

# NGN\_TEXT\_LAYER.H

## *(Funciones de La capa de texto)*

```
NGN_TextLayer(  
    NGN_TextFont* default_font,  
        // Fuente por defecto  
    NGN_TextureData* bg = NULL,  
        // Textura de fondo  
    int32_t position_x = 0,  
        // Posicion X (0 por defecto)  
    int32_t position_y = 0,  
        // Posicion Y (0 por defecto)  
    uint32_t _width = DEFAULT_VALUE,  
        // Ancho de la capa (Toda la pantalla por defecto)  
    uint32_t _height = DEFAULT_VALUE,  
        // Alto de la capa (Toda la pantalla por defecto)  
    bool _filtering = false  
        // Filtrado del contenido?  
);
```

Crea una nueva capa de texto, usando la fuente y parámetros especificados.

```
NGN_TextLayer* textbox = new NGN_TextLayer(my_font, NULL, 0, 0, 200, 64);
```

```
void Position(float position_x, float position_y);  
void Position(Vector2 pos);
```

Posiciona la capa de texto en la coordenada dada.

```
textbox->Position(1200, 900);
```

```
void Translate(float speed_x, float speed_y);  
void Translate(Vector2 spd);
```

Mueve la capa de texto en la dirección y velocidades dadas.

```
textbox->Translate(5.0f, 0.0f);
```

```
void Size(float w, float h);
```

Cambia el tamaño de la capa de texto.

```
textbox->Size(64, 48);
```

```
void Scale(float w, float h);  
void Scale(float scale);
```

Escala la capa de texto, según el factor dado. Según la sobrecarga usada, escalara los ejes en conjunto o por separado. La escala por defecto es 1.0f.

```
textbox->Scale(1.5f);  
textbox->Scale(2.0f, 0.75f);
```

```
void Rotate(double degrees);
```

Rota la capa de texto cada frame el número de unidades dado, en grados.

```
textbox->Rotate(1.2f);
```

```
void SetCenter(float x, float y);
```

Especifica, en coordenadas relativas y desde el centro real de la capa de texto, donde se ubicará el centro de rotación de la capa de texto.

```
textbox->SetCenter(-10, -5);
```

***Vector2 position***

Posición de la capa de texto (global o en pantalla).

***float width***  
***float height***

Tamaño de la capa de texto.

***bool visible***

Indica si la capa de texto es o no visible.

***int32\_t alpha***

Nivel de transparencia de la capa de texto, entre 0 y 255.

***SDL\_BlendMode blend\_mode***

Modo de mezcla de color de la capa de texto. Los modos disponibles son: *NGN\_BLENDMODE\_NONE*, *NGN\_BLENDMODE\_ALPHA*, *NGN\_BLENDMODE\_ADDITIVE* y *NGN\_BLENDMODE\_MODULATE*. El valor por defecto de esta propiedad es *NGN\_BLENDMODE\_ALPHA*.

### ***bool filtering***

Activa o desactiva el filtrado bilineal del contenido de la capa de texto.

### ***double rotation***

Rotación de la capa de texto, en grados.

### ***bool flip\_h***

### ***bool flip\_v***

Volteado vertical y horizontal de la capa de texto.

Nota:

Los cambios de tamaño o escala no afectan al tamaño original del contenedor, solo se cambia el tamaño del contenido al representarse en la pantalla.

## ***(Funciones de escritura)***

### **void Cls();**

Borra el contenido de la capa de texto y restaura la posición del cabezal de escritura a la esquina superior-izquierda.

```
textbox->Cls();
```

### **void Locate(int32\_t x, int32\_t y);**

Posiciona el cabezal de escritura en las coordenadas de la capa especificadas.

```
textbox->Locate(100, 50);
```

### **void Padding(uint32\_t pd);**

Define el margen interior que tendrá la capa de texto a partir de ese momento.

```
textbox->Padding(16);
```

### **void Font(NGN\_TextFont\* fnt);**

Selecciona que fuente se usara en la escritura del texto a partir de ese momento.

```
textbox->Font(my_font);
```

```
void InkColor(uint8_t r, uint8_t g, uint8_t b); // [R, G, B]
void InkColor(uint32_t rgb);                  // [0xRRGGBB]
```

Selecciona que color se usará para el texto a partir de ese momento.

```
textbox->InkColor(255, 200, 40);
textbox->InkColor(0xFFAA33);
```

```
void CanvasColor(uint8_t r, uint8_t g, uint8_t b, uint8_t a);
// [R, G, B, A]
void CanvasColor(uint32_t rgba);
// [0xRRGGBBAA];
```

Selecciona que color se usará para el fondo de la capa de texto a partir del próximo borrado CIs(), si no existe una textura de fondo.

```
textbox->CanvasColor(0, 0, 0, 128);
textbox->CanvasColor(0xFF0000FF);
```

```
void Print(std::string text);
```

Escribe el texto dado a partir de la posición actual del cabezal de escritura, usando el color definido. El comando \n será reconocido.

```
textbox->Print("Hello World!");
```

### ***Vector2I32 Locate***

Posición actual del cabezal de escritura en la capa de texto.

```
struct {
    uint8_t r;
    uint8_t g;
    uint8_t b;
} ink;
```

Color actual para el texto.

```
struct {
    uint8_t r;
    uint8_t g;
    uint8_t b;
    uint8_t a;
} canvas;
```

Color actual del fondo de la capa de texto.

***int32\_t padding***

Margen interior actual de la capa de texto.

***bool word\_wrap***

Salto automático de línea (habilitado por defecto).

***bool auto\_home***

Reinicio automático de posición al llegar al final de la capa de texto (deshabilitado por defecto).