Жизнь Пикселей

Yuri Artiukh

Something about me

- Coded a couple of websites
- I do some salsa
- I like math
- Livecode random frontend stuff on Sundays (join me!)

Why?

Awwwards Nominees

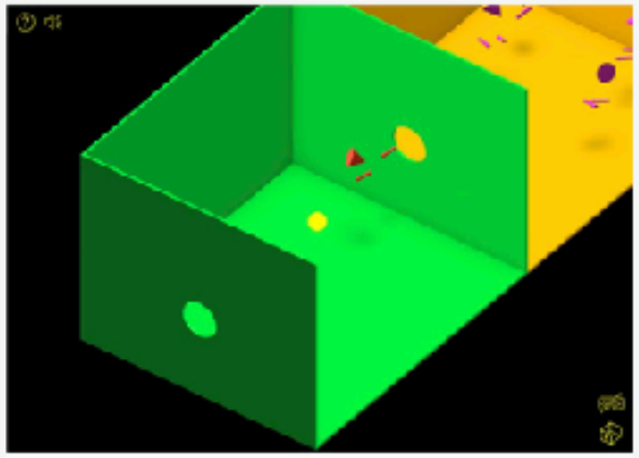
Secure | https://www.awwwards.com/websites/?award=sites_of_the_day&categories=art-illustration

MENU

AWWWARDS®

ARE YOU MEMBER? REGISTER / LOG IN

Filter By | SITES OF THE DAY | ART & ILLUSTRATION | TAG | TECHNOLOGY | COLOR | COUNTRY



Dance Tonite

By Jonathan Puckey from Netherlands

September 03, 2017 in **SOTD**

Heart of Travel

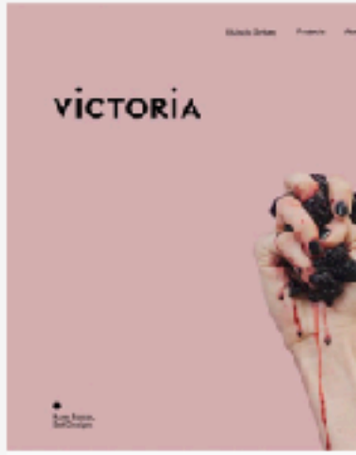
By Superhero Cheesecake from Netherlands

August 16, 2017 in **SOTD**

La peau de l'ours

By Studio Meta from France

August 11, 2017 in **SOTD**



Victoria Spicer

By Nick de Jardine from N

August 06, 2017 in **SOTD**

Tbilisi Gardens

Emmit Fenn

- Fun
- Math
- Mobile
- Money, Fame

What we need for awwward winning website?

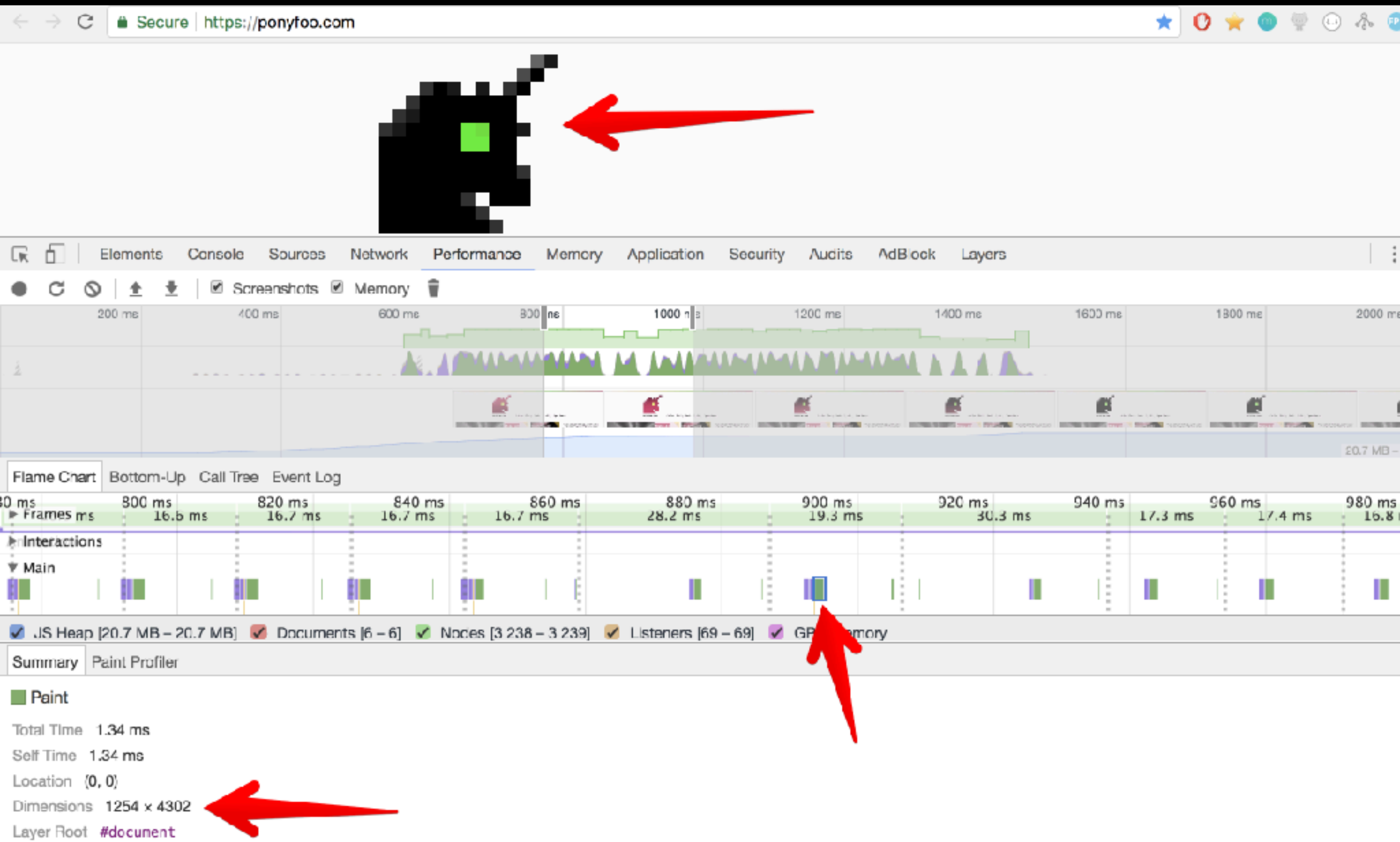
- Good designer, you can talk too
- Super-duper animations
- Blood, sweat and tears
- Good performance (well, not really)

Performance

Update Status | Add Photos/Video

What's on your mind?

Paint


Secure
https://ponyfoo.com
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
AdBlock
Layers
Screenshots
Memory
Flame Chart
Bottom-Up
Call Tree
Event Log
Frames
Interactions
Main
JS Heap [20.7 MB – 20.7 MB]
Documents [6 – 6]
Nodes [3 238 – 3 239]
Listeners [69 – 69]
Summary
Paint Profiler
Paint
Total Time 1.34 ms
Self Time 1.34 ms
Location (0, 0)
Dimensions 1254 × 4302
Layer Root #document

CPU: No throttling

800 ms 1000 ms 1200 ms 1400 ms 1600 ms

ms 1281.4 ms 1281.6 ms 1281.8 ms 1282.0 ms 1282.2 ms 1282.4 ms

26.4 ms

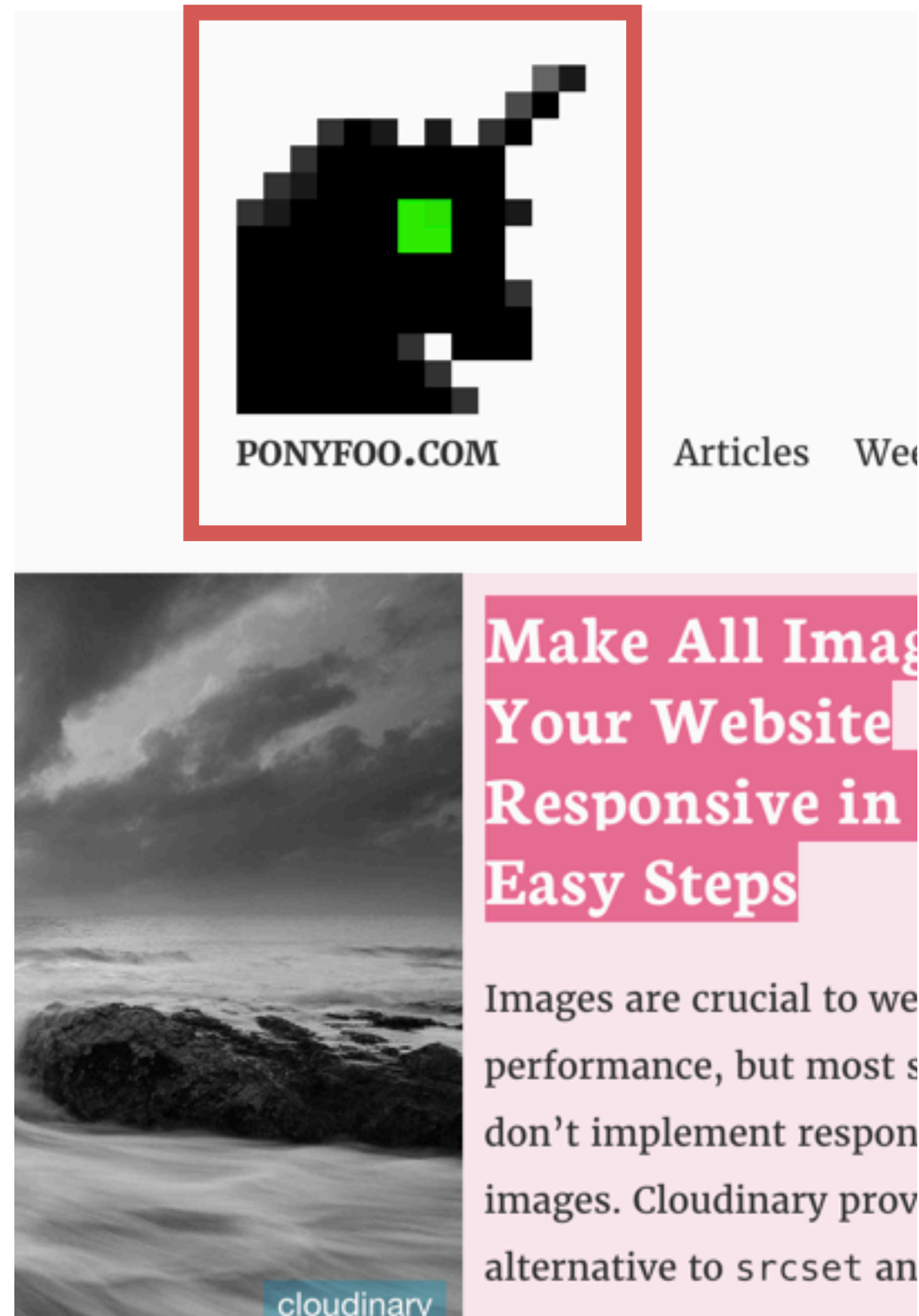
Update Layer Tree

Paint (1517 × 4389)

Paint

Total Time	1.22 ms
Self Time	1.22 ms
Location	(0, 0)
Dimensions	1517 × 4389
Layer Root	`#document`

Preview



Paint

Total Time 99 µs

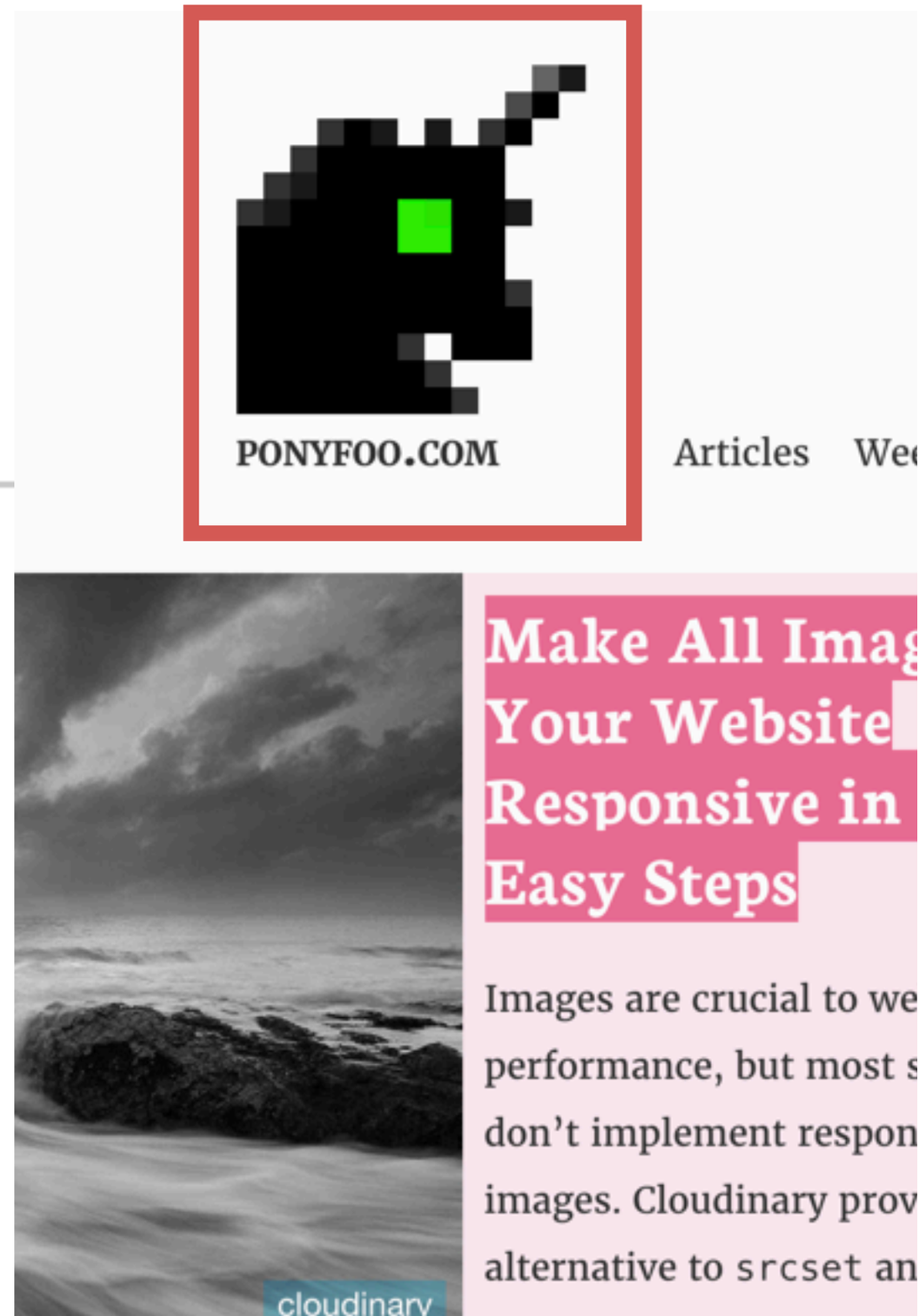
Self Time 99 µs

Location (459, 30)

Dimensions 156 × 156

Layer Root `svg.go-logo`

Preview



Elements Console Sources Network Performance Memory Application Security Audits AdBlock Layers

Screenshots Memory

200 ms 400 ms 600 ms 800 ms 1000 ms 1200 ms 1400 ms 1600 ms

Flame Chart Bottom-Up Call Tree Event Log

1470 ms 1472 ms 1474 ms 1476 ms 1478 ms 1480 ms 1482 ms 1484 ms 1486 ms 1488 ms

JS Heap [21.7 MB – 21.7 MB] Documents [6 – 6] Nodes [5 790 – 5 790] Listeners [115 – 115] GPU Memory

Summary Paint Profiler

```
▼ drawTextBlob(740.703125, 1541, paint)
    x:740.703125
    y:1541
  ▼ paint:
      textSize:12
      textScaleX:1
      textSkewX:0
      color:"#FFFCFCFC"
      strokeWidth:0
      strokeMiter:4
      flags:"AntiAlias"
      filterLevel:"Low"
      textAlign:"Left"
      strokeCap:"Butt"
      strokeJoin:"Miter"
      styleName:"Fill"
```

Coffee break

12:00

CSS

Hugo Giraudel
Accessibility Advocate
N26

«Clever, stop being so»

Asking for information online is hard. Users are cautious about the data they share. Companies are quite terrible at asking for the right piece of information, let alone phrasing it correctly.

Zarema
Front-end
Uploadcare

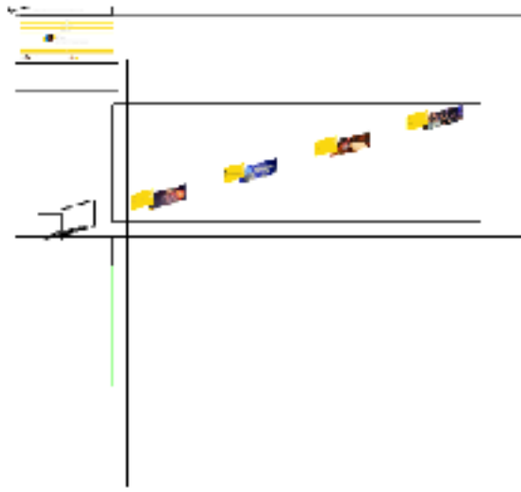
«Play

Take pho
capabilit

Elements Console Sources Network Performance Memory Application Security Audits AdBlock Layers ×

- #document(1254 × 387)
  - main.content(1254 × 387)
    - main.content(1379 × 5257)
  - .date-place(1505 × 134)
  - .date-place(93 × 47)
  - section#about(1128 × 181)
  - section#about(6835 × 2723)
  - .fact__pic-wrap(474 × 326)
  - .fact__pic-wrap(4515 × 1433)
  - section#organizers(1128 × 361)
  - section#organizers(1130 × 1461)
  - img.logo(227 × 155)
  - .fact__desc(283 × 307)
  - .facts__nav(511 × 25)
  - header.header.__logo-visible(1254 ×

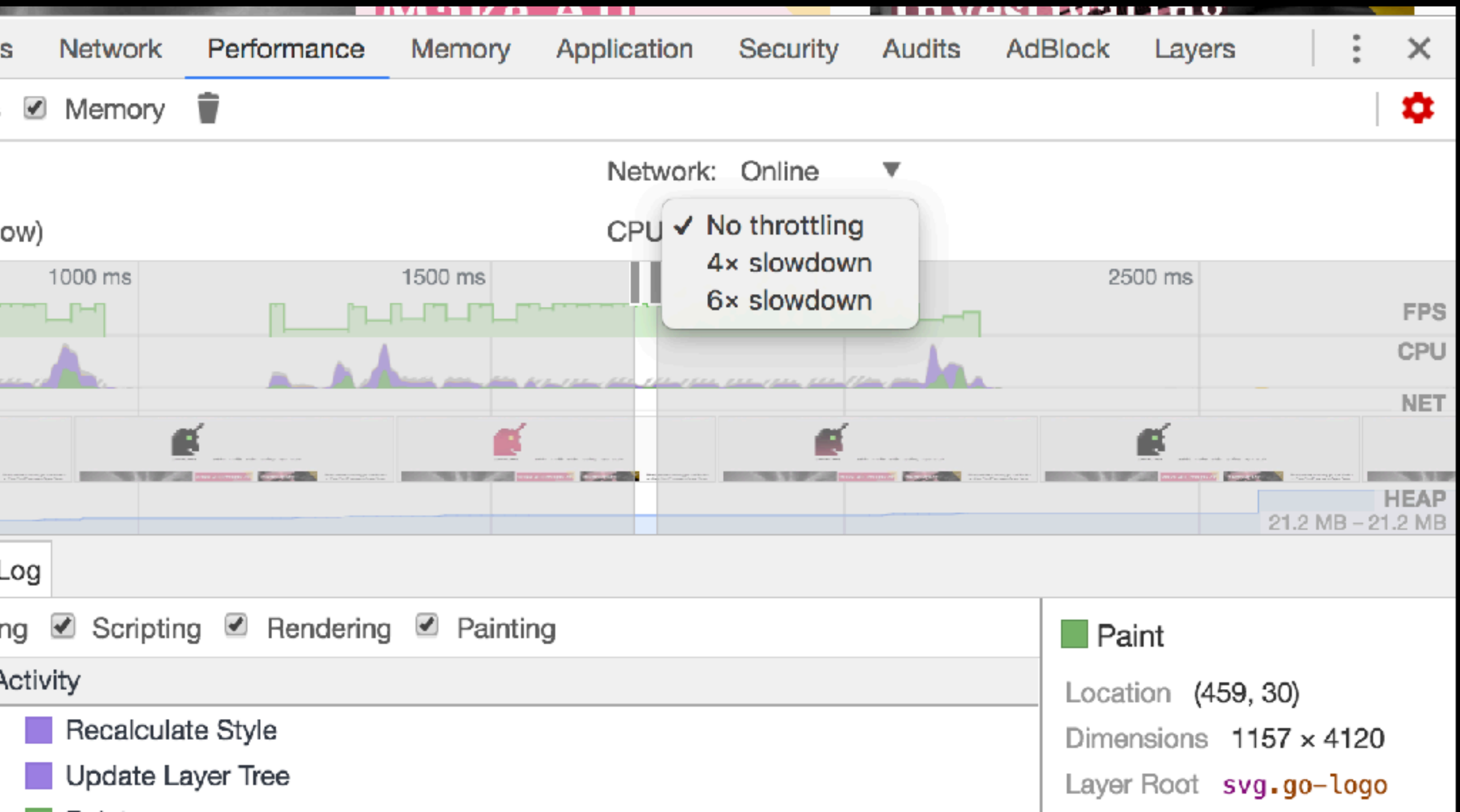
Slow scroll rects Paints

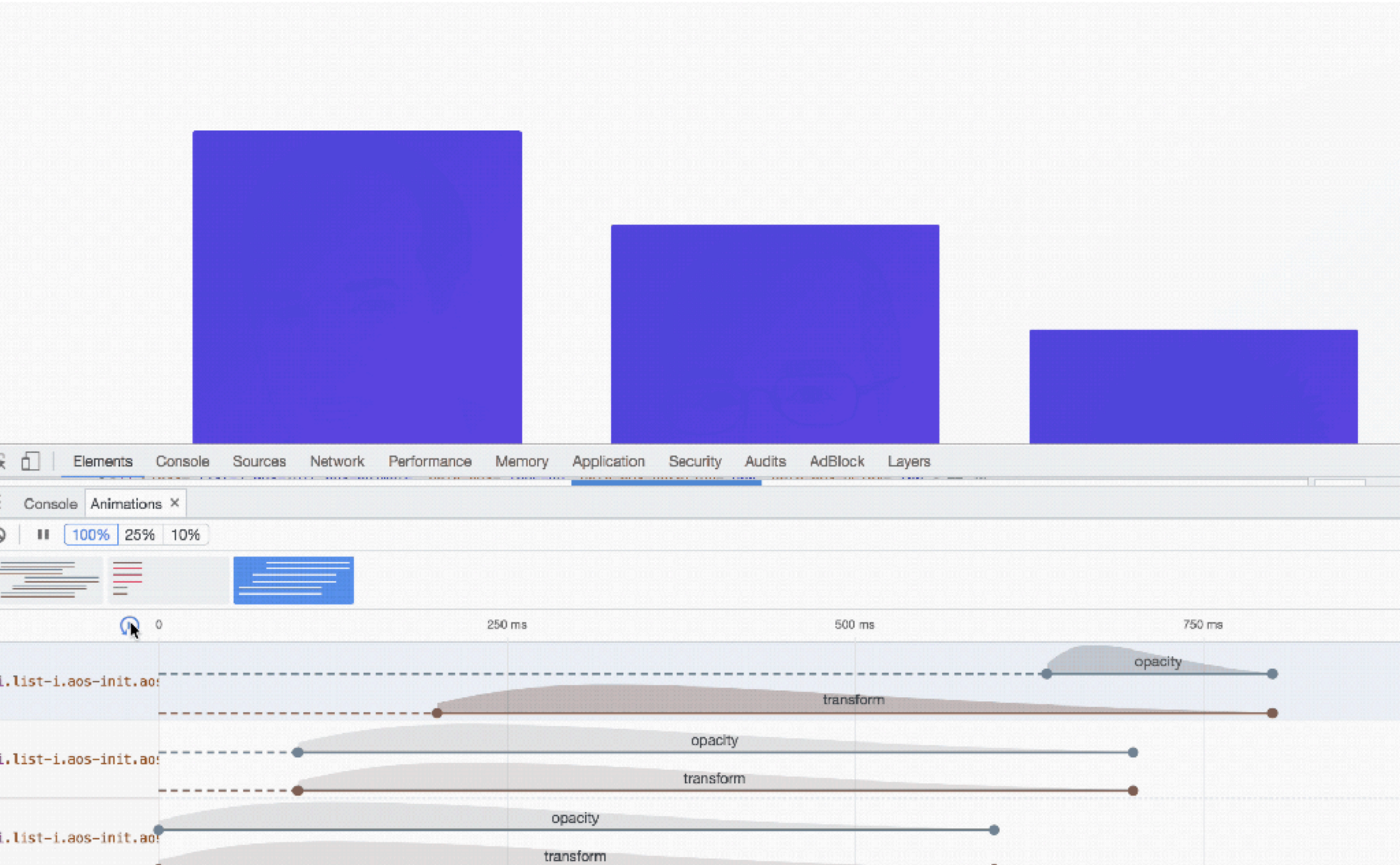


Details

Size	1130 × 1461 (at 62,2546)
Compositing Reasons	layerForSquashingContents
Memory estimate	6.3 MB
Paint count	3561
Slow scroll regions	

CPU Throttling




Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
AdBlock
Layers
Console
Animations
100%
25%
10%
0
250 ms
500 ms
750 ms
opacity
transform
opacity
transform
opacity
transform

Lessons

- Do NOT optimize blindly
- Know what is happening all the time
- Know your tools

Tools

THREE.JS

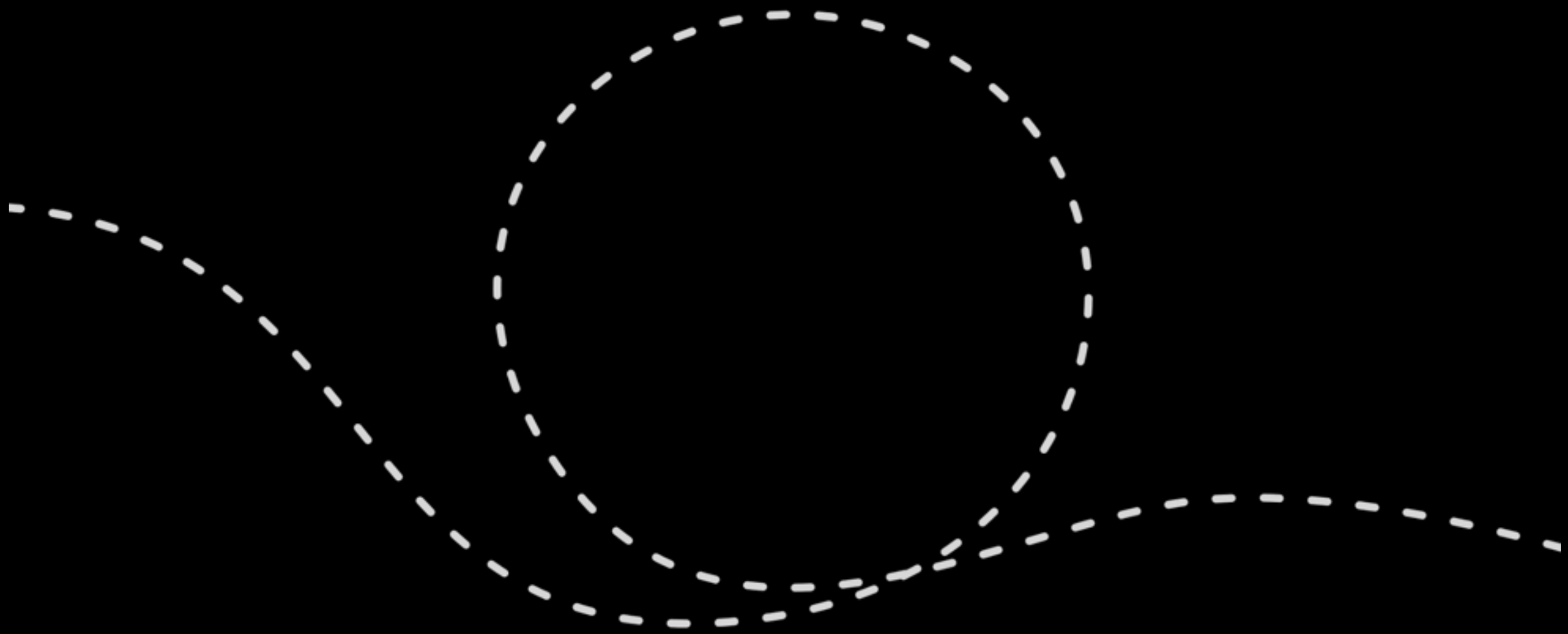
PIXI.JS

SNAP.SVG

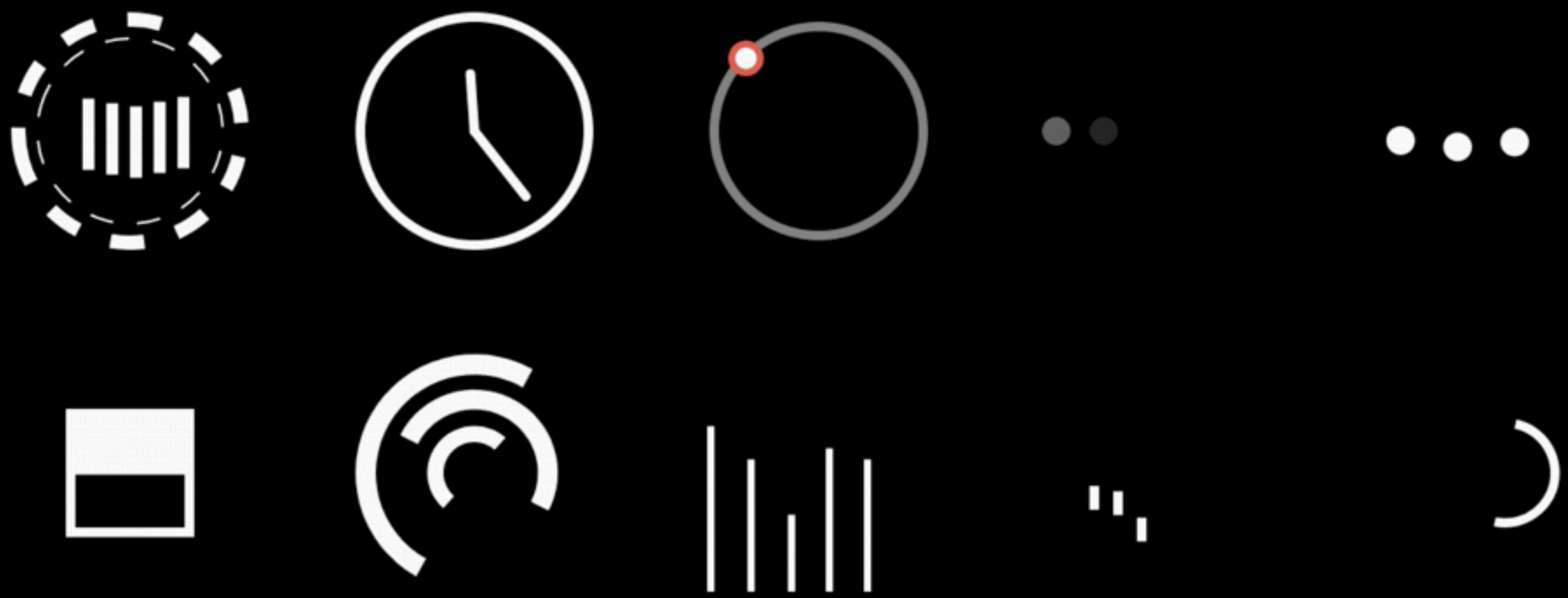
Cinema 4D

Arsenal,
or what is possible

SVG Animation



Typical SVG animation

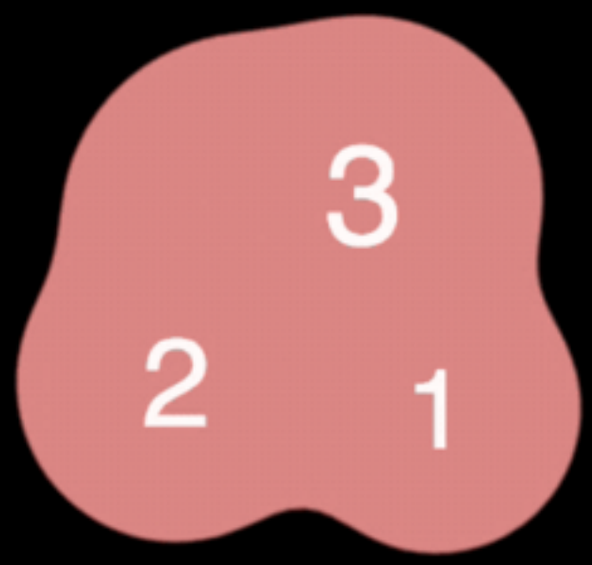


Mostly because

- Easier to implement
- Static objects
- Morphing
- Already painted by browser

SVG Filters

- Outstanding effects
- Low performance



https://css-tricks.com/gooey-effect/

CSS Animations

- No JS
- We have DOM already
- transform + opacity only, usually

Canvas 2D

- Old school, straightforward API
- Lots of ready to use stuff
- CPU, so be careful (use throttling)
- Working with pixels, painting by yourself
- Mobile limitations

Paper.js — Examples

paperjs.org/examples/

Paper.js

Examples


Boolean Operations


Candy Crash

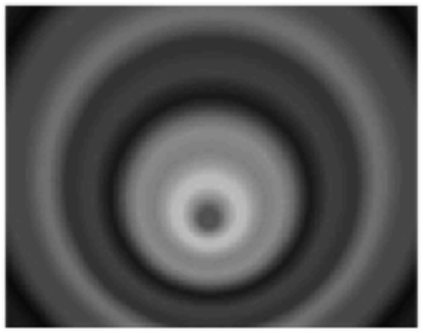

Satie Liked To Draw


Chain

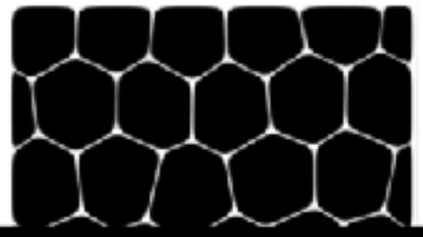

Tadpoles


Nyan Rainbow


Rounded Rectangles


Radial Rainbows


Meta Balls



Paper.js — Bouncing Balls

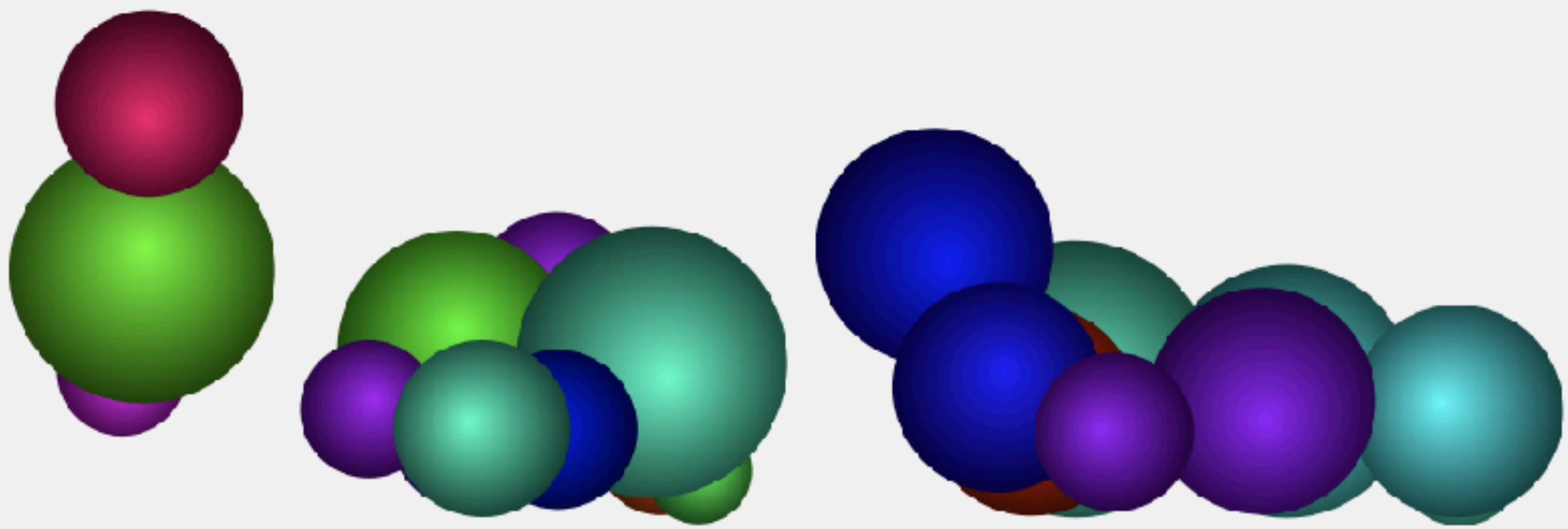
paperjs.org/examples/bouncing-balls/

Paper.js

About
Features

Examples
- Boolean Operations
- Candy Crash
- Satie Liked To Draw
- Chain
- Tadpoles
- Nyan Rainbow
- Rounded Rectangles
- Radial Rainbows
- Meta Balls
- Voronoi
- Future Splash
- Smoothing
- Spiral Raster
- Division Raster
- Q-bertify
- Path Intersections
- Path Simplification

Click, drag and release to add balls.



Elements Console Sources Network Performance Memory Application Security Audits AdBlock Layers

Screenshots Memory

Disable JavaScript samples

Enable advanced paint instrumentation (slow)

Network: Online

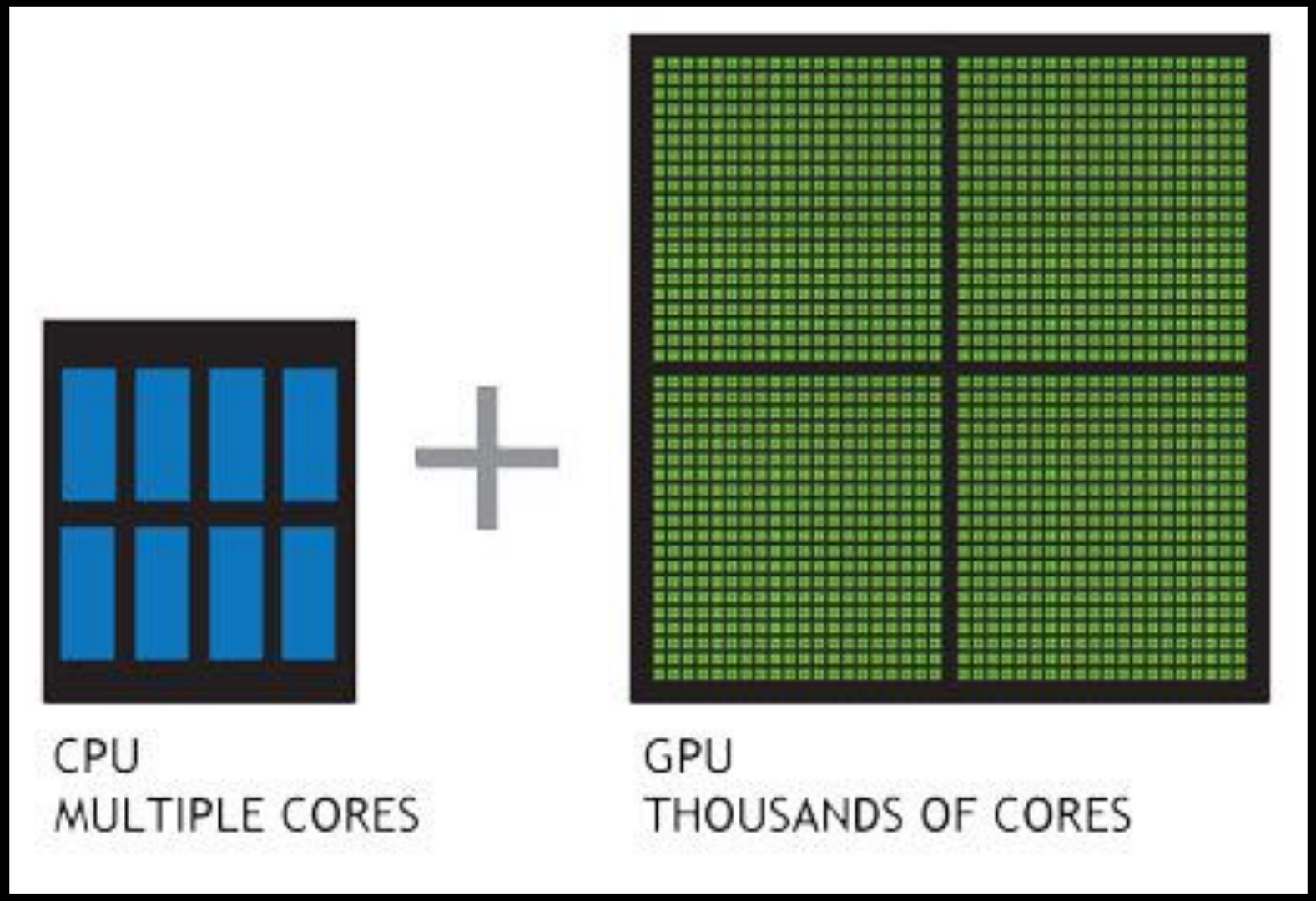
CPU: 6x slowdown

Click the record button ● or hit ⌘ E to start a new recording.

WebGL

- Runs on GPU, mostly
- Complicated API
- You need to draw yourself

GPU

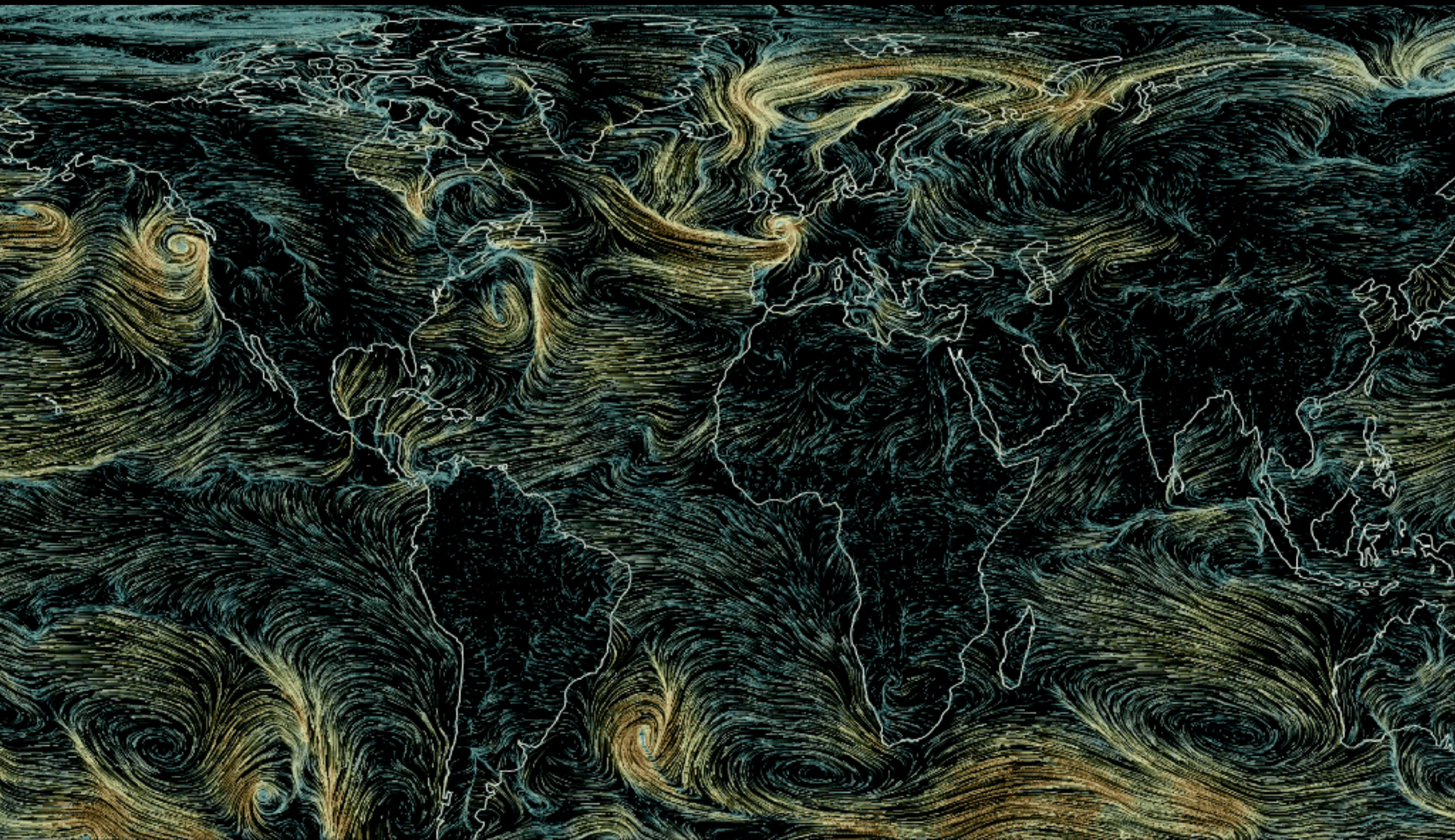


What’s that good for?

- You can render millions of objects
- You can work directly with GPU (via shaders)
- Your CPU is almost idle, if you do it right

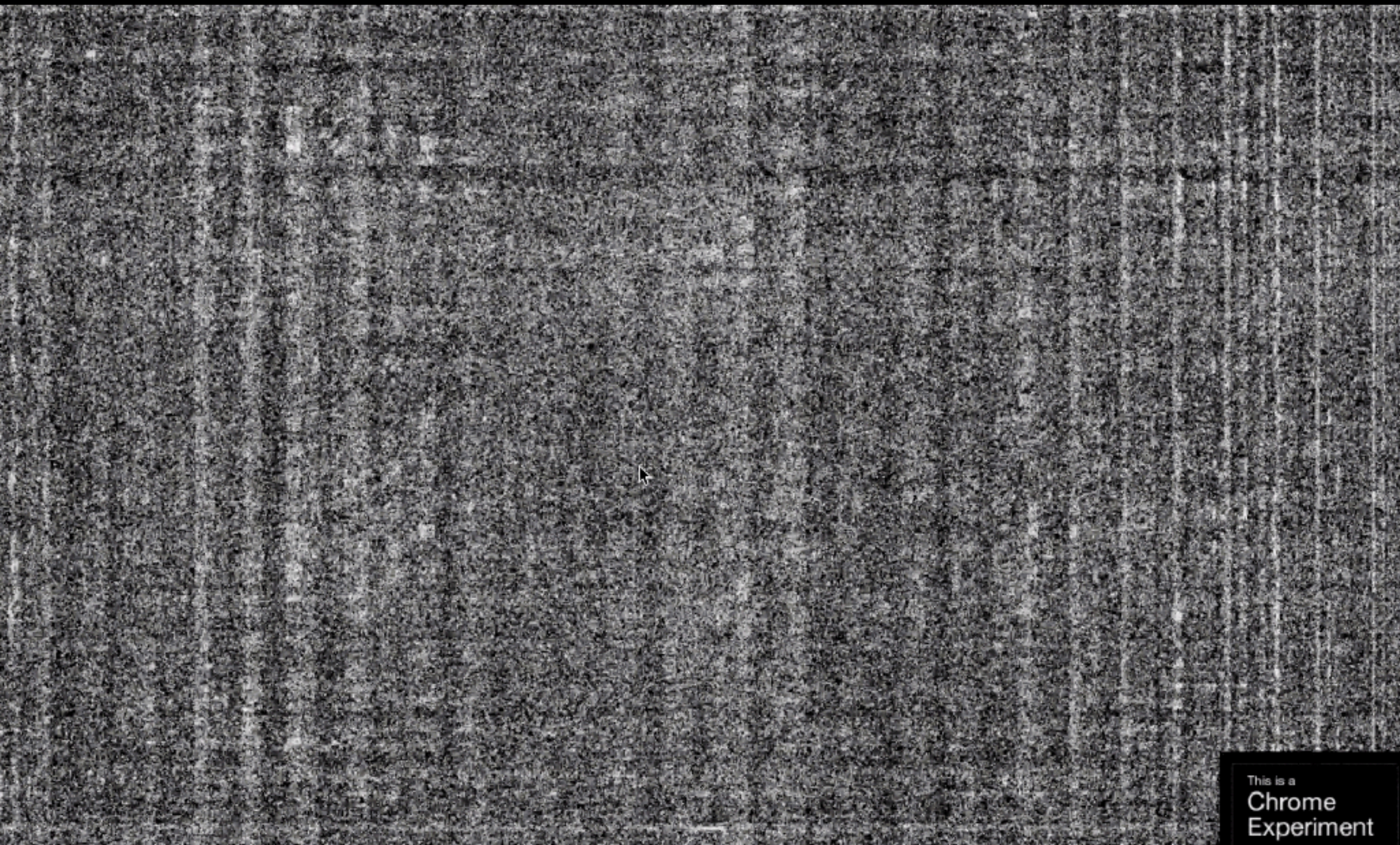
Wind map

https://blog.mapbox.com/how-i-built-a-wind-map-with-webgl-b63022b5537f



What’s that shader thing?

Million of particles



Shaders

- Not javascript
- Key to GPU calculations
- If you are good at math, you are a god of pixels


```
void main() {
    gl_FragColor = vec4(0.,0.,0.,1.0);
}
```

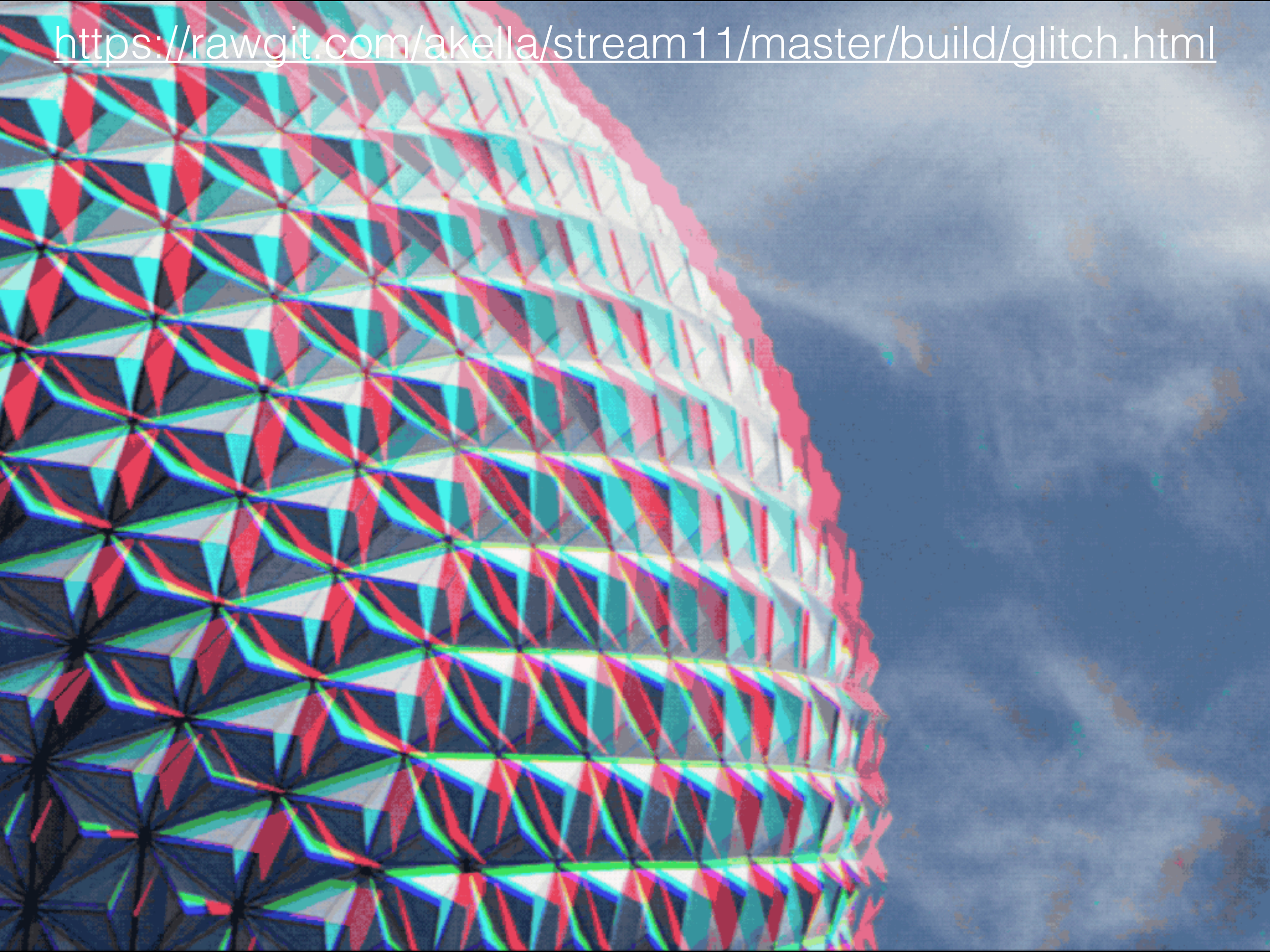

```
uniform vec2 u_resolution;
void main() {
    vec3 color = vec3(0.);
    vec2 st = gl_FragCoord.xy/
        u_resolution;
    color += step(.5+cos(st.y*PI)*.25,
                  st.x);
    gl_FragColor = vec4(color,1.);
}
```

wave.frag

```
precision mediump float;
uniform sampler2D uSampler;
uniform float time;
varying vec2 vTextureCoord;

void main(void) {
    vec2 uv = vTextureCoord;

    float distort = sin(uv.y * 100.0 + time)*0.001;

    gl_FragColor = texture2D(uSampler, vec2(uv.x + distort,uv.y));
}
```

https://codepen.io/akella/pen/gRzjej

https://rawgit.com/akella/stream11/master/build/glitch.html



untitled UNREGISTERED

untitled

```gl_fragment_shader
precision highp float;

uniform sampler2D u_particles;
uniform sampler2D u_wind;
uniform vec2 u_wind_res;
uniform vec2 u_wind_min;
uniform vec2 u_wind_max;
uniform float u_rand_seed;
uniform float u_speed_factor;
uniform float u_drop_rate;
uniform float u_drop_rate_bump;

varying vec2 v_tex_pos;

// pseudo-random generator
const vec3 rand_constants = vec3(12.9898, 78.233, 4375.85453);
float rand(const vec2 co) {
    float t = dot(rand_constants.xy, co);
    return fract(sin(t) * (rand_constants.z + t));
}

// wind speed lookup; use manual bilinear filtering based on 4 adjacent pixels for smooth interpolation
vec2 lookup_wind(const vec2 uv) {
    // return texture2D(u_wind, uv).rg; // lower-res hardware filtering
    vec2 px = 1.0 / u_wind_res;
    vec2 vc = (floor(uv * u_wind_res)) * px;
    vec2 f = fract(uv * u_wind_res);
    vec2 tl = texture2D(u_wind, vc).rg;
    vec2 tr = texture2D(u_wind, vc + vec2(px.x, 0)).rg;
    vec2 bl = texture2D(u_wind, vc + vec2(0, px.y)).rg;
    vec2 br = texture2D(u_wind, vc + px).rg;
    return mix(mix(tl, tr, f.x), mix(bl, br, f.x), f.y);
}

void main() {
    vec4 color = texture1D(u_particles, v_tex_pos);
    vec2 pos = vec2(
        color.r / 255.0 + color.b,
        color.g / 255.0 + color.a); // decode particle position from pixel RGBA

    vec2 velocity = mix(u_wind_min, u_wind_max, lookup_wind(pos));
    float speed_t = length(velocity) / length(u_wind_max);

    // take EPSG:4236 distortion into account for calculating where the particle moved
    float distortion = cos(radians(pos.y * 180.0 - 90.0));
    vec2 offset = vec2(velocity.x / distortion, -velocity.y) * 0.0001 * u_speed_factor;

    // update particle position, wrapping around the date line
    pos = fract(1.0 + pos + offset);

    // a random seed to use for the particle drop
    vec2 seed = (pos + v_tex_pos) * u_rand_seed;

    // drop rate is a chance a particle will restart at random position, to avoid degeneration
```

Line 37, Column 21 Spaces: 4 OpenGL SL

```gl
uniform sampler2D u_particles;
uniform sampler2D u_wind;
uniform vec2 u_wind_res;
uniform vec2 u_wind_min;
uniform vec2 u_wind_max;
uniform float u_rand_seed;
uniform float u_speed_factor;
uniform float u_drop_rate;
uniform float u_drop_rate_bump;

varying vec2 v_tex_pos;

// pseudo-random generator
const vec3 rand_constants = vec3(12.9898, 78.233, 4375.85453);
float rand(const vec2 co) {
    float t = dot(rand_constants.xy, co);
    return fract(sin(t) * (rand_constants.z + t));
}

// wind speed lookup; use manual bilinear ¤_¤ filtering based on 4 adjacent pixels for smooth interpolation
vec2 lookup_wind(const vec2 uv) {
    // return texture2D(u_wind, uv).rg; // lower-res hardware filtering
    vec2 px = 1.0 / u_wind_res;
    vec2 vc = (floor(uv * u_wind_res)) * px;
    vec2 f = fract(uv * u_wind_res);
    vec2 tl = texture2D(u_wind, vc).rg;
    vec2 tr = texture2D(u_wind, vc + vec2(px.x, 0)).rg;
    vec2 bl = texture2D(u_wind, vc + vec2(0, px.y)).rg;
    vec2 br = texture2D(u_wind, vc + px).rg;
    return mix(mix(tl, tr, f.x), mix(bl, br, f.x), f.y);
}

void main() {
    vec4 color = texture2D(u_particles, v_tex_pos);
    vec2 pos = vec2(
        color.r / 255.0 + color.b,
        color.g / 255.0 + color.a); // decode particle position from pixel RGBA

    vec2 velocity = mix(u_wind_min, u_wind_max, lookup_wind(pos));
    float speed_t = length(velocity) / length(u_wind_max);

    // take EPSG:4236 distortion into account for calculating where the particle moved
    float distortion = cos(radians(pos.y * 180.0 - 90.0));
```

```
uniform float u_rand_seed;
uniform float u_speed_factor;
uniform float u_drop_rate;
uniform float u_drop_rate_bump;

varying vec2 v_tex_pos;

// pseudo-random generator
const vec3 rand_constants = vec3(12.9898, 78.233, 4375.85453);
float rand(const vec2 co) {
    float t = dot(rand_constants.xy, co);
    return fract(sin(t) * (rand_constants.z + t));
}

// wind speed lookup; use manual bilinear ʊ_ʊ filtering based on 4 adjacent pixels for smooth interpo
vec2 lookup_wind(const vec2 uv) {
    // return texture2D(u_wind, uv).rg; // lower-res hardware filtering
    vec2 px = 1.0 / u_wind_res;
    vec2 vc = (floor(uv * u_wind_res)) * px;
    vec2 f = fract(uv * u_wind_res);
    vec2 tl = texture2D(u_wind, vc).rg;
    vec2 tr = texture2D(u_wind, vc + vec2(px.x, 0)).rg;
    vec2 bl = texture2D(u_wind, vc + vec2(0, px.y)).rg;
    vec2 br = texture2D(u_wind, vc + px).rg;
    return mix(mix(tl, tr, f.x), mix(bl, br, f.x), f.y);
}

void main() {
    vec4 color = texture2D(u_particles, v_tex_pos);
    vec2 pos = vec2(
        color.r / 255.0 + color.b
```

```
iform float u_drop_rate;
iform float u_drop_rate_bump;

rying vec2 v_tex_pos;

 pseudo-random generator
nst vec3 rand_constants = vec3(12.9898, 78.233, 4375.85453);
oat rand(const vec2 co) {
  float t = dot(rand_constants.xy, co);
  return fract(sin(t) * (rand_constants.z + t));


 wind speed lookup; use manual bilinear ʊ_ʊ filtering based on 4 adjacent pixels for s
c2 lookup_wind(const vec2 uv) {
  // return texture2D(u_wind, uv).rg; // lower-res hardware filtering
  vec2 px = 1.0 / u_wind_res;
  vec2 vc = (floor(uv * u_wind_res)) * px;
  vec2 f = fract(uv * u_wind_res);
  vec2 tl = texture2D(u_wind, vc).rg;
  vec2 tr = texture2D(u_wind, vc + vec2(px.x, 0)).rg;
  vec2 bl = texture2D(u_wind, vc + vec2(0, px.y)).rg;
  vec2 br = texture2D(u_wind, vc + px).rg;
  return mix(mix(tl, tr, f.x), mix(bl, br, f.x), f.y);
```

```
nts = vec3(12.9898, 78.233, 4375.85453);
co) {
l_constants.xy, co);
) * (rand_constants.z + t));


use manual bilinear o_o filtering based on 4 a
: vec2 uv) {
2D(u_wind, uv).rg; // lower-res hardware filter
_wind_res;
uv * u_wind_res)) * px;
* u_wind_res);
2D(u_wind, vc).rg;
```

```
bilinear ⊙ ⊙ filterin

v).rg; // lower-res h
```

$$0.0$$

LEARN
MATH

LEARN
MATH

LEARN MATH

Also

- thebookofshaders.com/
- pixelspiritdeck.com/

What should we use?!

Three Little Pigs Hostel Berlin, Stresem

Eberswalder Straße, 10435 Berlin, Germ

Leave now

OPTIONS

Send directions to your phone

7:50 AM–8:12 AM 22 min

S1 / S2 / S25 > Tram 12 / M10

7:53 AM from S Anhalter Bahnhof (Berlin) · 1 min late

3 min every 3 min

DETAILS

7:48 AM–8:12 AM 24 min

BUS M29 > U2

7:51 AM–8:16 AM 25 min

BUS M41 > > U2

7:54 AM–8:20 AM 26 min

S25 > Tram M1

Some schools have a liking for extra-long swords. From my point of view these are weak schools.

This is because they do not appreciate the principle of cutting the enemy by any means.

Real life stories

#1

SVG

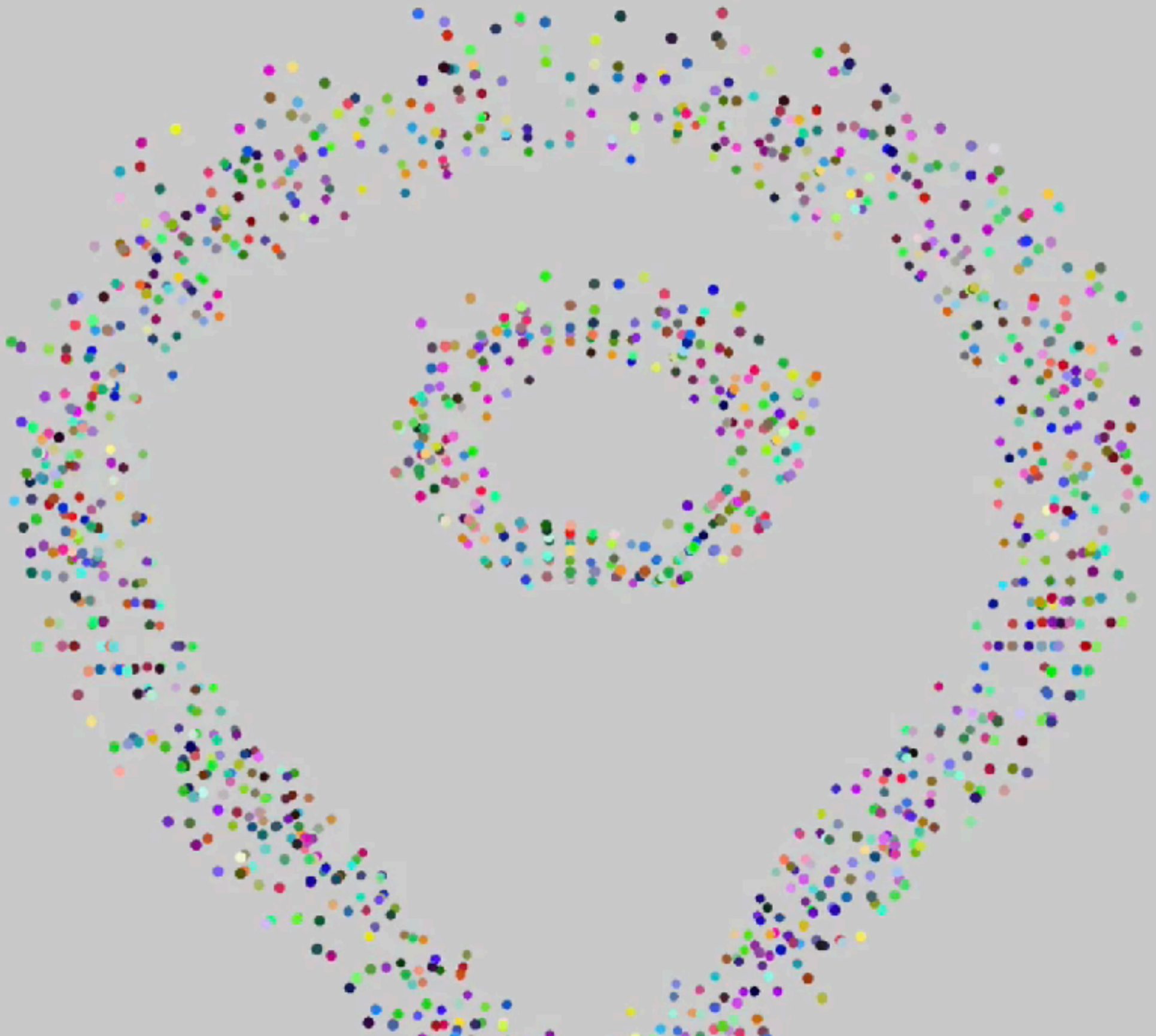
+

Canvas2D

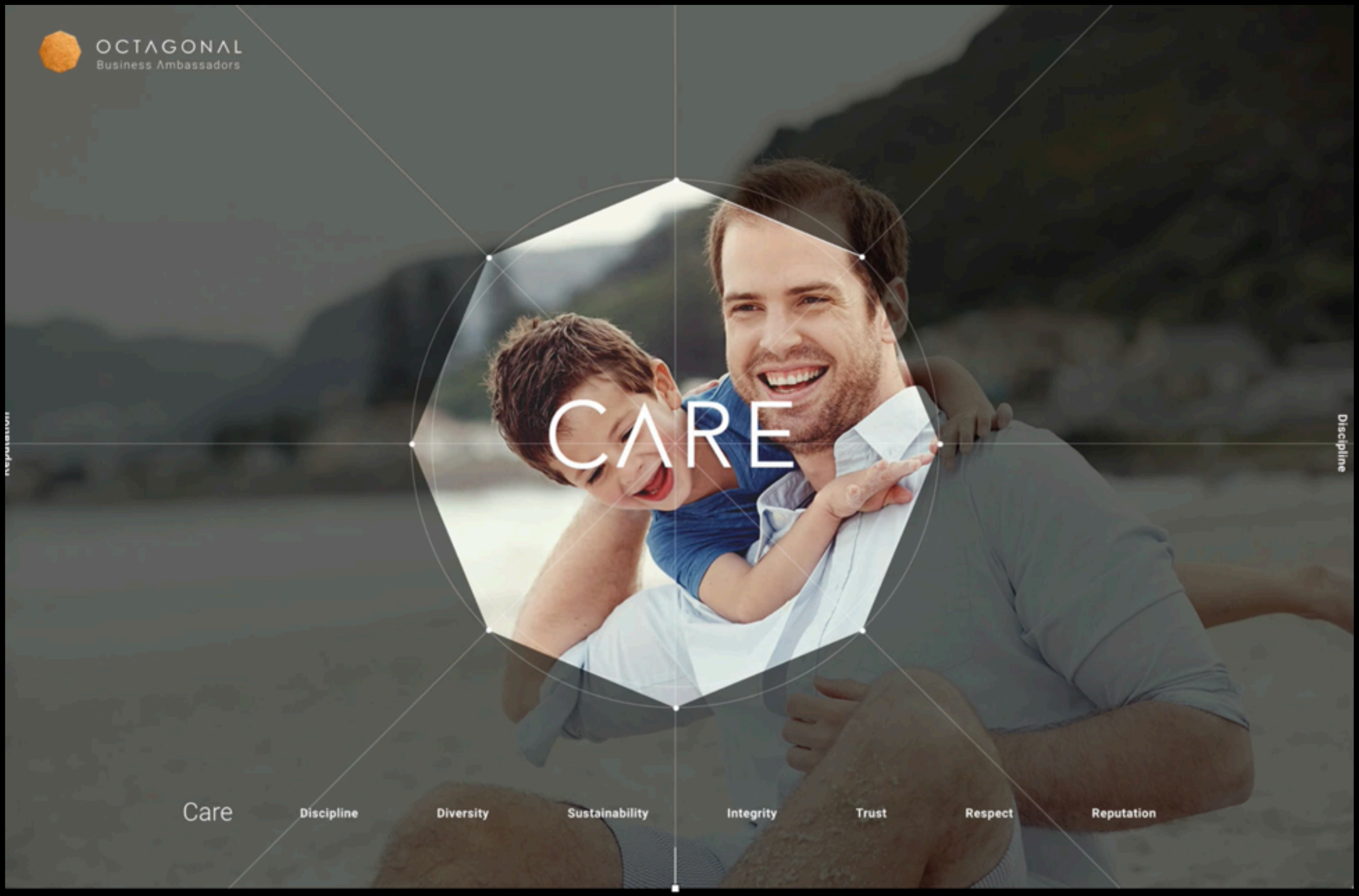
+

WebGL

https://www.youtube.com/watch?v=Q9BXGh9sdZw
https://cdn.rawgit.com/akella/dots-animation/b9abad87/index.html



#2
Canvas2D
+
SVG
+
CSS Blend Mode
+
SVG Mask


OCTAGONAL
Business Ambassadors
CARE
Discipline
Care
Discipline
Diversity
Sustainability
Integrity
Trust
Respect
Reputation

What i learned

sometimes your crazy
animation is just
a VIDEO

30fps consistent

is better than

60fps jaggy

stop RAF when
nothing happens

use just one RAF when possible

https://jsperf.com/single-raf-draw-calls-vs-multiple-raf-draw-calls

Testing in Chrome 60.0.3112 / Mac OS X 10.12.6

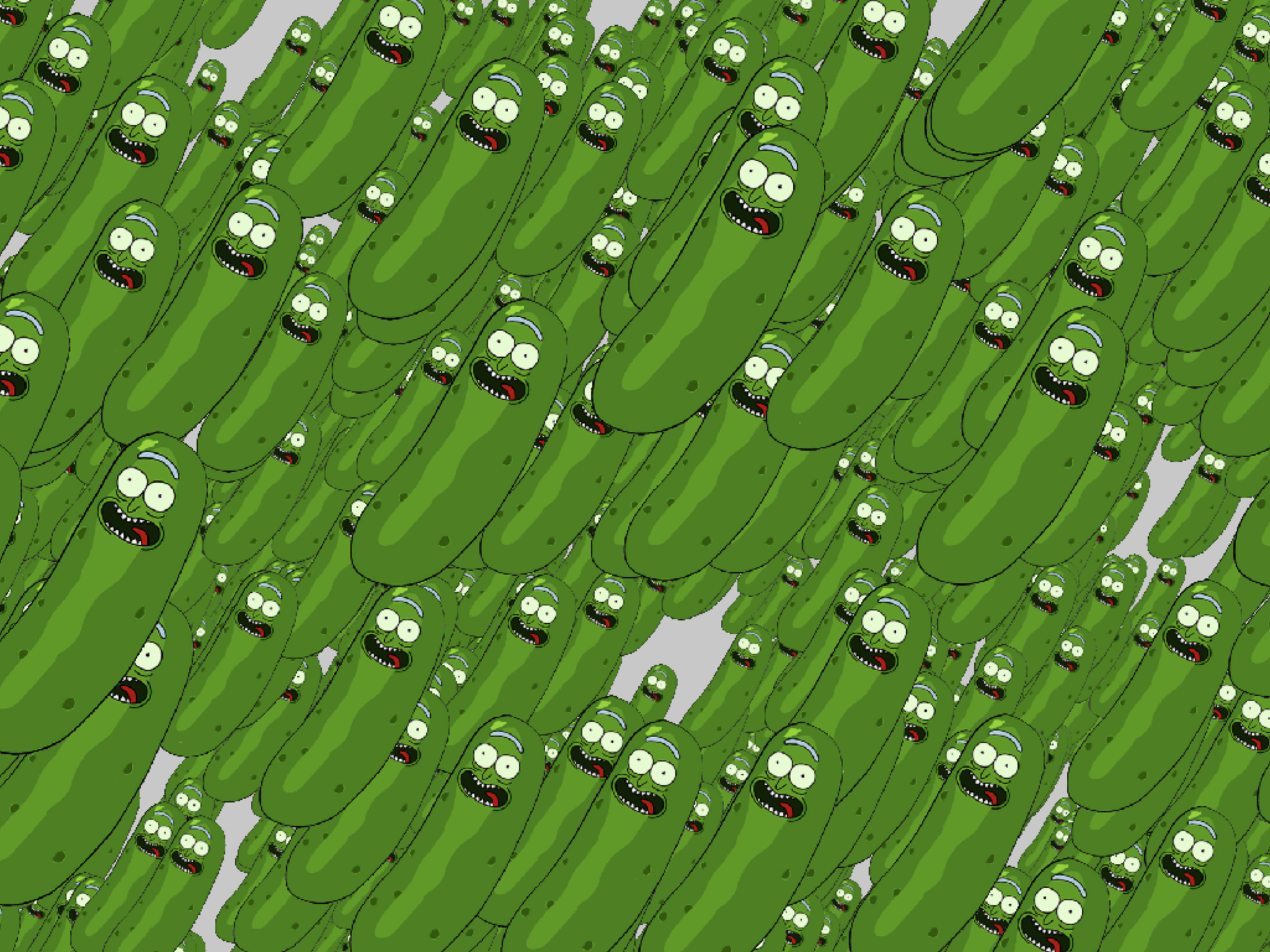
	Test	Ops/sec
Single RAF	`requestAnimationFrame(render, canvas);`	1,556,284 ±10.42% fastest

Multiple RAF	`requestAnimationFrame(render1, canvas);`
	`requestAnimationFrame(render2, canvas);`
	`requestAnimationFrame(render3, canvas);`
		583,504
		±5.57%
		61% slower

you need to love this

pickle rick tribute

http://cssing.org.ua/examples/picklerick/



I hope i inspired you

- If you know how to code, you are an artist already
- Go make something beautiful

Thank you

- twitter.com/akella
- facebook.com/akella
- youtube.com/user/flintyara/live