TwistedBrush Pro Studio Lua Filters

TwistedBrush Pro Studio Lua Filters

1 Overview

2 Reference

Overview

Overview of Lua Filter Scripts

ALua script filter in TwistedBrush is a small piece code written in the Lua programming language (versin 5.1) that can be used to quickly implement image processing filters right within TwistedBrush.

The Lua scripting filter is designed to be somewhat compatible with the gluas plug-ins used in The Gimp image editing software package. The gluas plugin system was developed by Øyvind Kolås. There is a nice tutorial of gluas titled Image Processing with gluas a large part is applicable to TwistedBrush and most (if not all) of the sample scripts shown should work within TwistedBrush.

Additional information on gluas can be found at http://www.gluas.org.

An excellent online book covering all the details of the Lua programming language can be found here.

In case you haven't determined this yet, this topic and programming of filters with Lua is designed for those with computer programming experience. However, this doesn't limited anyone from using filters that are either included with TwistedBrush or shared from developers.

Example Lua Filter

Here is an example of a Lua Script Filter that does a simple invert (negative).

```
for y = 0, height - 1 do
  for x = 0, width - 1 do
    local r, g, b
    r, g, b = get_rgb(x, y)
    r = 1 - r
    g = 1 - g
    b = 1 - b
    set_rgb(x, y, r, g, B)
  end
end
```

Reference

```
v = get_value(x, y)

set_value(x, y, v)

r, g, b = get_rgb(x, y)

set_rgb(x, y, r, g, B)

r, g, b, a = get_rgba(x, y)

set_rgba(x, y)

h, s, v = get_hsv(x, y)

set_hsv(x, y, h, s, v)

h, s, l = get_hsl(x, y)

set_hsl(x, y, h, s, l)

l, a, b = get_lab(x, y)

set_lab(x, y)

c, m, y, k = get_cmyk(x, y)

set_cmyk(x, y, c, m, y, k)

flush()
```

progress(v) // only a stub, progress is not shown in TwistedBrush

```
width

height

bound_x0 -- Always 0 in TwistedBrush

bound_y0 -- Always 0 in TwistedBrush

bound_x1 -- Always width - 1 in TwistedBrush

bound_y1 -- Always height - 1 in TwistedBrush
```

```
a = get_alpha(x, y)

set_alpha(x, y, a)

m = get_mask(x, y)    -- added in 16.02

set_mask(x, y, m)    -- added in 16.02

r, g, b = get_color()  -- Gets the current brush color

r, g, b = get_color1()   -- Gets the current color from slot 1

r, g, b = get_color2()  -- Gets the current color from slot 2

r, g, b = get_color3() -- Gets the current color from slot 3

r, g, b = get_color4() -- Gets the current color from slot 4

single_buffer() -- All functions work from the same buffer

message_box(message, title) -- Display message box

merge(merge_mode, opacity) -- Like the flush() method but a merge
mode and opacity can be specified for the
merging of the destination buffer back into the source buffer.
The valid merge types are listed here.

output_debug_string(message) -- Send the string to the Windows
debug monitor.

is_escape() -- Returns 1 if the escape key is currently pressed.
```

TwistedBrush Lua Filter Bitwise APIs

```
r = bit_not(v1) -- Bitwise Not (added in version 15.53)

r = bit_and(v1, v2) -- Bitwise And (added in version 15.53)

r = bit_or(v1, v2) - Bitwise Or (added in version 15.53)

r = bit_xor(v1, v2) -- Bitwise Xor (added in version 15.53)

r = bit_shfr(v1, bits) -- Bitwise shift right (added in version
15.53)

r = bit_shfl(v1, bits) -- Bitwise shift left (added in version
15.53)
```

```
paint_open(seed) -- Seed is a random seed

paint_close() -- End the paint session

paint_stroke_start(x, y) - Start a new stroke.

paint_stroke_stop() -- End the stroke

paint_stroke_pos(x, y, pressure) -- New stroke position

paint_select_brush(artsetName, brushName, mergeEffects)

paint_brush_size(size)

paint_brush_density(density)

paint_brush_opacity(opacity)

paint_brush_color(r, g, B)

paint_set_color_banks(selectedBank, r1, g1, b1, r2, g2, b2, r3,
g3, b3, r4, g4, b4)

paint_line(x1, y1, x2, y2)

paint_spline(x1, y1, x2, y2, cx1, cy1, cx2, cy2)

paint_dab(x, y, pressure)

-- The parameters for the paint_filter() function directly
correspond to the controls for the filter in question.
paint_filter(filtername, variant, slider1, slider2, slider3,
slider4, slider5, slider6, style1, style2, flag1, flag2, flag3,
flag4,

flag5, flag6, r, g, b, ustring)

-- These paint functions were available starting in 17.28
paint_buffer_from_layer(bufNum)  -- Copy contents of the current
layer into the bufNum
```

```
paint_buffer_to_layer(bufNum)    -- Copy contents of the bufNum
into the current layer

paint_buffer_copy(srcBufNum, dstBufNum)   -- Copy contents of
SrcBufNum to DstBufNum

paint_buffer_clear(bufNum)        -- Clear content of the bufNum

paint_buffer_merge_layer(bufNum, mixMode, opacity)  -- Merge
conents of the bufNum into the current layer, using the
mix mode (merge type) and opacity

paint_buffer_from_mask(bufNum)    -- Copy contents of the current
mask into the bufNum

paint_buffer_to_mask(bufNum)     -- Copy contents fo the bufNum
into the current mask

paint_flip_horizontal()          -- Flip horizontal

paint_flip_vertical()            -- Flip vertical

paint_clear_page()               -- Clear the current layer

paint_fill_page()                -- Fill the current layer with
the current color.

paint_set_page(r, g, b, a)       -- Set current layer to the r,
g, b, a values

paint_mask_on(flag)              -- Turn mask on or off (values 1
or 0)

paint_mask_clear()               -- Clear the mask

paint_mask_invert()              -- Invert the mask

paint_create_from_mask()         -- Set current layer values from
current mask

paint_alpha_create_from_mask()      -- Set alpha values of
current layer from the current mask
```

```
paint_mask_create_from_image()      -- Create mask from current
layer data

paint_mask_create_from_image_luma() -- Create mask from current
layer data using luma values

paint_mask_create_from_alpha()      -- Create mask from current
layer alpha data

paint_flood_fill(r, g, b, x, y, tolerance, fillType)   -- Flood
fill tool

paint_rectangle(r, g, b, x1, y1, x2, y2, opacity)
-- Rectangle tool

paint_ellipse(r, g, b, x1, y1, x2, y2, opacity)
-- Ellipse tool

paint_text(r, g, b, x, y, fontName, fontHeight, fontBold,
fontItalic, string)           -- Text tool

paint_gradient(r1, g1, b1, r2, g2, b2, r3, g3, b3, r4, g4, b4,
colorCount, x1, y1, x2, y2, gradType)  -- Gradient tool

paint_mask_ellipse(x1, y1, x2, y2, maskLevel)        -- Old Mask
Ellipse tool

paint_mask_rectangle(x1, y1, x2, y2, maskLevel)      -- Old Mask
Rectangle tool

paint_mask_wand(x, y, tolerance, opacity, maskWandType)
-- Mask Wand tool

paint_copy(x1, y1, x2, y2)
-- Old Copy tool

paint_paste(x, y, size, rotateAngle, highQualityFlag)
-- Old Paste tool

paint_warp(x1, y1, x2, y2, warpMode, warpStrength)
-- Warp tool

paint_rotate_brush(x1, y1, x2, y2)
```

```
variantVal  -- The variant slider value

slideVal1 -- The sliderValx values come from gui sliders when
defined.

slideVal2

slideVal3

slideVal4

slideVal5

slideVal6

flagVal1 -- The flagValx values come from the gui checkboxes

flagVal2

flagVal3

flagVal4

flagVal5

flagVal6

styleVal1 -- 1 based index into the option choosen

styleVal2
```

Merge Types

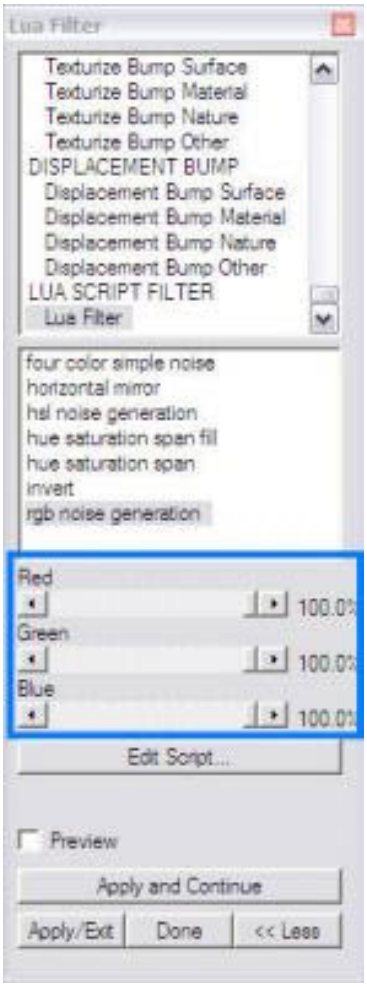
```
MERGE_NORMAL
MERGE_MULTIPLY
MERGE_SCREEN
MERGE_DARKEN
MERGE_LIGHTEN
MERGE_DIFFERENCE
MERGE_NEGATION
MERGE_EXCLUSIO
MERGE_OVERLAY
MERGE_HARD_LIGHT
MERGE_SOFT_LIGHT
MERGE_DODGE
MERGE_BURN
MERGE_REFLECT
MERGE_GLOW
MERGE_ADD
MERGE_SUBTRACT
MERGE_COLOR
MERGE_HARD_COLOR
MERGE_HSL_HUE
MERGE_HSL_SAT
MERGE_HSL_LUM
MERGE_HSL_COLOR
MERGE_TEXTURIZE
MERGE_TEXTURIZE2
MERGE_TEXTURIZE3
```

TwistedBrush Lua Filter User Interface Directives

Within a comment in the Lua script a directive can be defined for creating up to 6 user slider controls and up to 6 check box controls that appear within the TwistedBrush Filter dialog. This allows users to dynamically adjust values in the script without ever needing to look at a script. In addition this also allows for the filters to be recorded like all the other filters in TwistedBrush. The format of this directives are:

```
--@TBCONFIG VARIANT: name, default value
--@TBCONFIG SLIDER: name, default value
--@TBCONFIG FLAG: name, default value
--@TBCONFIG STYLE: name, default index, choice1, choice2, ...
choice20
--@TBCONFIG INFO: message for the info box.
```

Example of the user interface directives



For example these three lines in a script will define three sliders Red, Green and Blue with values of a middle gray.

```
--@TBCONFIG SLIDER: Red, 50.0
```

```
--@TBCONFIG SLIDER: Green, 50.0
--@TBCONFIG SLIDER: Blue, 50.0
```

Above is the Filter dialog that shows the sliders that will be created based on the directives. However, the sample image shows values of 100.0 for each rather than the default shown above (it's a screen shot from a different example)