
key_LEFT_ALT key_RIGHT_ALT
key_F1 key_F2 key_F3 key_F4 key_F5 key_F6
key_F7 key_F8 key_F9 key_F10 key_F11 key_F12
key_INSERT key_DELETE key_HOME key_END key_PAGE_UP key_PAGE_DOWN
key_GRAVE key_MINUS key_EQUAL
key_LEFT_BRACKET key_RIGHT_BRACKET
key_SEMICLON key_APOSTROPHE key_BACK_SLASH
key_COMMA key_PERIOD key_SLASH
key_PRINT_SCREEN key_SCROLL_LOCK key_PAUSE
nkp_SLASH nkp_ASTERISK nkp_MINUS
nkp_7 nkp_8 nkp_9
nkp_4 nkp_5 nkp_6
nkp_1 nkp_2 nkp_3
nkp_0 nkp_PERIOD nkp_RETURN nkp_PLUS
NGN_Key key_ANY_KEY
```

Listado de propiedades disponibles:

```
bool down           // Nueva pulsación
bool held           // Se mantiene apretada
bool up             // Se ha soltado
```

Ejemplo de uso:

```
if (ngn->input->key_ESC->down) …
```

Lectura del estado del ratón

Propiedades disponibles:

```
int32_t x                       // Posición X del ratón en la escena
int32_t y                       // Posición Y del ratón en la escena
bool LB [.down, .held, .up]     // Estado del botón izquierdo
bool MB [.down, .held, .up]     // Estado del botón central
bool RB [.down, .held, .up]     // Estado del botón derecho
int32_t wheel_x                 // Scroll de la rueda en X [-1, 0, 1]
int32_t wheel_y                 // Scroll de la rueda en Y [-1, 0, 1]
int32_t raw_x                   // Valor relativo del desplazamiento en X
int32_t raw_y                   // Valor relativo del desplazamiento en Y
```

Ejemplo de uso:

```
if (ngn->input->mouse.RB.held) {
    my_sprite.x = ngn->input->mouse.x;
    my_sprite.y = ngn->input->mouse.y;
}
```

Lectura del estado del controlador (pad o joystick)

Propiedades disponibles (comunes):

```
int32_t controllers       // Indica el número de controladores conectados
```

Por defecto, se pueden gestionar hasta un máximo de 8 controladores diferentes, con 8 axis, 20 botones y un POV (cruceta digital) cada uno. La asignación de los axis o botones puede variar en función del controlador conectado. En caso de conexión o desconexión de un controlador en caliente, la librería reiniciara la lista de controladores disponibles.

Propiedades disponibles (de cada controlador):

La clase “input” dispone de acceso a los controladores a partir de la propiedad “controler[n]”, la cual a su vez está compuesta de:

```
bool available                          // El controlador está disponible?
int32_t id                              // ID de la instancia del controlador
int32_t axis_number                     // Numero de axis disponibles
float axis[n]                           // Axis [-1 a 1]
int32_t button_number                   // Numero de botones disponibles
bool button[n] [.down, .held, .up]      // Botones
bool pov_available                      // POV (Cruceta digital) disponible?
uint8_t pov                             // Valor del POV [bitmask de 4 bits]
bool pov_up [.down, .held, .up]         // POV como teclas cursor (Arriba)
bool pov_down [.down, .held, .up]       // POV como teclas cursor (Abajo)
bool pov_left [.down, .held, .up]       // POV como teclas cursor (Izquierda)
bool pov_right[.down, .held, .up]       // POV como teclas cursor (Derecha)
```

Ejemplos de uso:

```
if (ngn->input->controller[0].button[3].held) …
if (ngn->input->controller[0].pov_up.down)…
position.x += (ngn->input->controller[0].axis[1] * speed);
```

Efecto “rumble”

```
int32_t ControllerRumble(
    uint32_t controller_id,     // ID del controlador (0 – 7)
    float intensity,            // Intensidad (0.0f – 1.0f)
    uint32_t duration           // Duración en ms (> 0)
);
```

```
ngn->input->ControllerRumble(0, 0.75f, 500);
```

Haz vibrar (si la función está disponible) el controlador nº0 a un 75% de intensidad durante un periodo de 500 ms.

Constantes definidas para la identificación de X-INPUT (Controladores de la familia Xbox 360 y superiores)

```
static const uint32_t XBOX_BUTTON_A = 0;
static const uint32_t XBOX_BUTTON_B = 1;
static const uint32_t XBOX_BUTTON_X = 2;
static const uint32_t XBOX_BUTTON_Y = 3;
static const uint32_t XBOX_BUTTON_L = 4;
static const uint32_t XBOX_BUTTON_R = 5;
static const uint32_t XBOX_BUTTON_BACK = 6;
static const uint32_t XBOX_BUTTON_START = 7;
static const uint32_t XBOX_BUTTON_STICK_L = 8;
static const uint32_t XBOX_BUTTON_STICK_R = 9;
static const uint32_t XBOX_BUTTON_XBOX = 10;
static const uint32_t XBOX_STICK_L_AXIS_X = 0;
static const uint32_t XBOX_STICK_L_AXIS_Y = 1;
static const uint32_t XBOX_STICK_R_AXIS_X = 2;
static const uint32_t XBOX_STICK_R_AXIS_Y = 3;
static const uint32_t XBOX_TRIGGER_AXIS = 4;
static const float XBOX_AXIS_DEADZONE = 0.25f;
```