
Shapemaker: Créations audiovisuelles procédurales musicalement synchrones

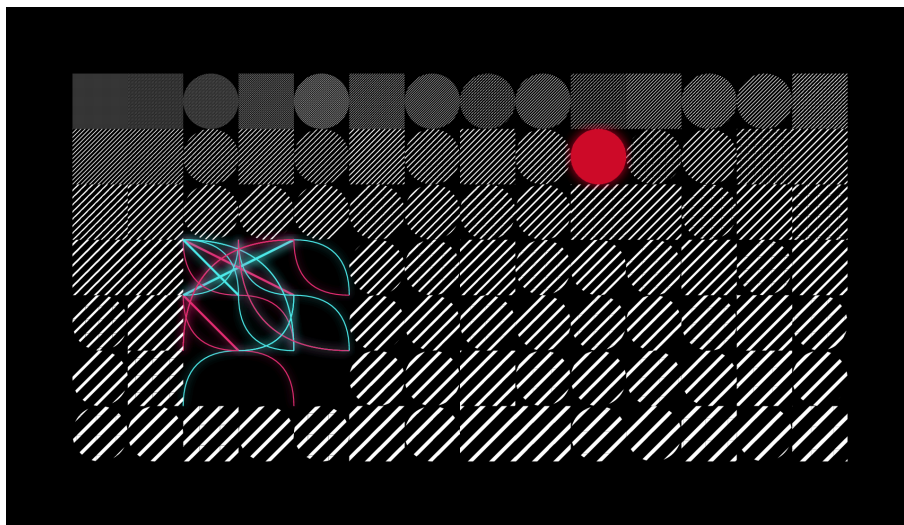
Gwenn Le Bihan

gwenn.lebihan@etu.inp-n7.fr

ENSEEIH

22 Mars 2025

Mots clés audiovisuel · procédural · SVG · Rust · WASM · WebMIDI · VST



```
use shapemaker::*;  
use rand::Rng;  
  
pub fn dna_analysis_machine() -> Canvas {  
    let mut canvas = Canvas::new(vec![]);  
  
    canvas.colormap = ColorMapping {  
        black: "#000000".into(),  
        white: "#ffffff".into(),  
        red: "#cf0a2b".into(),  
        green: "#22e753".into(),  
        blue: "#2734e6".into(),  
        yellow: "#f8e21e".into(),  
        orange: "#f05811".into(),  
        purple: "#6a24ec".into(),  
        brown: "#a05634".into(),  
        pink: "#e92e76".into(),  
        gray: "#81a0a8".into(),  
        cyan: "#4fecec".into(),  
    };  
  
    canvas.set_grid_size(16, 9);  
    canvas.set_background(Color::Black);  
}
```

```

let draw_in = canvas.world_region.resized(-2, -2);

let filaments_area =
  Region::from_bottomleft(draw_in.bottomleft().translated(2, -1), (3, 3))
    .unwrap();

let red_circle_at =
  Region::from_topright(draw_in.topright().translated(-3, 0), (4, 3))
    .unwrap()
    .random_point();

let mut hatches_layer = Layer::new("hatches");
let mut red_dot_layer = Layer::new("red dot");

for (i, point) in draw_in.iter().enumerate() {
  if filaments_area.contains(&point) {
    continue;
  }

  if point == red_circle_at {
    red_dot_layer.add_object(
      format!("red circle @ {}", point),
      Object::BigCircle(point)
        .color(Fill::Solid(Color::Red))
        .filter(Filter::glow(5.0)),
    );
  }

  hatches_layer.add_object(
    point,
    if rand::thread_rng().gen_bool(0.5) || point == red_circle_at {
      Object::BigCircle(point)
    } else {
      Object::Rectangle(point, point)
    },
    .color(Fill::Hatched(
      Color::White,
      Angle(45.0),
      (i + 5) as f32 / 10.0,
      0.25,
    )),
  );
}

let mut filaments =
  canvas.n_random_linelikes_within("splines", &filaments_area, 30);

















for (i, object) in filaments.objects.values_mut().enumerate() {
  object.recolor(Fill::Solid(if i % 2 == 0 {
    Color::Cyan
  } else {
    Color::Pink
  }));
}

filaments.filter_all_objects(Filter::glow(4.0));

canvas.layers.push(red_dot_layer);
canvas.layers.push(hatches_layer);
canvas.layers.push(filaments);
canvas
}

```

Table des matières

1	 Introduction	4
1.1	 À la recherche d'une impossible énumération des formes	4
1.2	 Une approche procédurale ?	5
1.3	 Excursion dans le monde physique	6
1.3.1	 Interprétation collective	6
1.4	 Lien musical	7
2	 Une <i>crate</i> Rust avec un API sympathique	7
3	 Render loop et hooks	7
4	 Sources de synchronisation	7
4.1	 Temps réel: WASM et WebMIDI	7
4.2	 Amplitudes de <i>stems</i>	7
4.3	 Export MIDI	7
4.4	 Fichier de projet	7
4.5	 Dépôt de « sondes » dans le logiciel de MAO	7
5	 Performance	7
6	 Conclusion	7
	Bibliographie	7

1 Introduction

1.1 À la recherche d'une impossible énumération des formes

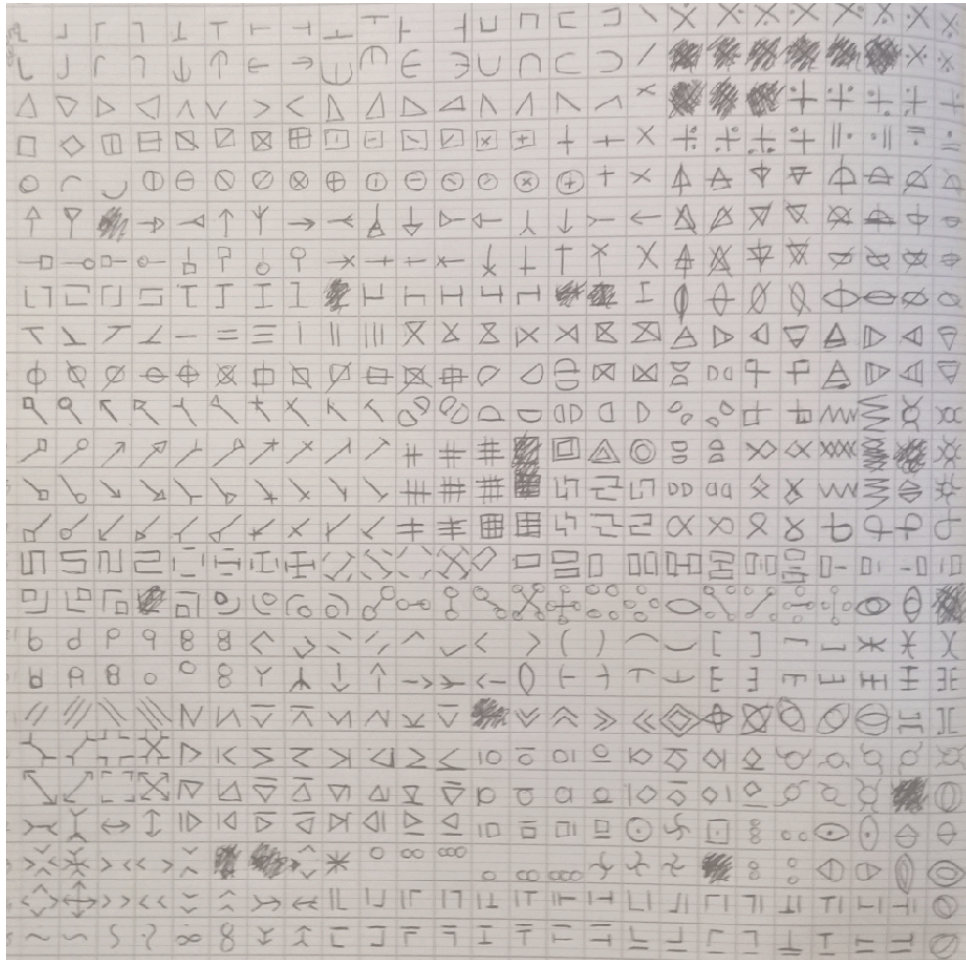


Fig. 2. – Un “alphabet” incomplet

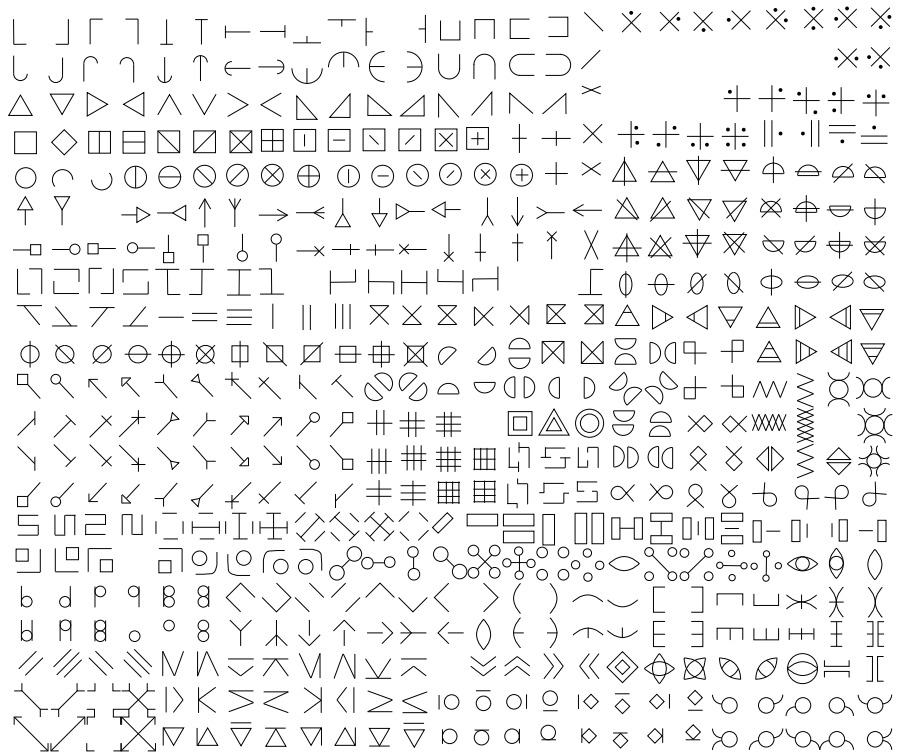
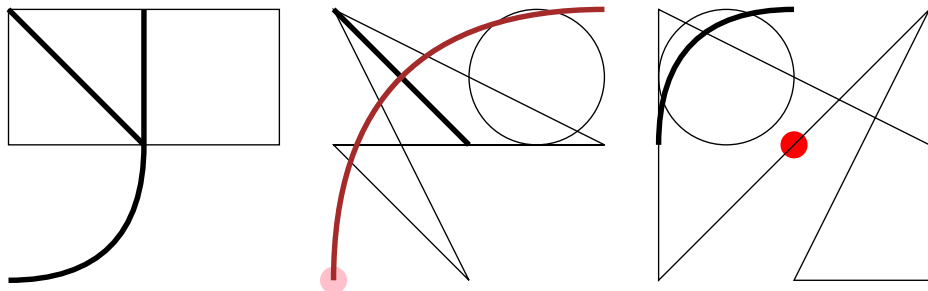
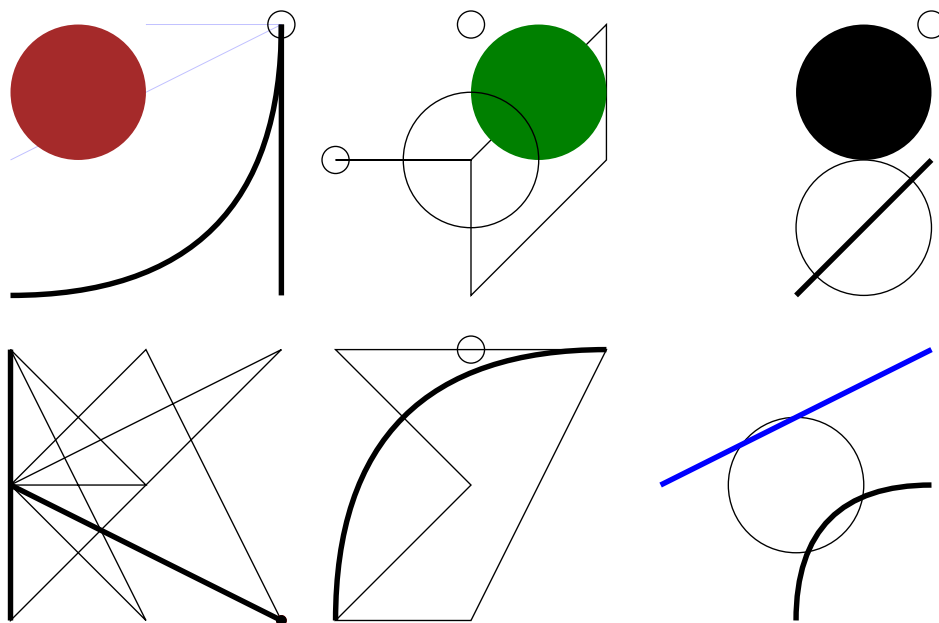


Fig. 3. – Une vectorisation sur Adobe Illustrator

1.2 ✨ Une approche procédurale ?





1.3 🌍 Excursion dans le monde physique

1.3.1 🧑🧑 Interpretation collective

<https://shapemaker.gwen.works/soon.noredir>

1.4 🎵 Lien musical

2 ❤️ Une crate Rust avec un API sympathique

3 🔗 Render loop et hooks

4 🌀 Sources de synchronisation

4.1 🎹 Temps réel: WASM et WebMIDI

4.2 📊 Amplitudes de stems

4.3 📁 Export MIDI

4.4 📁 Fichier de projet

4.5 🦄 Dépôt de « sondes » dans le logiciel de MAO

5 ⚡ Performance

6 📖 Conclusion

Bibliographie