EFL Vector Graphics

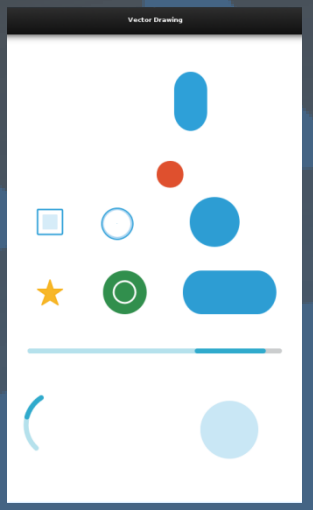
Subhransu Mohanty

Content

- Overview
- Object and APIs
- Examples
- Render Sequence

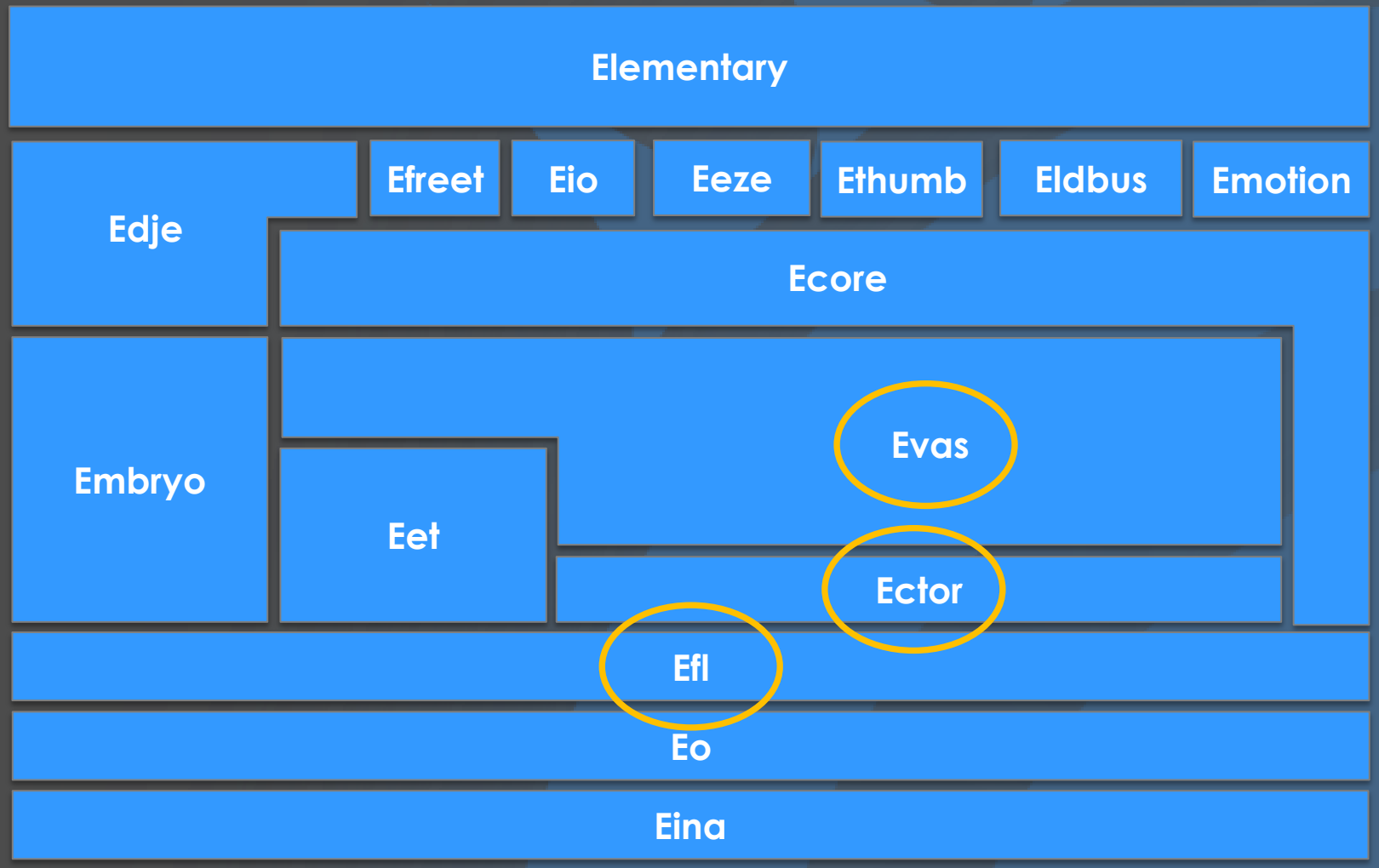
Overview

- What is EFL Vector Graphics ?
  - Retain mode drawing canvas.
  - Supports antialiasing vector drawing
    - Line, Curves, Shapes or Polygons.
    - Solid filling , Gradient Filling , Stroking.
    - Supports affine transformation.
    - Shape interpolation.
  - Well integrated into the Evas.
    - All evas object feature's works out of box.
  - Supports multiple drawing backend.
    - Cairo, Native
  - Can use hardware acceleration if backend Supports.



Overview

- New EFL structure for Vector Graphics



Overview

Graphics Canvas :

Canvas + Object + Drawing Code

Evas :

The canvas , Object.

Ector :

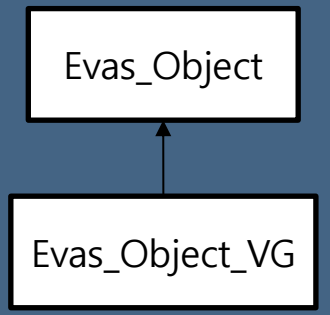
Drawing Code(Backends)

Efl :

Class and Interface shared by both evas and ector

VG Canvas : Evas_Object_VG (1/2)

- It is the canvas of vector graphics.
- It manages vg object's lifecycle and responsible for showing it on Evas Canvas.
- Supports hierarchy of vg objects to achieve complex use case.
- Any property change of vg object is automatically reflected in the canvas.



VG Canvas : Evas_Object_VG (2/2)

Evas_Object ***evas_object_vg_add(Evas** *e);

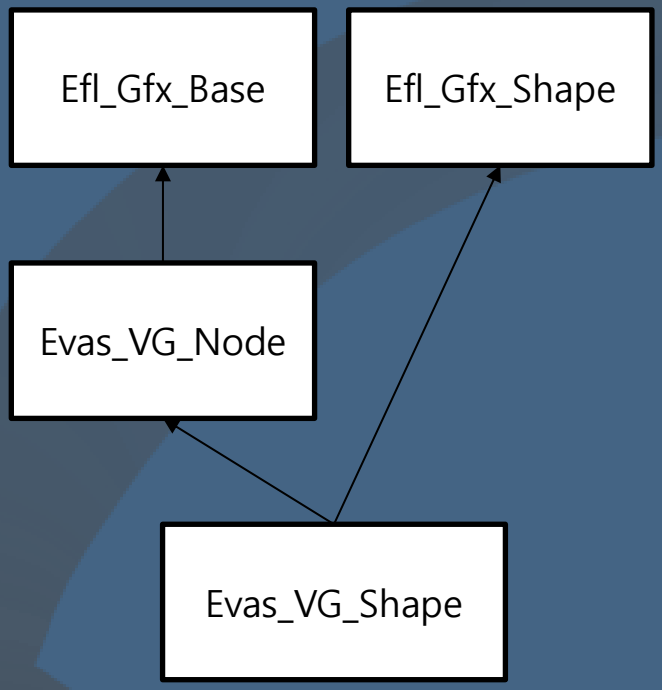
Creates a new vector object on the given Evas @p e canvas.

Efl_VG ***evas_object_vg_root_node_get(Evas_Object** *vg);

Get the root node of the evas_object_vg.

VG Object : Efl_VG (1/4)

- It is the vg object recognized by vg canvas.
- It has interface for adding
  - Shape data.
  - Fill and Stroke info.
  - Transformation info.



VG Object : Efl_VG (2/4)

void **evas_vg_node_color_set** (Efl_VG *obj, int r, int g, int b, int a);

Sets the general/main color of the given Evas object to the given one.

void **evas_vg_shape_stroke_color_get** (Efl_VG *shape, int r, int g, int b, int a);

Sets the color to be used for stroking the path.

void **evas_vg_shape_stroke_width_set** (Efl_VG *shape, double width);

Sets the stroke width to be used for stroking the path.

void **evas_vg_shape_stroke_cap_set** (Efl_VG *shape, Efl_Gfx_Cap c);

Sets the cap style to be used for stroking the path. The cap will be used for capping the end point of a open sub path.

void **evas_vg_shape_stroke_join_set** (Efl_VG *shape, Efl_Gfx_Join join);

Sets the join style to be used for stroking the path. The join style will be used for joining the two line segment while stroking the path.

VG Object : Efl_VG(3/4)

void **evas_vg_shape_shape_append_move_to** (Efl_VG *shape, double x, double y);

Moves the current point to the given point, implicitly starting a new subpath and closing the previous one.

void **evas_vg_shape_shape_append_line_to** (Efl_VG *shape, double x, double y);

Moves the current point to the given point, implicitly starting a new subpath and closing the previous one. If no current position present, it draws a line to itself, basically a point.

void **evas_vg_shape_shape_append_close**(Efl_VG *shape);

Closes the current subpath by drawing a line to the beginning of the subpath, automatically starting a new path. The current point of the new path is (0, 0).

VG Object : Efl_VG(4/4)

void **evas_vg_shape_shape_path_set** (Efl_VG *shape, Efl_Gfx_Path_Command *op, double *points);

Set the list of commands and points to be used to create the content of shape.

void **evas_vg_shape_shape_append_rect** (Efl_VG *shape, double x, double y, double w, double h, double rx, double ry);

Append the given rectangle with rounded corner to the path. The xr and yr arguments specify the radii of the ellipses defining the corners of the rounded rectangle.

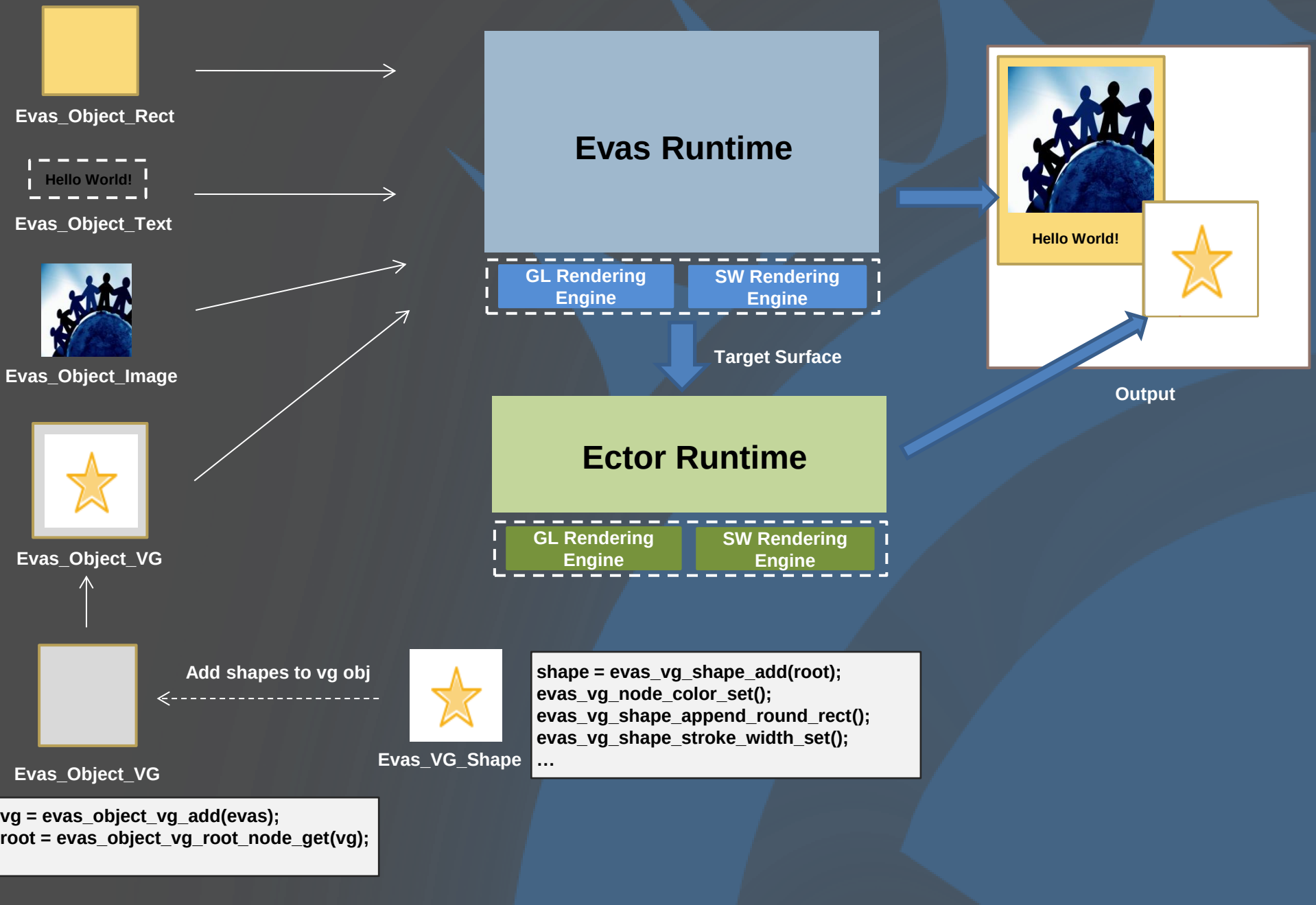
void **evas_vg_shape_shape_append_circle** (Efl_VG *shape, double x, double y, double radius);

Append a circle with given center and radius.

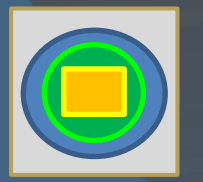
VG Backend: ECTOR_BACKEND

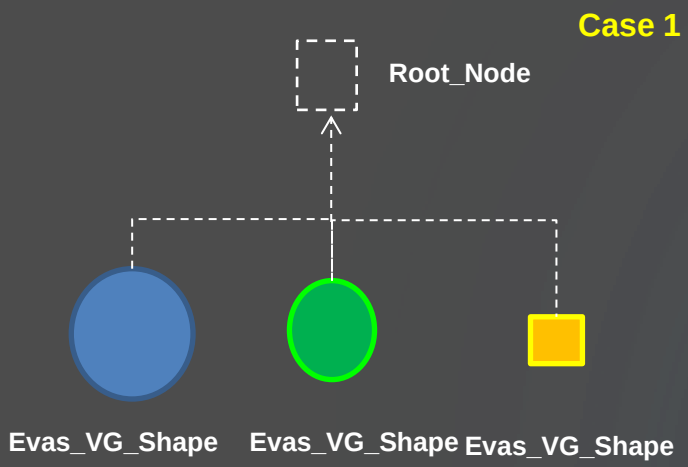
- Environment variable to switch between backend
  - Not set , uses **CAIRO** backend.
  - ECTOR_BACKEND= **default** , uses **NATIVE** backend.

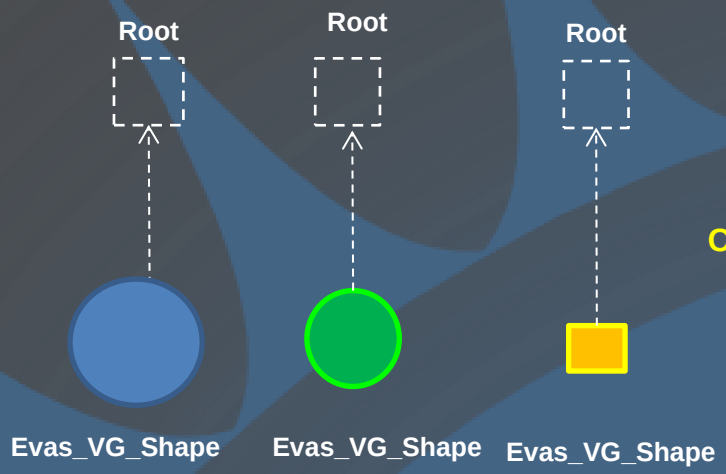
Example

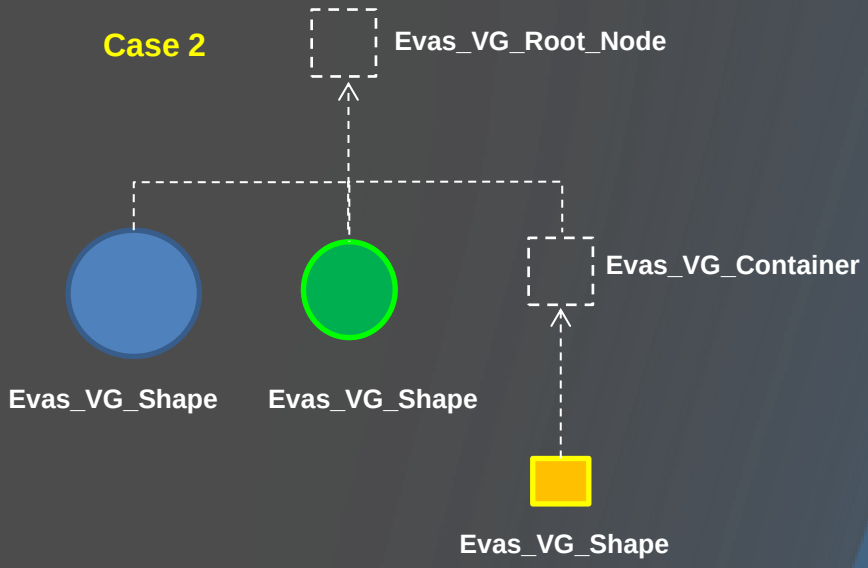

Evas_Object_Rect
Hello World!
Evas_Object_Text
Evas_Object_Image
Evas_Object_VG
Evas Runtime
GL Rendering Engine
SW Rendering Engine
Target Surface
Ector Runtime
GL Rendering Engine
SW Rendering Engine
Hello World!
Output
Add shapes to vg obj
Evas_Object_VG
Evas_VG_Shape
vg = evas_object_vg_add(evas);
root = evas_object_vg_root_node_get(vg);
shape = evas_vg_shape_add(root);
evas_vg_node_color_set();
evas_vg_shape_append_round_rect();
evas_vg_shape_stroke_width_set();
…

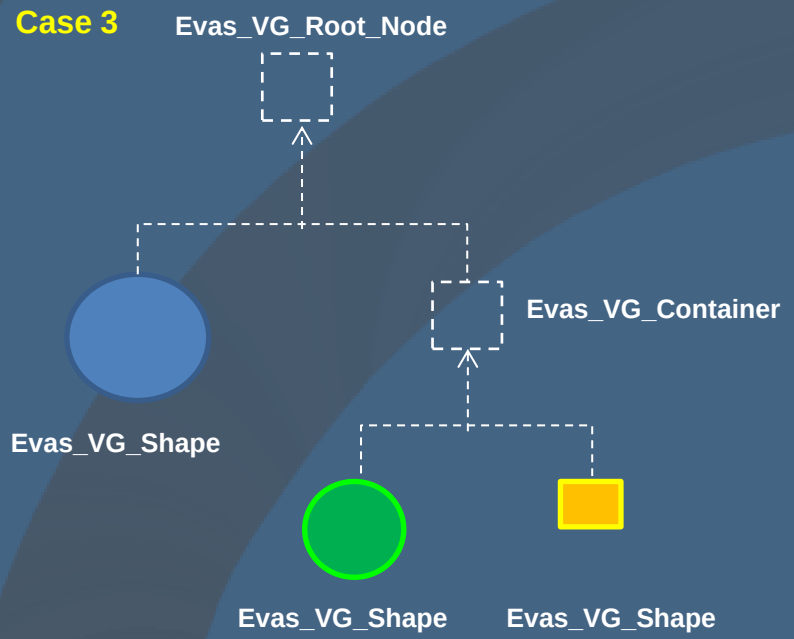
Example : (Node Tree)











Example – Circle (1/2)

```
/* gcc eina_file.c -o eina_file `pkg-config --cflags --libs eo efl ector elementary` */


//To use the EO APIs, Declare below macros
#define EFL_BETA_API_SUPPORT 1
#define EFL_EO_API_SUPPORT 1

#include <Elementary.h>

void draw_circle2(Evas *evas)
{
   //Create a vector object
  Evas_Object *vg = evas_object_vg_add(evas);
  evas_object_resize(vg, 200, 200);
  evas_object_show(vg);

  //Attach a shape node to vector object
  Evas_VG_Root_Node *root = evas_object_vg_root_node_get(vg);
  Efl_VG *shape = evas_vg_shape_add( root);       //Circle Shape
  Efl_VG *shape2 = evas_vg_shape_add( root);     //Vertical Line Shape

  //This API fills the path_cmd  and path_pts with the circle path information.
 evas_vg_shape_shape_append_circle(shape,
                                   100,        //circle center position x
                                   100,        //circle center position y
                                   100);       //circle radius

   //Set Circle Shape Attributes
   evas_vg_node_color_set(shape, 80, 141, 102, 255);        //shape node color

   //Continue to  next page
```

Example – Circle 2 (2/2)

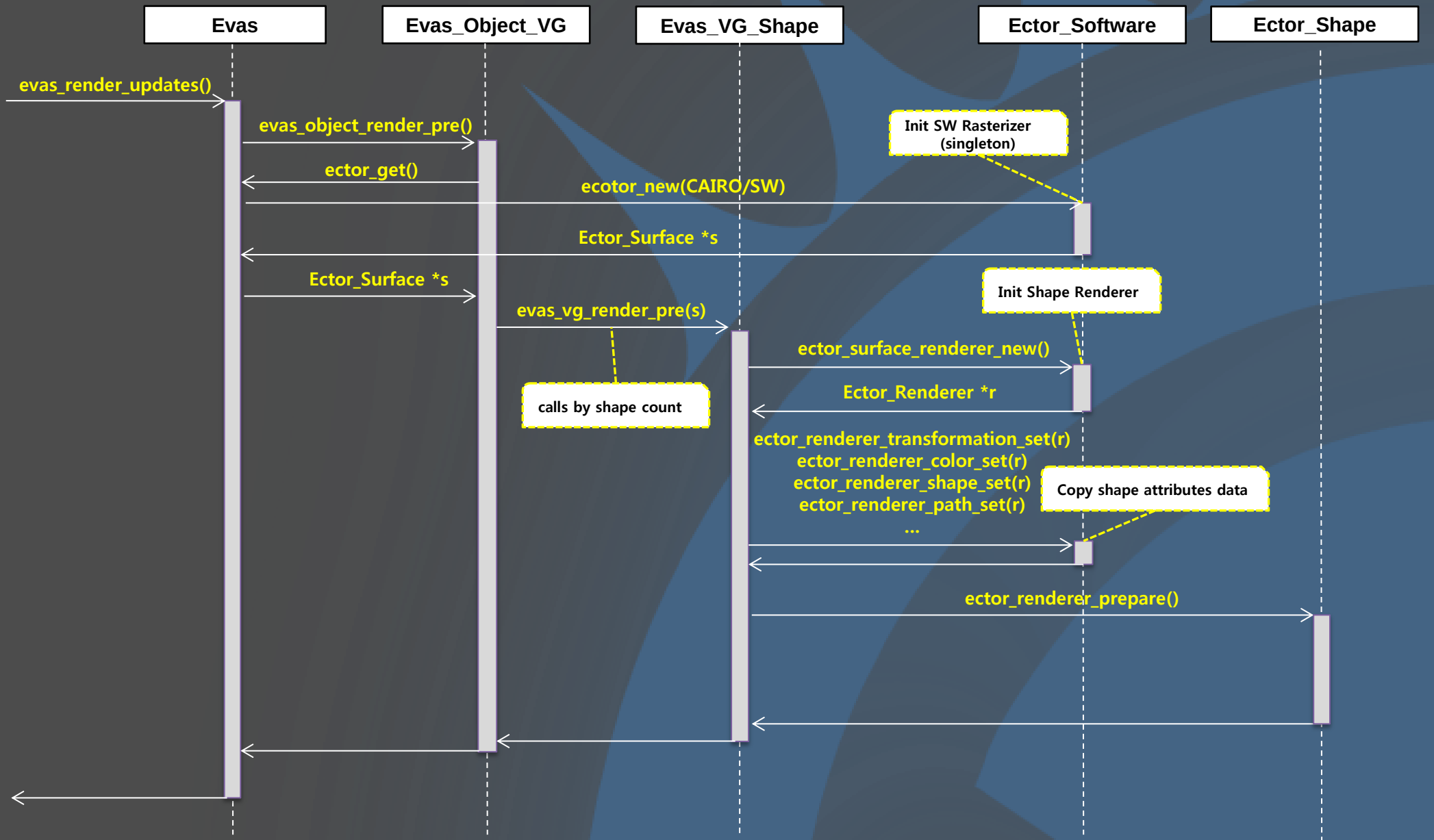
```
  //Set Line drawing commands
  evas_vg_shape_shape_append_move_to(shape2,
                                     100,        //Start position x of the line drawing
                                     60);        //Start position y of the line drawing

  evas_vg_shape_shape_append_line_to(shape2
                                     100,       //End position x of the line drawing
                                     140);      //End position y of the line drawing

   //Set Line Shape Attributes
   evas_vg_shape_stroke__width_set(shape2, 1.75);      //Stroke width size
   evas_vg_shape_stroke_cap_set(shape2, EFL_GFX_CAP_ROUND);         //Stroke cap style
   evas_vg_shape_stroke_color_set(shape2, 255, 255, 255, 255);        //Stroke color
}
```

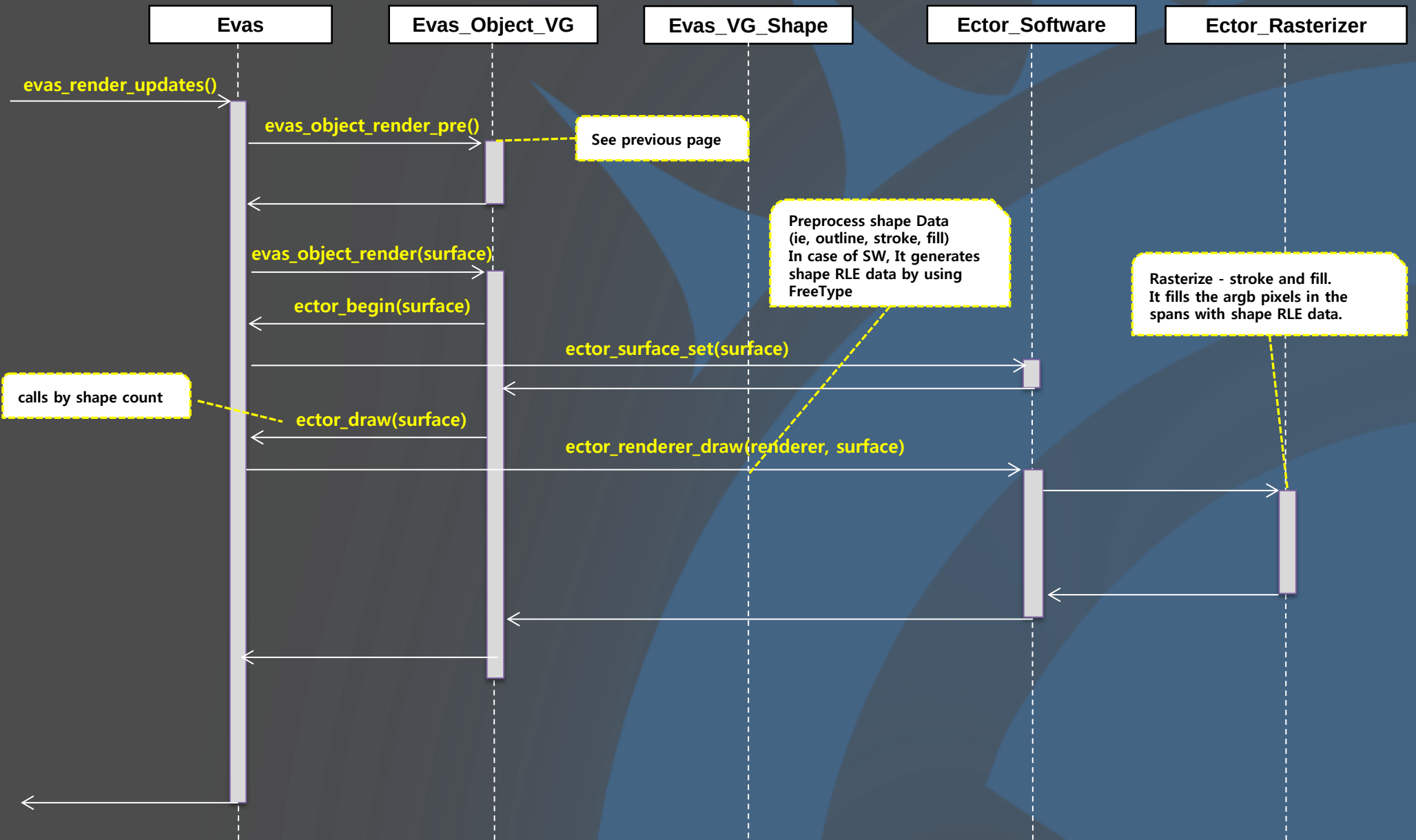
Rendering Sequence (SW/GL Backened)

- Shape rendering – render_pre()



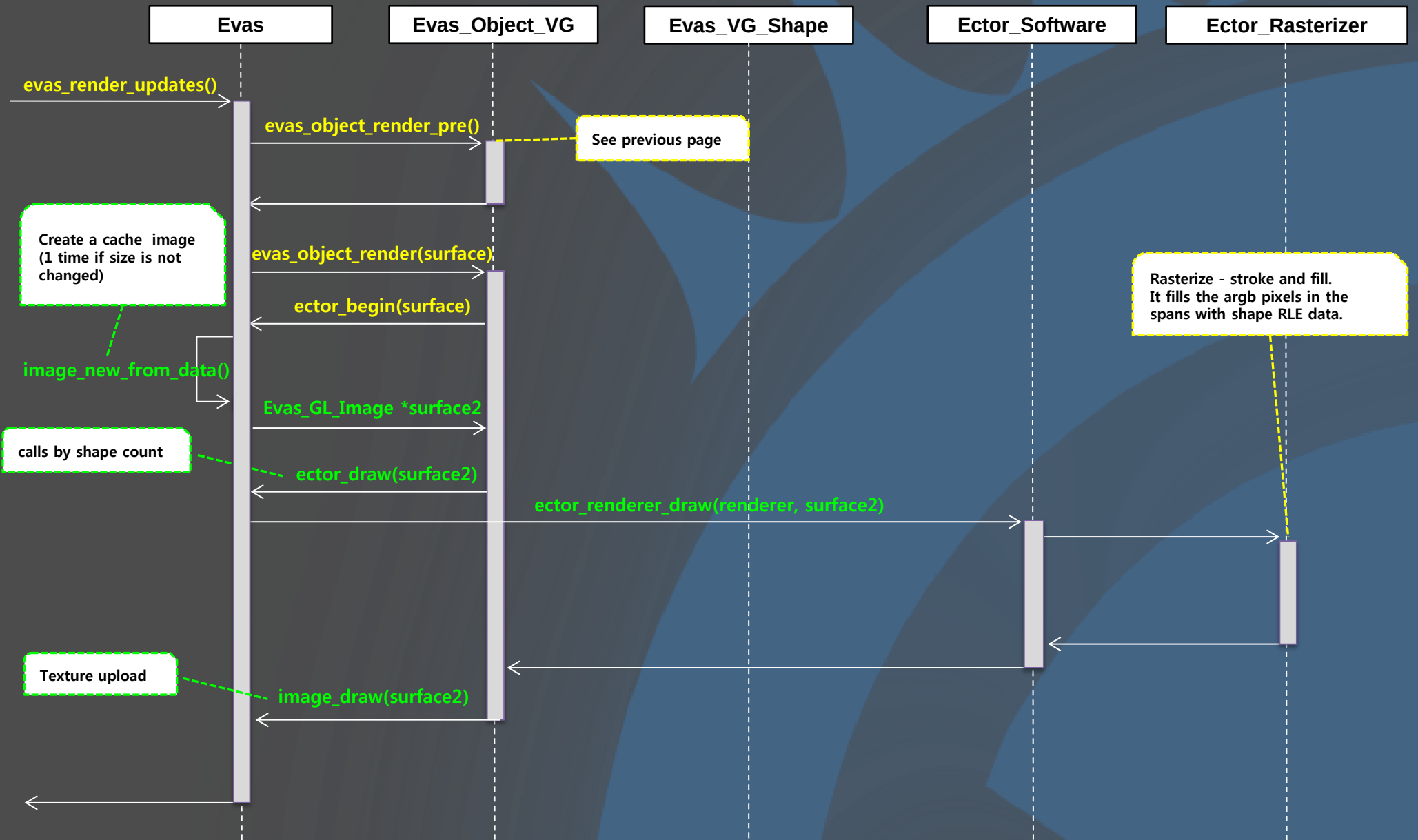
Rendering Sequence (SW Backened)

- Shape rendering - render()



Rendering Sequence (GL Backened)

- Shape rendering - render()



Thank you