Real-time Graphics in Blender

Jeroen Bakker
jeroen@blender.org

The human brain ...

- Eye only provides very limited information
- Fills in blank spots
  - Temporal feedback
  - Up-scaling
  - Hallucinates
- Can even imagine from just a piece of text
- But with recent developments ...

… it gets lazy

Who am I

- Demo scene (1994-1998)
- Enterprisy (1998-2017)
- Contributor since 2008
- Employed since 2018
- Co-owner of Viewport & EEVEE module
- Making sure that Blender runs well on “any” platform
  - Workbench engine
  - Vulkan backend
- First point of contact for many GPU related issues

CHARGE









User interface

Where it all starts...

source > blender > windowmanager > intern > wm.cc > ...

```
void WM_main(bContext *C)
{
  /* Single refresh before handling events.
   * This ensures we don't run operators before the depsgraph has been evaluated. */
  wm_event_do_refresh_wm_and_depsgraph(C);

  while (true) {

    /* Get events from ghost, handle window events, add to window queues. */
    wm_window_events_process(C);

    /* Per window, all events to the window, screen, area and region handlers. */
    wm_event_do_handlers(C);

    /* Events have left notes about changes, we handle and cache it. */
    wm_event_do_notifiers(C);

    /* Execute cached changes draw. */
    wm_draw_update(C);
  }
}
```

Data model (DNA)

source > blender > makesdna > C DNA_material_types.h > Material > specr

Habib Ganbiche, 4 weeks ago | 15 authors (Campbell Barton and others)

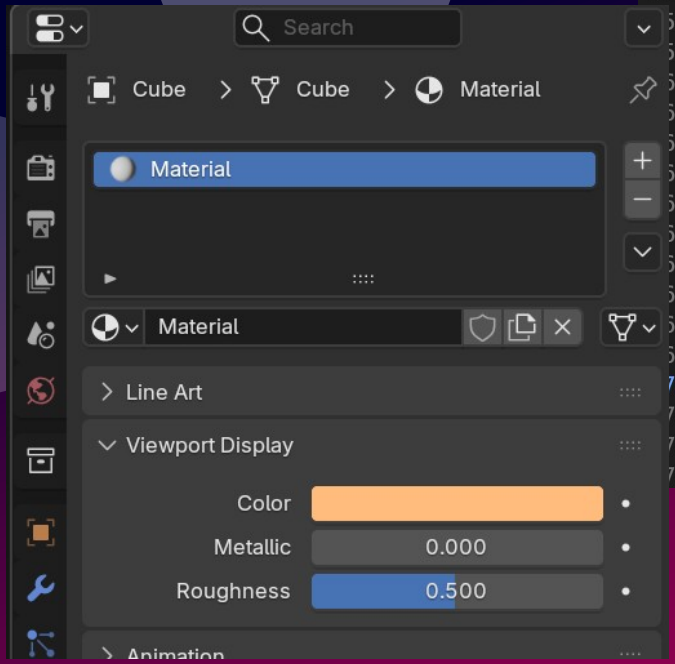
```
163  typedef struct Material {
164  #ifdef __cplusplus
165    DNA_DEFINE_CXX_METHODS(Material)
166    /** See #ID_Type comment for why this is here. */
167    static constexpr ID_Type id_type = ID_MA;
168  #endif
169
170    ID id;
171    /** Animation data (must be immediately after id for utilities to use it). */
172    struct AnimData *adt;
173
174    short flag;
175    /** Rendering modes for EEVEE. */
176    char surface_render_method;
177    char _pad1[1];
178
179    /* Colors from Blender Internal that we are still using. */
180    float r, g, b, a;
181    float specr, specg, specb;      Hans Lambermont, 23 years ago • Initial revision
182    float alpha DNA_DEPRECATED;
183    float ray_mirror DNA_DEPRECATED;
184    float spec;
185    /** Renamed and inversed to roughness. */
186    float gloss_mir DNA_DEPRECATED;
```

Define Access (RNA)

source > blender > makesrna > intern > rna_material.cc > rna_def_material_display(StructRNA *)

```
57   #ifdef RNA_RUNTIME
430  }
431
432  #else
433
434  static void rna_def_material_display(StructRNA *srna)
435  {
436    PropertyRNA *prop;
437
438    prop = RNA_def_property(srna, "diffuse_color", PROP_FLOAT, PROP_COLOR);
439    RNA_def_property_float_sdna(prop, nullptr, "r");
440    RNA_def_property_array(prop, 4);
441    RNA_def_property_override_flag(prop, PROPOVERRIDE_OVERRIDABLE_LIBRARY);
442    RNA_def_property_ui_text(prop, "Diffuse Color", "Diffuse color of the material");
443    /* See #82514 for details, for now re-define defaults here. Keep in sync with
444     * #DNA_material_defaults.h */
445    static const float diffuse_color_default[4] = {0.8f, 0.8f, 0.8f, 1.0f};
446    RNA_def_property_float_array_default(prop, diffuse_color_default);
447    RNA_def_property_update(prop, 0, "rna_Material_draw_update");      Brecht Van Lommel, 16 years ago
448
```

User interface



View Text Edit Select Format Templates properties_material.py

```
351
352 class MATERIAL_PT_viewport(MaterialButtonsPanel, Panel):
353     bl_label = "Viewport Display"
354     bl_context = "material"
355     bl_options = {'DEFAULT_CLOSED'}
356     bl_order = 10
357
58      @classmethod
59      def poll(cls, context):
60          mat = context.material
61          return mat and not mat.grease_pencil
62
63      def draw(self, context):
64          layout = self.layout
65          layout.use_property_split = True
66
67          mat = context.material
68
69          col = layout.column()
70          col.prop(mat, "diffuse_color", text="Color")
71          col.prop(mat, "metallic")
72          col.prop(mat, "roughness")
73
```

```
>>> bpy.data.materials['Material'].diffuse_color = (1.0, 0.5, 0.2, 1.0)
>>>
```

Define Access (RNA)

source > blender > makesrna > intern > rna_material.cc > rna_def_material_display(StructRNA *)

```
57   #ifdef RNA_RUNTIME
430  }
431
432  #else
433
434  static void rna_def_material_display(StructRNA *srna)
435  {
436    PropertyRNA *prop;
437
438    prop = RNA_def_property(srna, "diffuse_color", PROP_FLOAT, PROP_COLOR);
439    RNA_def_property_float_sdna(prop, nullptr, "r");
440    RNA_def_property_array(prop, 4);
441    RNA_def_property_override_flag(prop, PROPOVERRIDE_OVERRIDABLE_LIBRARY);
442    RNA_def_property_ui_text(prop, "Diffuse Color", "Diffuse color of the material");
443    /* See #82514 for details, for now re-define defaults here. Keep in sync with
444     * #DNA_material_defaults.h */
445    static const float diffuse_color_default[4] = {0.8f, 0.8f, 0.8f, 1.0f};
446    RNA_def_property_float_array_default(prop, diffuse_color_default);
447    RNA_def_property_update(prop, 0, "rna_Material_draw_update");      Brecht Van Lommel, 16 years ago
448
```

Notifiers

source > blender > makesrna > intern > rna_material.cc > rna_Material_active_paint_texture_index_update(bContext *, PointerRNA *)

```
57   #ifdef RNA_RUNTIME
134
135  static void rna_Material_draw_update(Main * /*bmain*/, Scene * /*scene*/, PointerRNA *ptr)
136  {
137    Material *ma = (Material *)ptr->owner_id;
138
139    DEG_id_tag_update(&ma->id, ID_RECALC_SHADING);
140    WM_main_add_notifier(NC_MATERIAL | ND_SHADING_DRAW, ma);
141  }
```

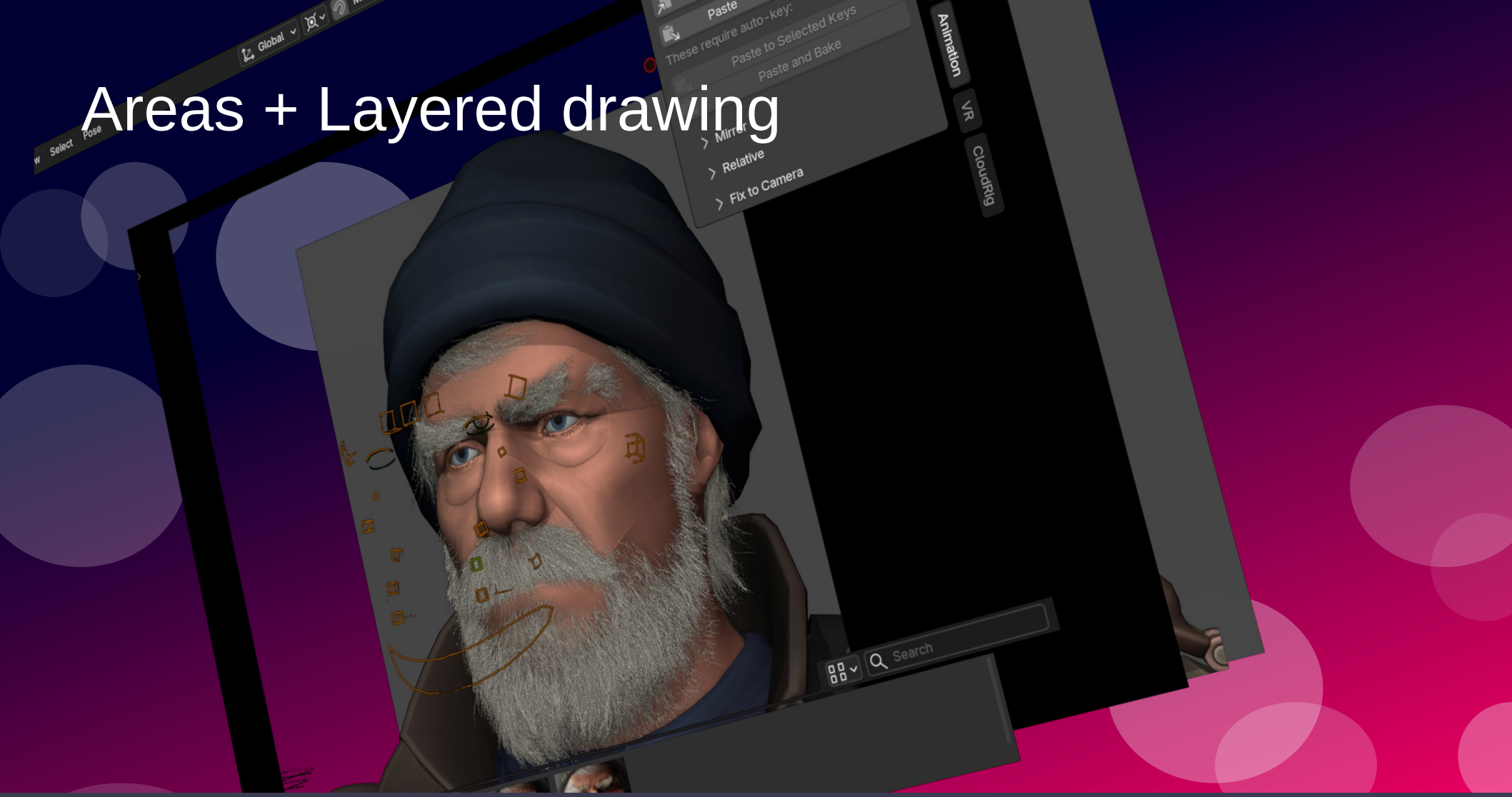
Window

Editors/Spaces

3D Viewport


Global Transform
Copy
Mirrored
Paste
These require auto-key:
Paste to Selected Keys
Paste and Bake
Mirror
Relative
Fix to Camera
Tool
View
Animation
VR
CloudRig
Search
Pose Mode
View
Select
Pose
Global
Mix
All

Areas + Layered drawing



Engine: Workbench

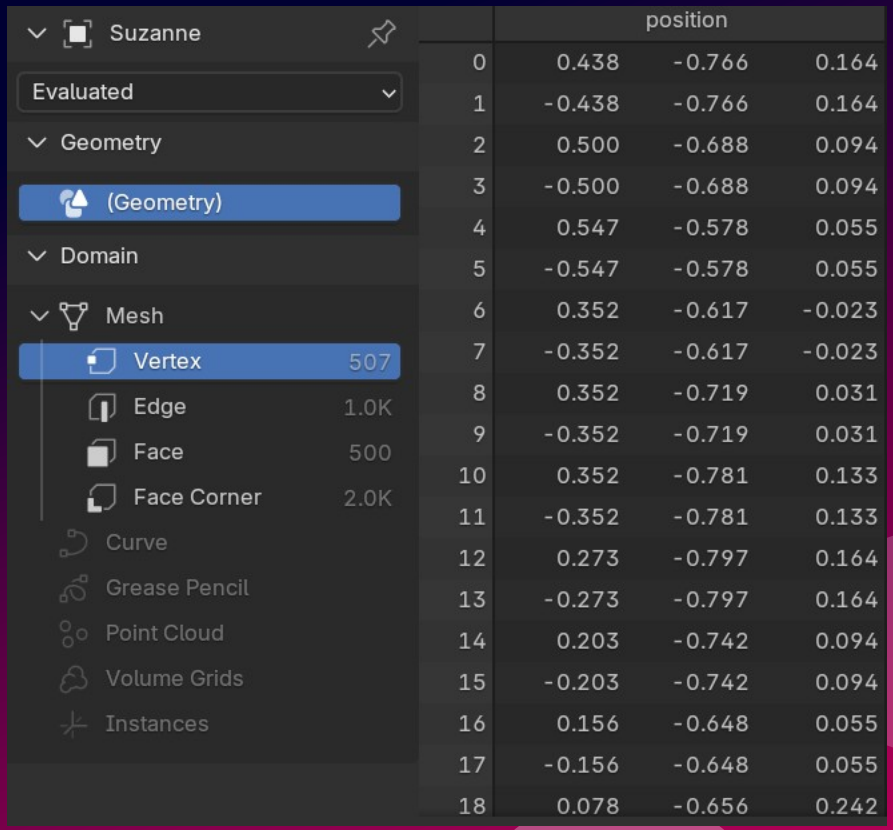
Engine: Cycles

Engine: EEVEE

So we do everything on the GPU? Right?

Geometry

- Container
  - Meshes
  - Curves
  - Grease pencil
  - Point cloud
  - Volume
  - Instances
- Mode specific acceleration structures



	position		
0	0.438	-0.766	0.164
1	-0.438	-0.766	0.164
2	0.500	-0.688	0.094
3	-0.500	-0.688	0.094
4	0.547	-0.578	0.055
5	-0.547	-0.578	0.055
6	0.352	-0.617	-0.023
7	-0.352	-0.617	-0.023
8	0.352	-0.719	0.031
9	-0.352	-0.719	0.031
10	0.352	-0.781	0.133
11	-0.352	-0.781	0.133
12	0.273	-0.797	0.164
13	-0.273	-0.797	0.164
14	0.203	-0.742	0.094
15	-0.203	-0.742	0.094
16	0.156	-0.648	0.055
17	-0.156	-0.648	0.055
18	0.078	-0.656	0.242

Mesh Edit mode

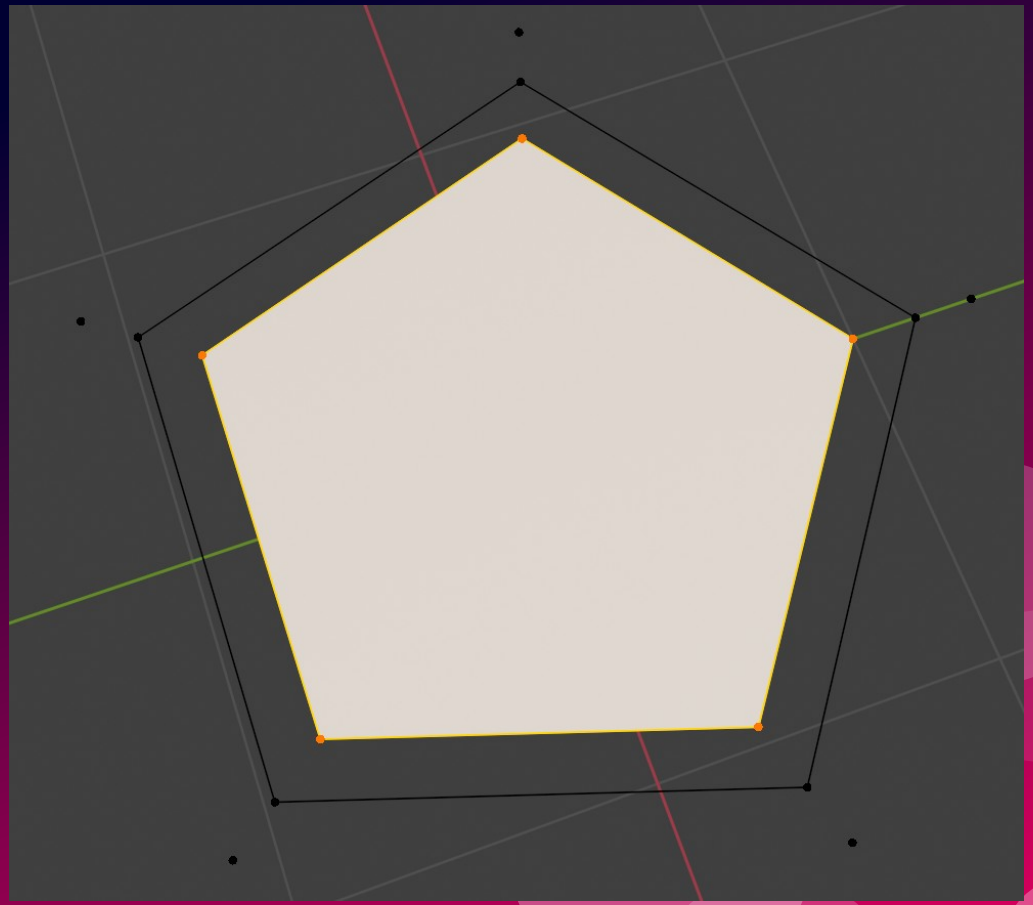
- Bmesh
- Optimized for operations/ manipulation
- Walk along
  - Polygons
  - Edges
  - Vertices

Sculpt/Paint mode

- PaintBVH
- Leaf contains draw batch
- Changed leaves are uploaded
- Topology can change every frame
- Minimize brush/feedback latency

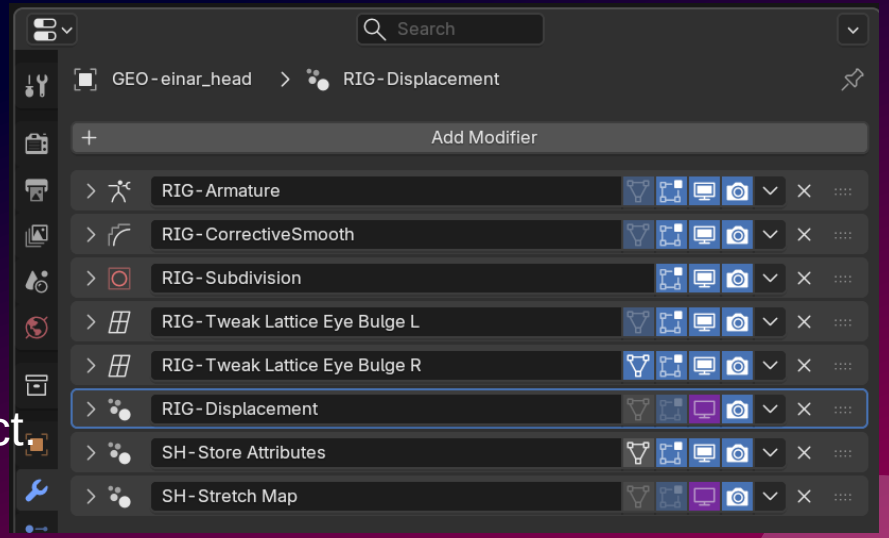
Object mode

- Ngons/Edges and vertices
- Optimized for animation
- Objects can share same geometry, but use different Modifier stack



Dependency graph

- Modifiers happens on CPU
- Animation is a modifier
- Animation is complex
  - Multiple objects in between your control rig and the actual rendered object.
    - Latices
    - Muscles
    - Clothes
  - Animate “everything”
  - Drive “any” parameter based on “anything”
  - Python evaluation along the way



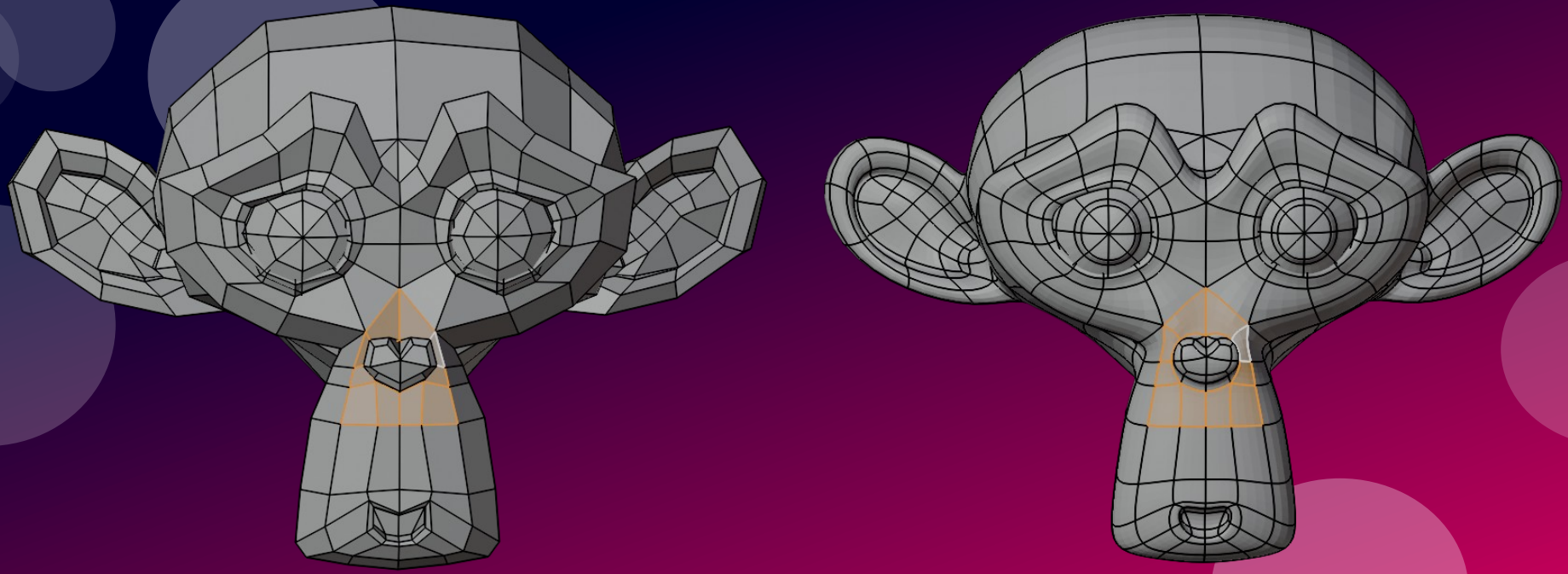
Solution: Discard what needs to be rebuild/uploaded, only build/upload what is actually used.

Remember me: Tag for update

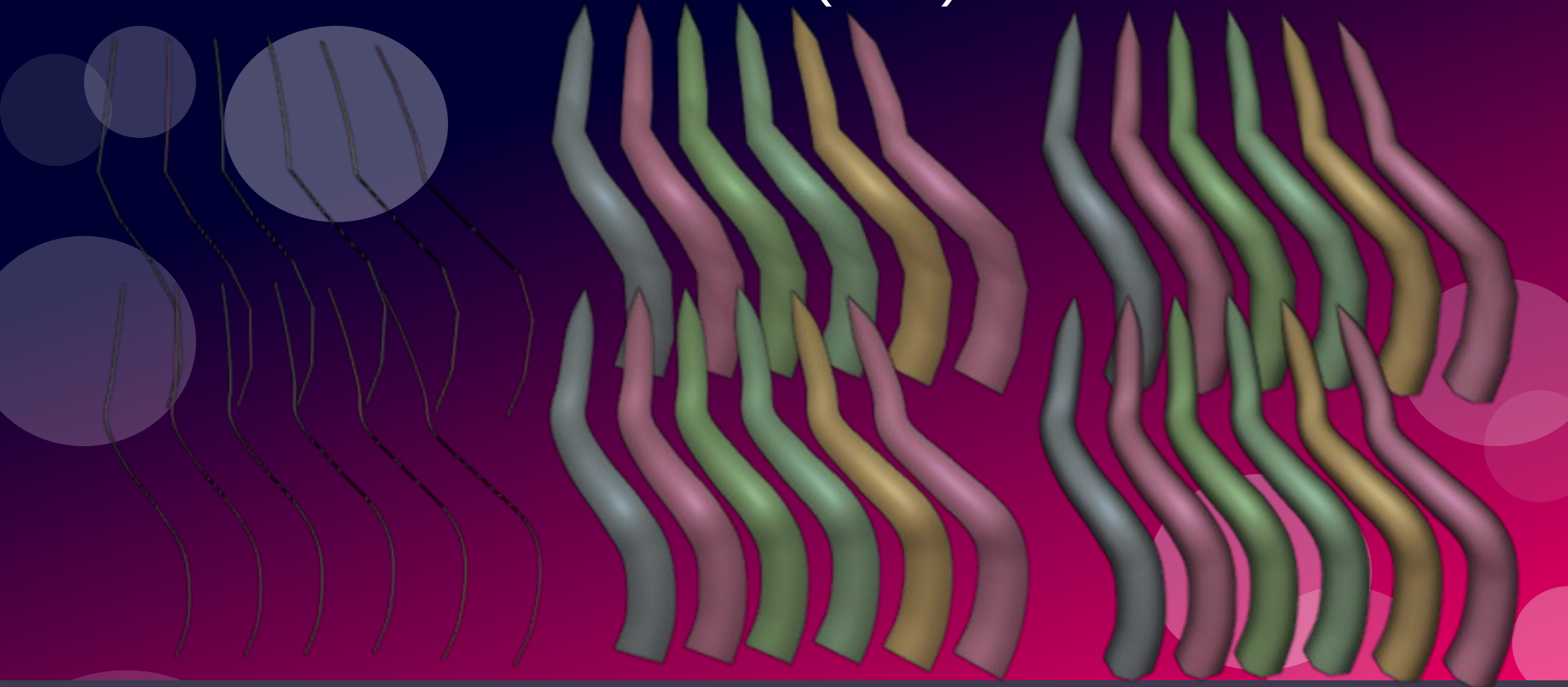
source > blender > makesrna > intern > rna_material.cc > rna_Material_active_paint_texture_index_update(bContext *, PointerRNA *)

```
57   #ifdef RNA_RUNTIME
134
135  static void rna_Material_draw_update(Main * /*bmain*/, Scene * /*scene*/, PointerRNA *ptr)
136  {
137    Material *ma = (Material *)ptr->owner_id;
138
139    DEG_id_tag_update(&ma->id, ID_RECALC_SHADING);
140    WM_main_add_notifier(NC_MATERIAL | ND_SHADING_DRAW, ma);
141  }
```

GPU accelerated subdivision



GPU accelerated curves (hair)



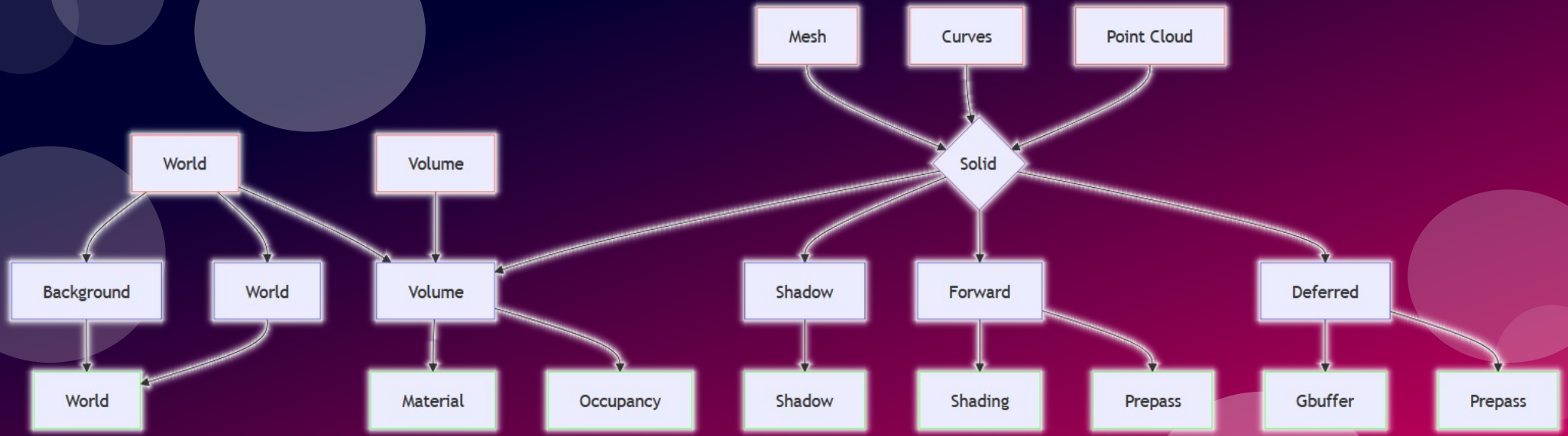
EEVEE

(Extra Easy Virtual Environment Engine)

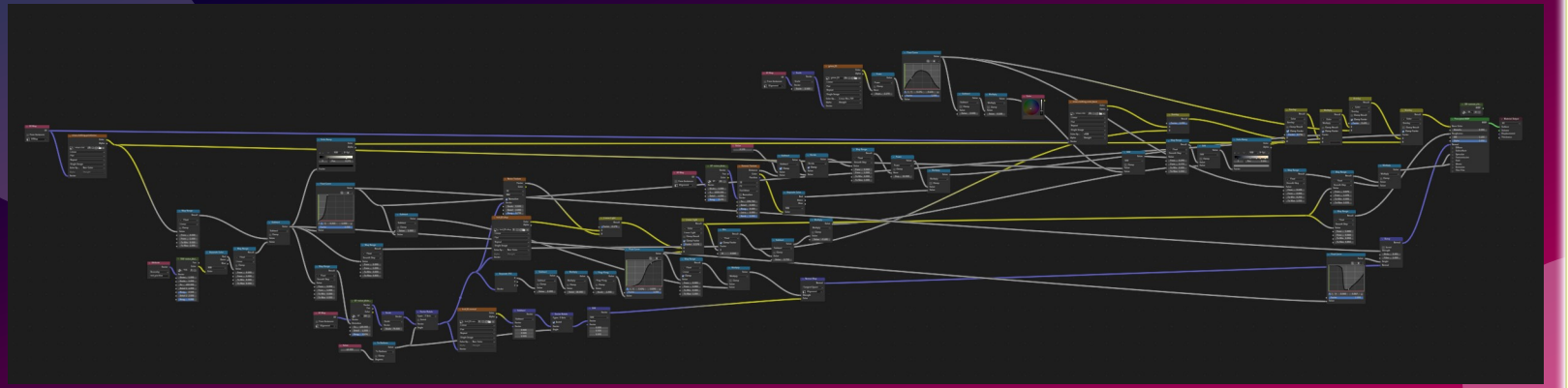
EEVEE

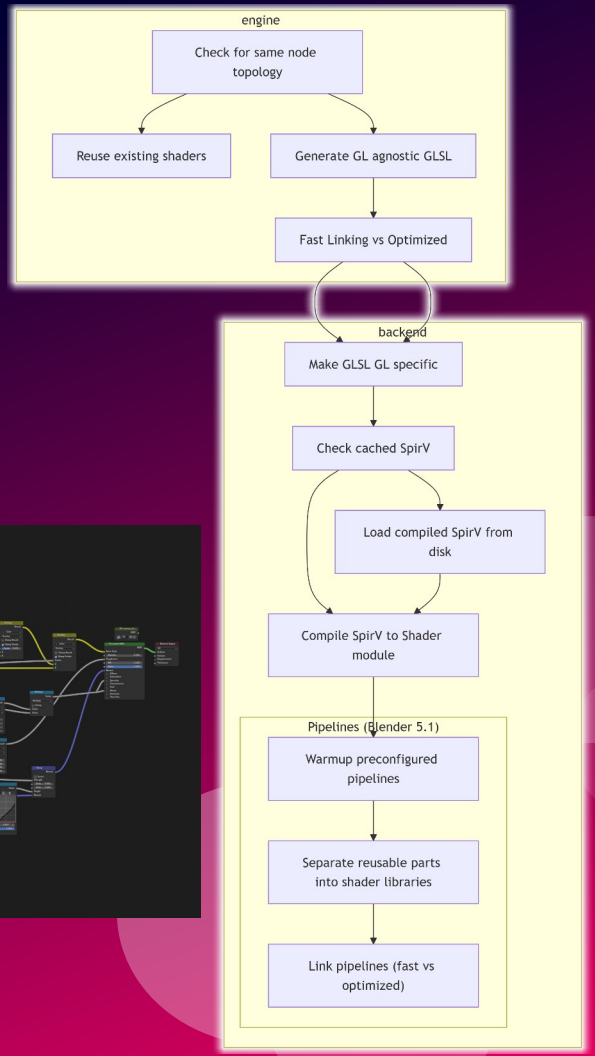
- The goal
  - Provide fast rendering and feedback
  - Using game engine like techniques
- The challenge
  - Any content can change at any moment
  - (Almost) No time for baking
  - Keeping quality when scaling between different GPUs

Materials


Mesh
Curves
Point Cloud
Solid
World
Volume
Background
World
Volume
Shadow
Forward
Deferred
World
Material
Occupancy
Shadow
Shading
Prepass
Gbuffer
Prepass

Material compilation





Shadows

- Local space shadow cubemaps
  - Point
  - Spot
  - Area
- Clipmap/Cascade
  - Sun
- Virtual shadow maps
  - cubeface
  - clipmap level
  - cascade level
- Jittered shadows
- Details: developer.blender.org/docs/features/eevee/modules/shadow/

Film

- Samples neighboring rendered pixels (weighted)
- Up-sampling (recomputed weights)
- Construct final output and render passes

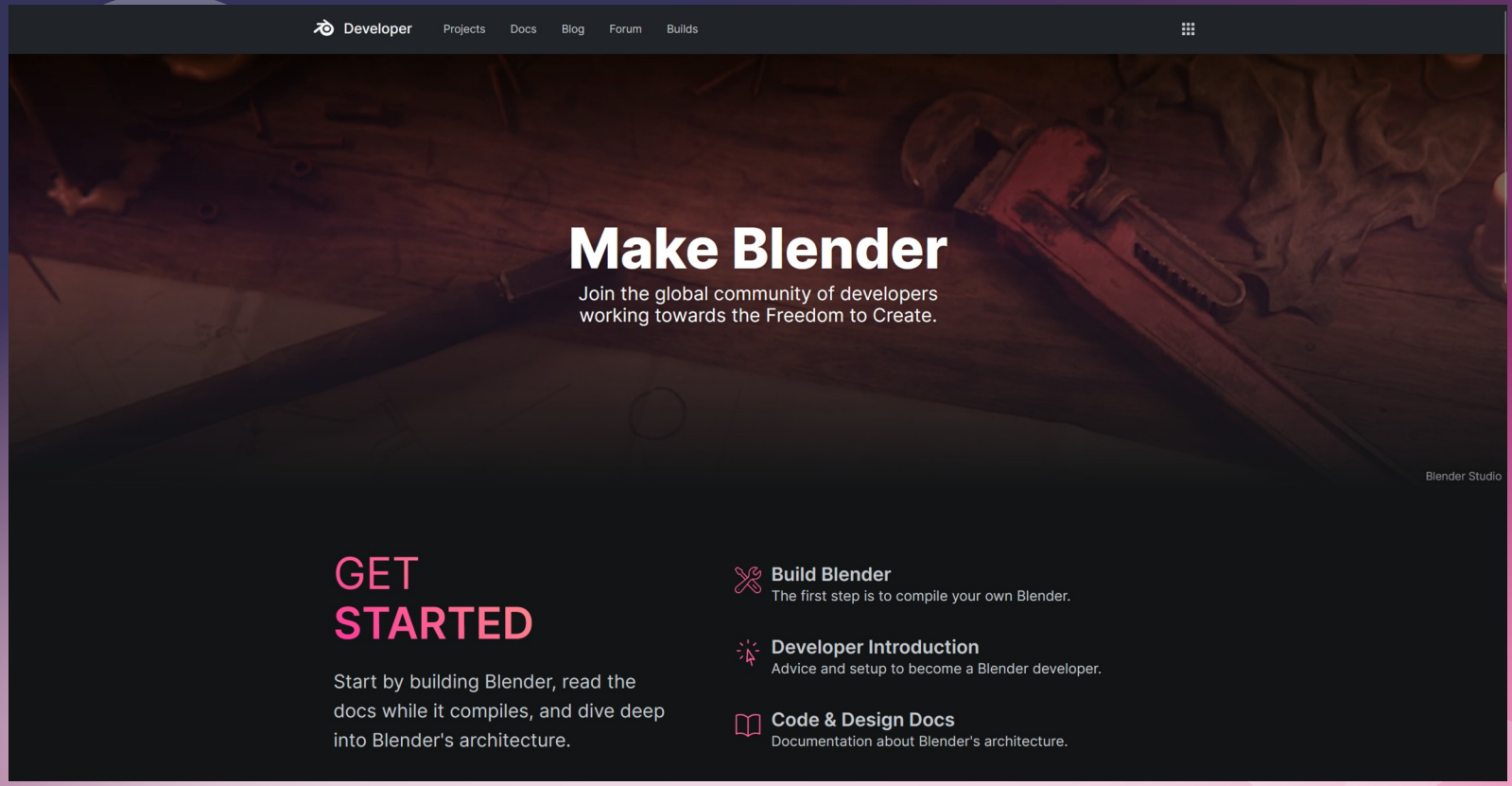
Upsampling

“Keeping complex scenes interactive on integrated GPUs”

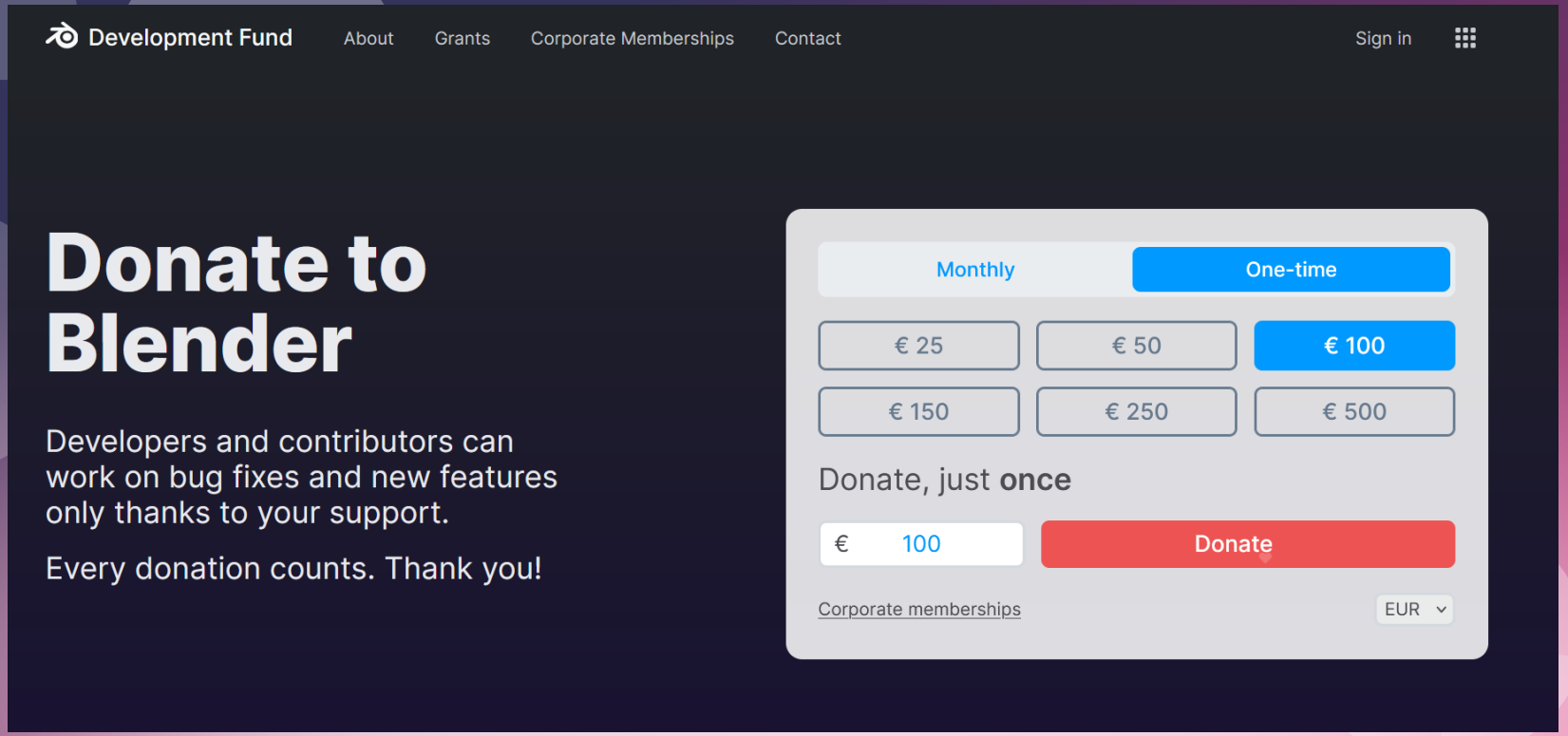
Render passes

Combined

Questions – developer.blender.org



Questions - fund.blender.org



Questions - studio.blender.org