Skip to content

Font Drawing (blf)

This module provides access to Blender's text drawing functions.

Hello World Text Example

Example of using the blf module. For this module to work we need to use the GPU module `gpu` as well.

```
# import stand alone modules
import blf
import bpy

font_info = {
    "font_id": 0,
    "handler": None,
}


def init():
    """init function - runs once"""
    import os
    # Create a new font object, use external TTF file.
    font_path = bpy.path.abspath('//Zeyada.ttf')
    # Store the font indice - to use later.
    if os.path.exists(font_path):
        font_info["font_id"] = blf.load(font_path)
    else:
        # Default font.
        font_info["font_id"] = 0

    # set the font drawing routine to run every frame
    font_info["handler"] = bpy.types.SpaceView3D.draw_handler_add(
        draw_callback_px, (None, None), 'WINDOW', 'POST_PIXEL')


def draw_callback_px(self, context):
    """Draw on the viewports"""
    # BLF drawing routine
    font_id = font_info["font_id"]
    blf.position(font_id, 2, 80, 0)
    blf.size(font_id, 50.0)
    blf.draw(font_id, "Hello World")


if __name__ == '__main__':
    init()
```

blf.**aspect(fontid, aspect)**

Set the aspect for drawing text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **aspect** (*float*) – The aspect ratio for text drawing to use.

blf.**clipping(fontid, xmin, ymin, xmax, ymax)**

Set the clipping, enable/disable using CLIPPING.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **xmin** (*float*) – Clip the drawing area by these bounds.
- **ymin** (*float*) – Clip the drawing area by these bounds.
- **xmax** (*float*) – Clip the drawing area by these bounds.
- **ymax** (*float*) – Clip the drawing area by these bounds.

blf.**color(fontid, r, g, b, a)**

Set the color for drawing text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **r** (*float*) – red channel 0.0 - 1.0.
- **g** (*float*) – green channel 0.0 - 1.0.
- **b** (*float*) – blue channel 0.0 - 1.0.
- **a** (*float*) – alpha channel 0.0 - 1.0.

blf.**dimensions(fontid, text)**

Return the width and height of the text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **text** (*str*) – the text to draw.

RETURNS:

the width and height of the text.

RETURN TYPE:

tuple[float, float]

blf.**disable(fontid, option)**

Disable option.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **option** (*int*) – One of ROTATION, CLIPPING, SHADOW or KERNING_DEFAULT.

blf.**draw(fontid, text)**

Draw text in the current context.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **text** (*str*) – the text to draw.

blf.**enable(fontid, option)**

Enable option.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **option** (*int*) – One of ROTATION, CLIPPING, SHADOW or KERNING_DEFAULT.

blf.**load(filepath)**

Load a new font.

PARAMETERS:

filepath (*str* | *bytes*) – the filepath of the font.

RETURNS:

the new font's fontid or -1 if there was an error.

RETURN TYPE:

int

blf.**position(fontid, x, y, z)**

Set the position for drawing text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **x** (*float*) – X axis position to draw the text.
- **y** (*float*) – Y axis position to draw the text.
- **z** (*float*) – Z axis position to draw the text.

blf.**rotation(fontid, angle)**

Set the text rotation angle, enable/disable using ROTATION.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **angle** (*float*) – The angle for text drawing to use.

blf.**shadow(fontid, level, r, g, b, a)**

Shadow options, enable/disable using SHADOW .

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **level** (*int*) – The blur level (0, 3, 5) or outline (6).
- **r** (*float*) – Shadow color (red channel 0.0 - 1.0).
- **g** (*float*) – Shadow color (green channel 0.0 - 1.0).
- **b** (*float*) – Shadow color (blue channel 0.0 - 1.0).
- **a** (*float*) – Shadow color (alpha channel 0.0 - 1.0).

blf.**shadow_offset(fontid, x, y)**

Set the offset for shadow text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **x** (*float*) – Vertical shadow offset value in pixels.
- **y** (*float*) – Horizontal shadow offset value in pixels.

blf.**size(fontid, size)**

Set the size for drawing text.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **size** (*float*) – Point size of the font.

blf.**unload(filepath)**

Unload an existing font.

PARAMETERS:

filepath (*str* | *bytes*) – the filepath of the font.

blf.**word_wrap(fontid, wrap_width)**

Set the wrap width, enable/disable using WORD_WRAP.

PARAMETERS:

- **fontid** (*int*) – The id of the typeface as returned by `blf.load()` , for default font use 0.
- **wrap_width** (*int*) – The width (in pixels) to wrap words at.

blf.**CLIPPING**

Constant value 2

blf.**MONOCHROME**

Constant value 128

blf.**ROTATION**

Constant value 1

blf.**SHADOW**

Constant value 4

blf.**WORD_WRAP**

Constant value 64