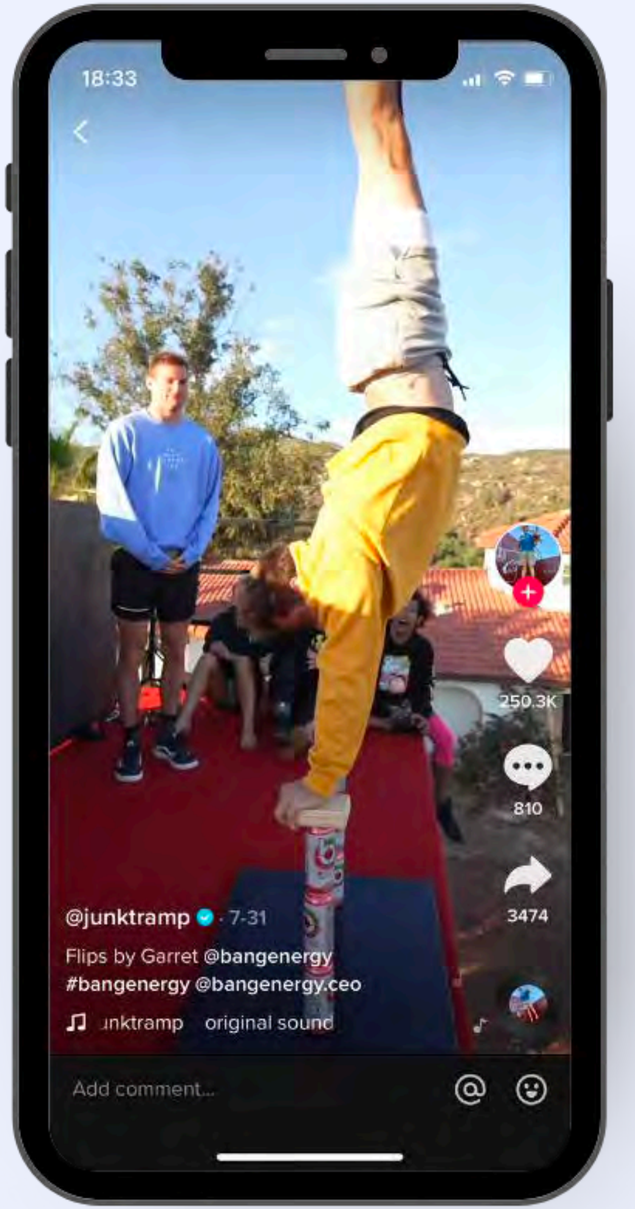
# The art of **programmatic** video with **rust**

BY THEGOOSE

# Videos

**are the most used  type of content on the internet**

more than 10 hours per week



# Technologies

are completely outdated

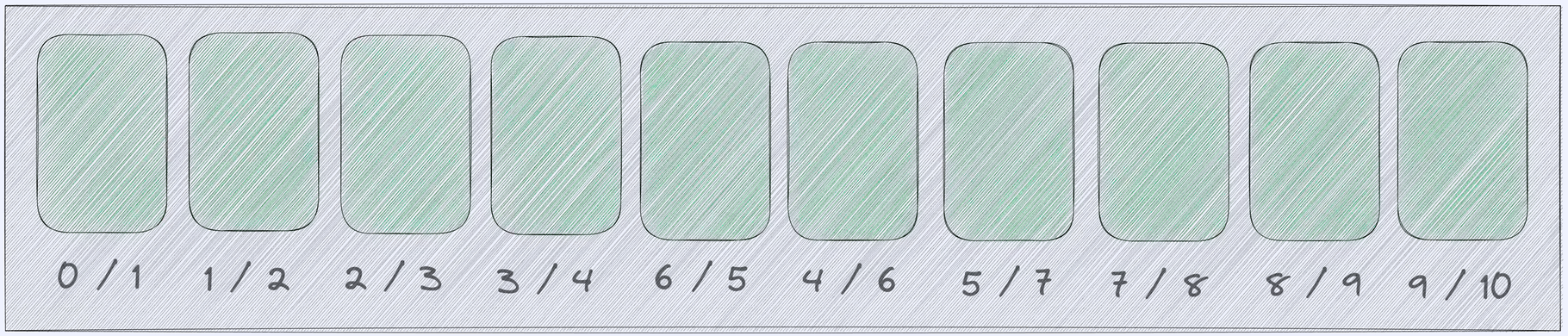
# It’s time for Rust

and always has been

# What is a video?

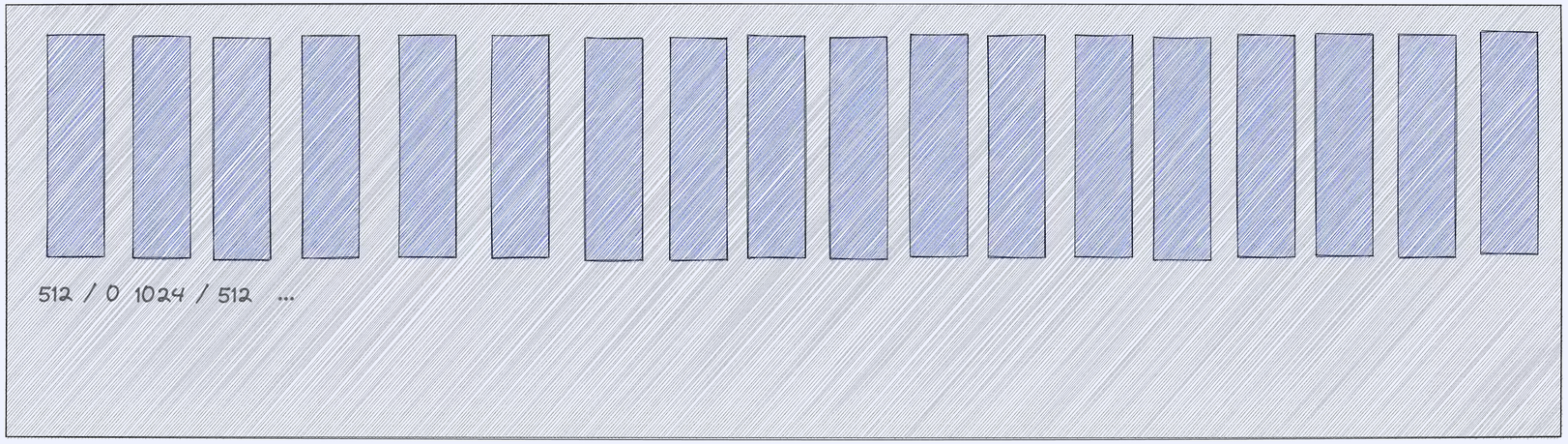
MyVideo.mp4

Images Stream

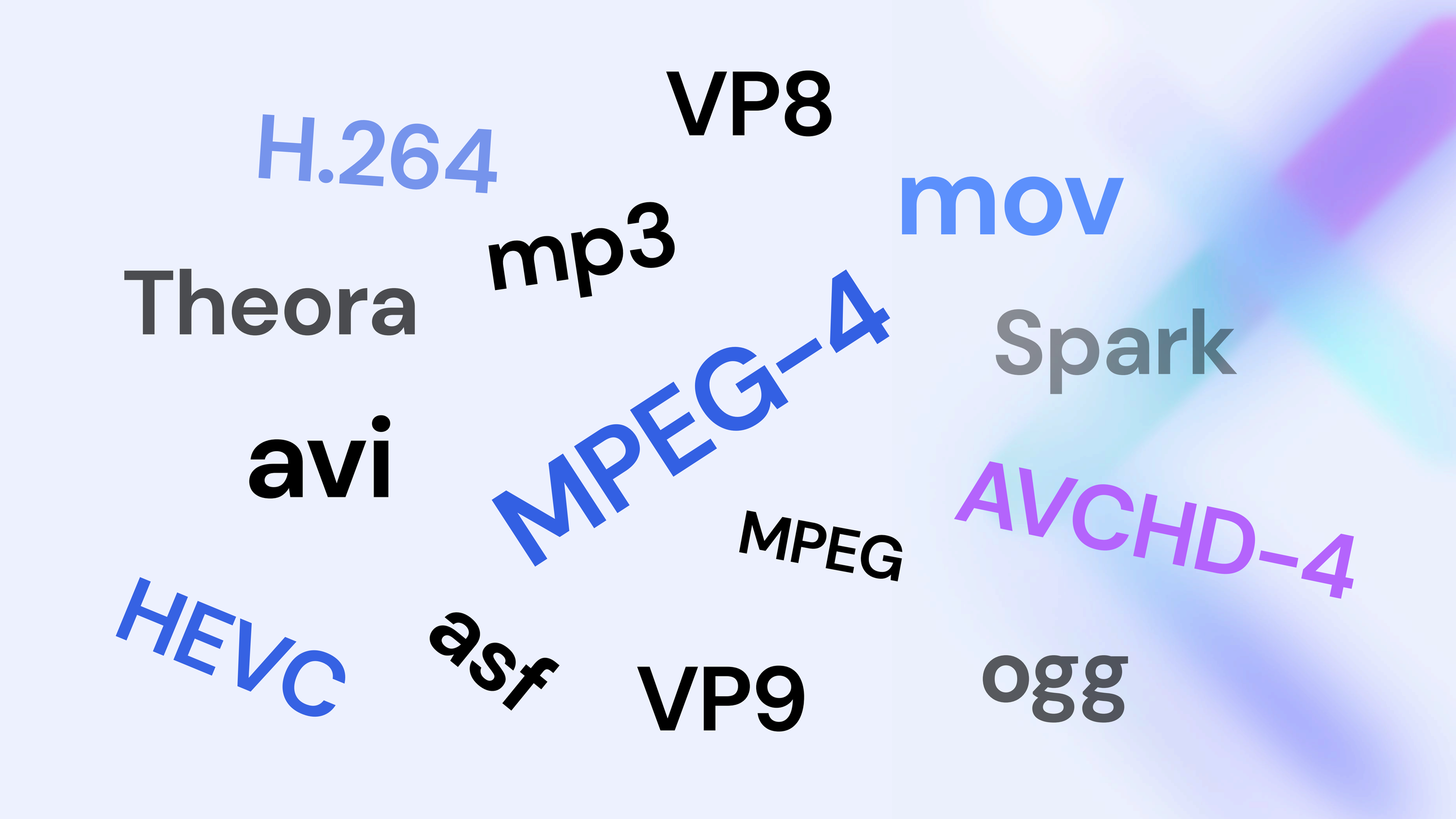



in 1/fps

Audio Stream

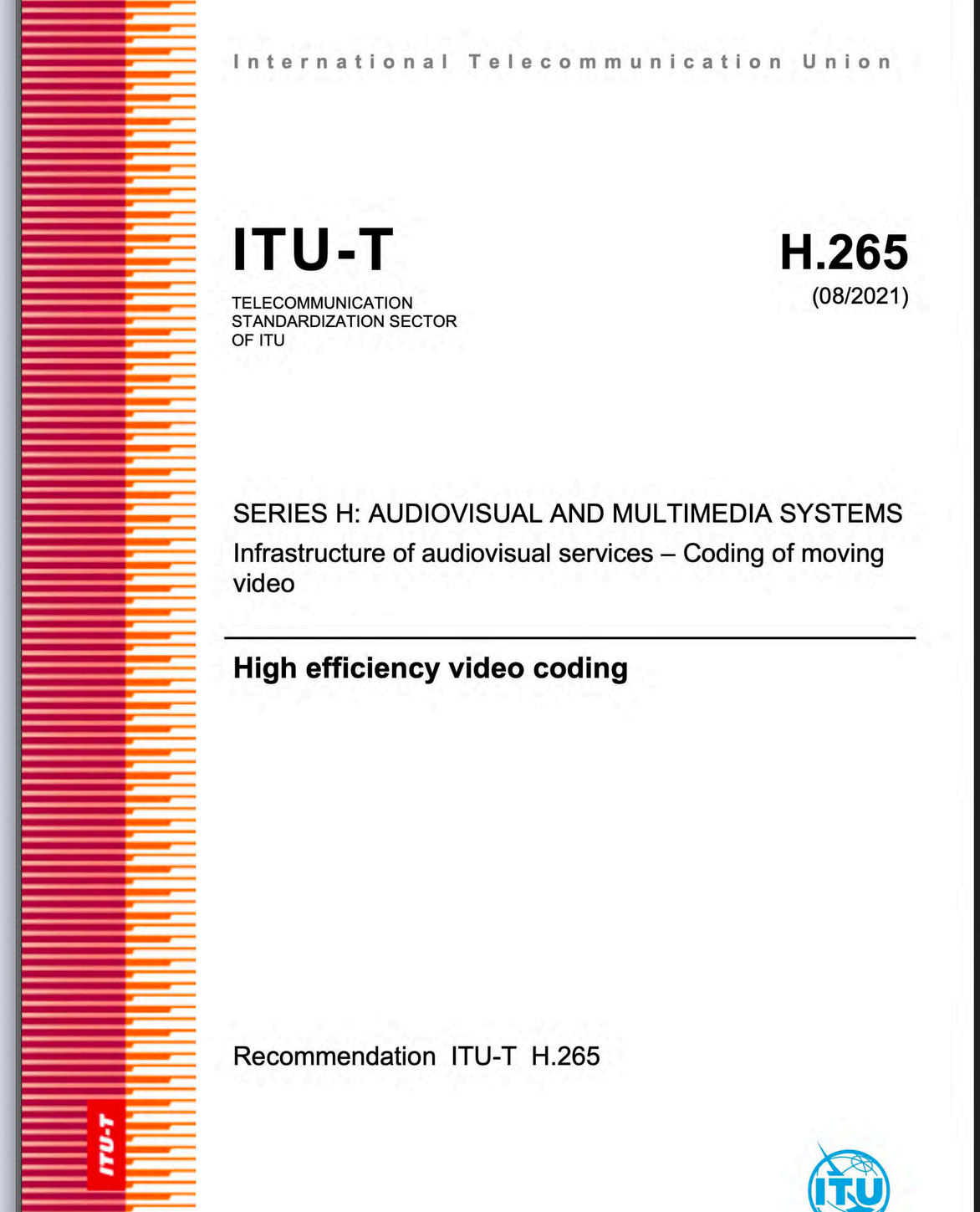



in 1/sample_rate


VP8
H.264
mov
mp3
Theora
MPEG-4
Spark
avi
AVCHD-4
MPEG
HEVC
asf
VP9
ogg

# HEVC 🤯

**specification 700 pages**



International Telecommunication Union

**ITU-T**

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

**H.265**
(08/2021)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Coding of moving video

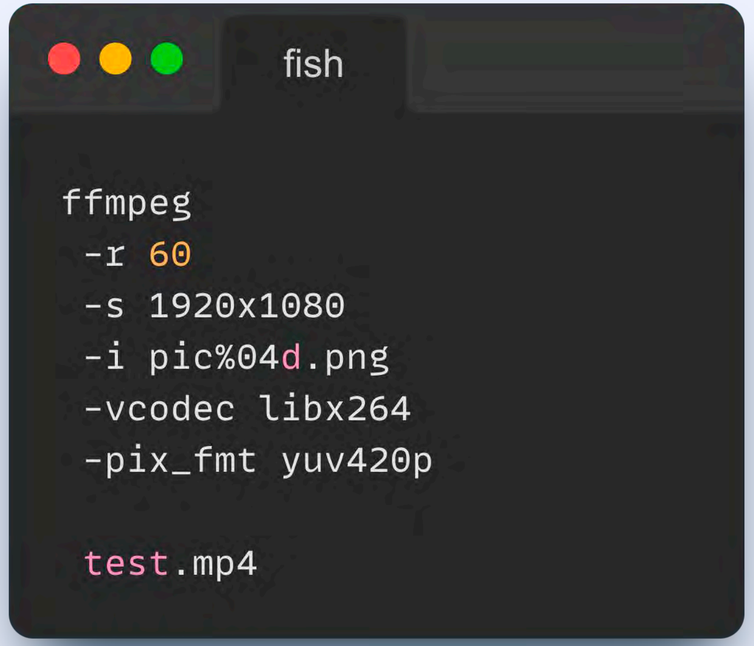
**High efficiency video coding**

Recommendation ITU-T H.265

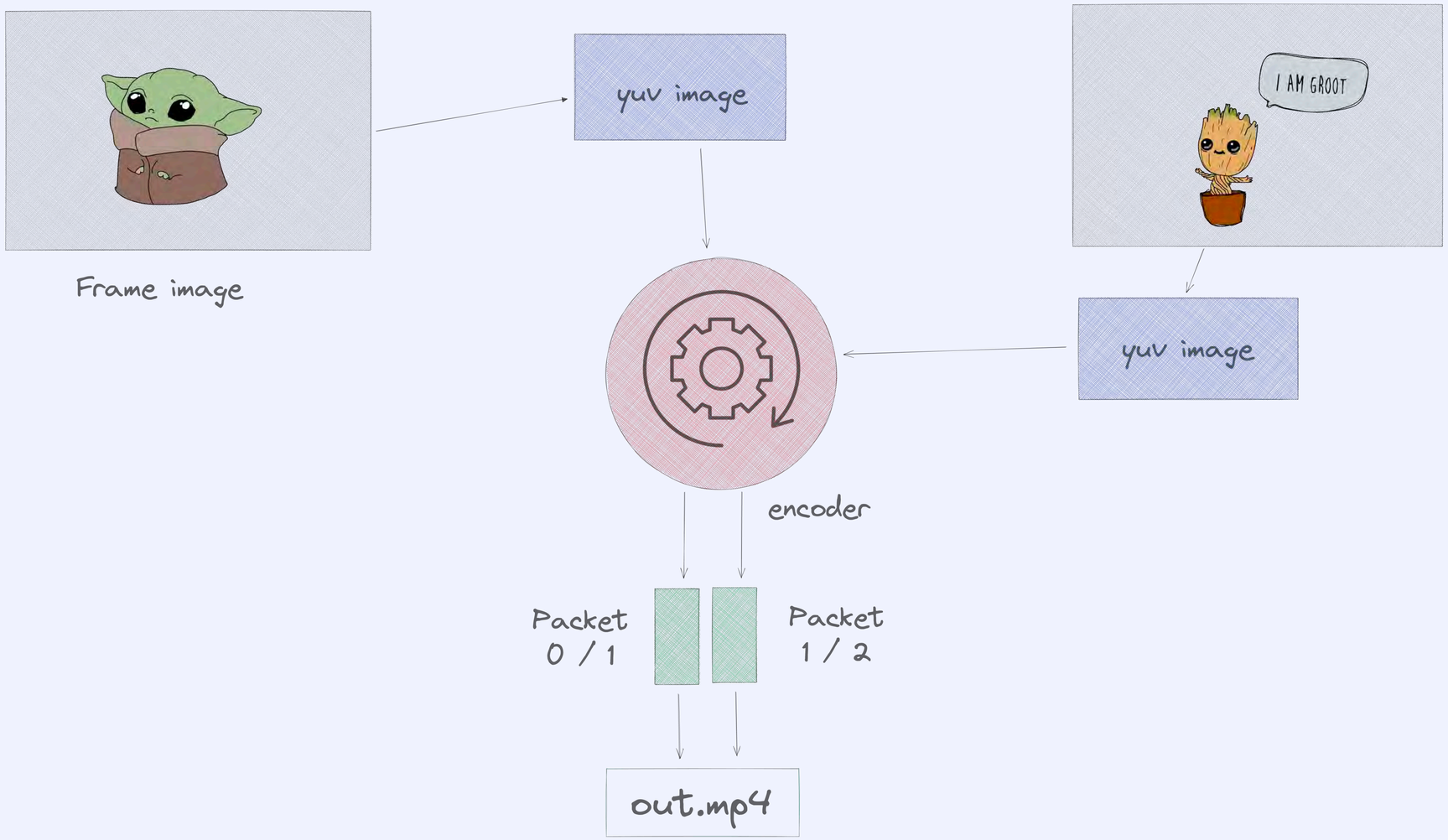
ITU-T

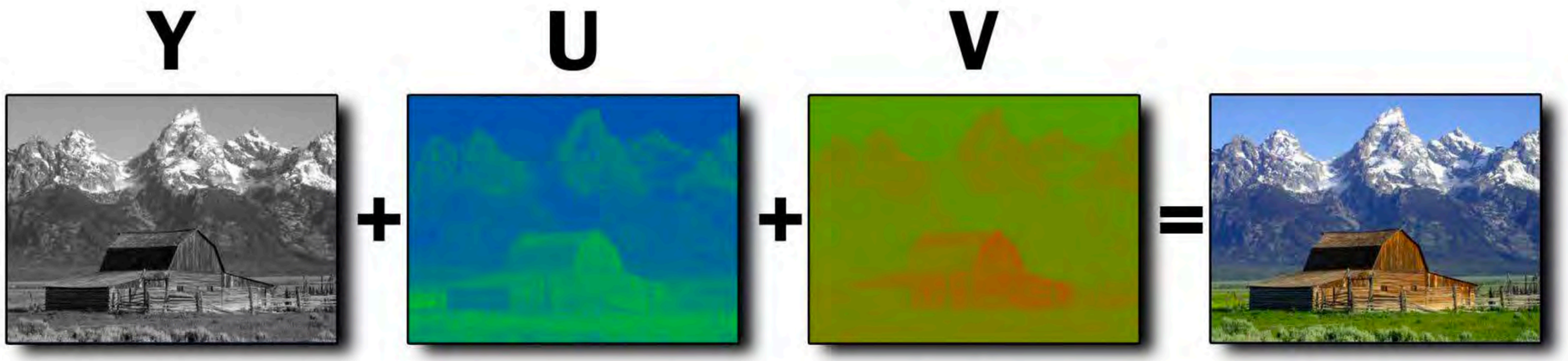

and no problems

# video generation



# Manual encoding






physical linear-space brightness

blue projection

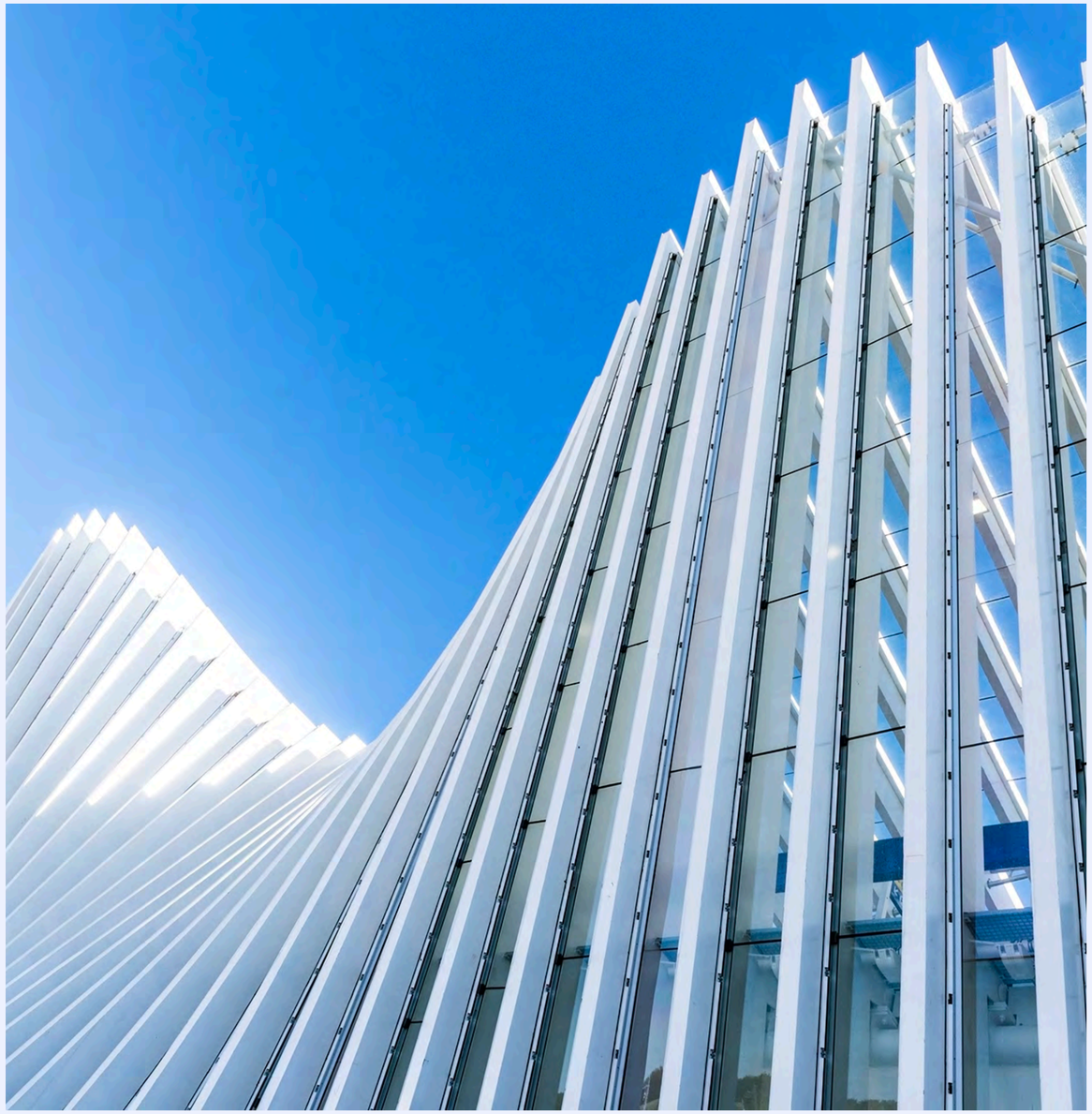
red projection


encoder.rs

```rust
let frame_size: usize = height * (*av_frame).linesize[0] as usize + width;

let y_pixels = std::slice::from_raw_parts_mut((*av_frame).data[0], frame_size);
let cb_pixels = std::slice::from_raw_parts_mut((*av_frame).data[1], frame_size / 4);
let cr_pixels = std::slice::from_raw_parts_mut((*av_frame).data[2], frame_size / 4);

for y in 0..height {
    for x in 0..width {
        let (r, g, b) = EncoderFrame::get_rgb(rgb_pixels, y * width + x);

        // use a linesize to get the correct index for the pixel as it can differ
        y_pixels[(y * (*av_frame).linesize[0] as usize + x) as usize] =
            (16 + (66 * r + 129 * g + 25 * b) >> 8) as u8;

        if y % 2 == 0 && x % 2 == 0 {
            // the bounds are 1/4 of the image size
            let x = x / 2;
            let y = y / 2;

            cb_pixels[(y * (*av_frame).linesize[1] as usize + x) as usize] =
                (128 + ((-38 * r - 74 * g + 112 * b) >> 8)) as u8;
            cr_pixels[(y * (*av_frame).linesize[2] as usize + x) as usize] =
                (128 + ((112 * r - 94 * g - 18 * b) >> 8)) as u8;
        }
    }
}
```

# Images

render them or die trying



# Browser

the most popular way to render static content

# Find similarity


dmtrKovalenko/
**odiff**

The fastest pixel-by-pixel image visual difference tool in the world.

4 Contributors · 13 Issues · 2 Discussions · 1k Stars · 64 Forks

# A format we need

Fixed

Animatable

DX ❤️‍🔥 Friendly

GPU First 🐌

Specificated

**Debuggable </>**

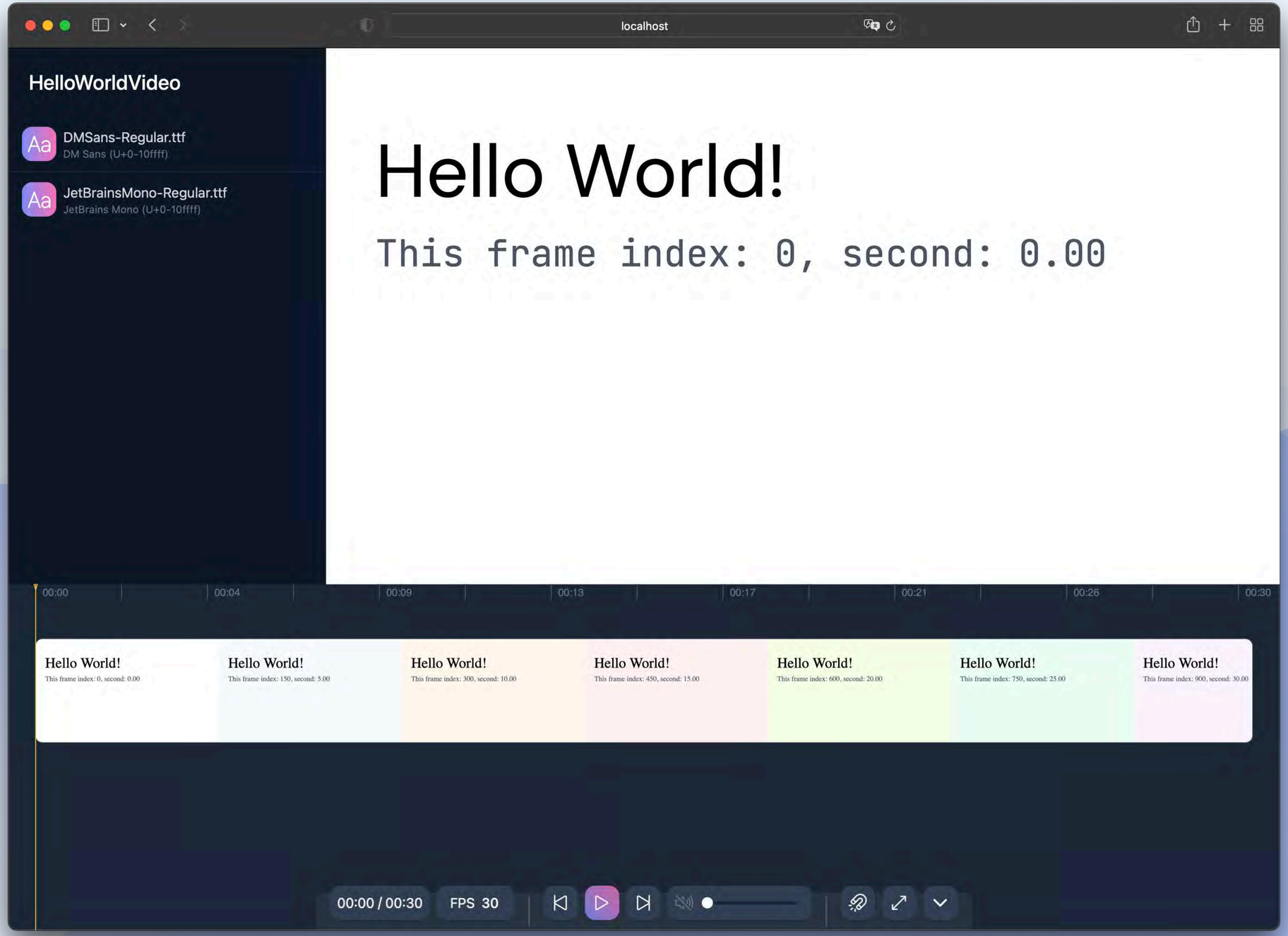
Clear

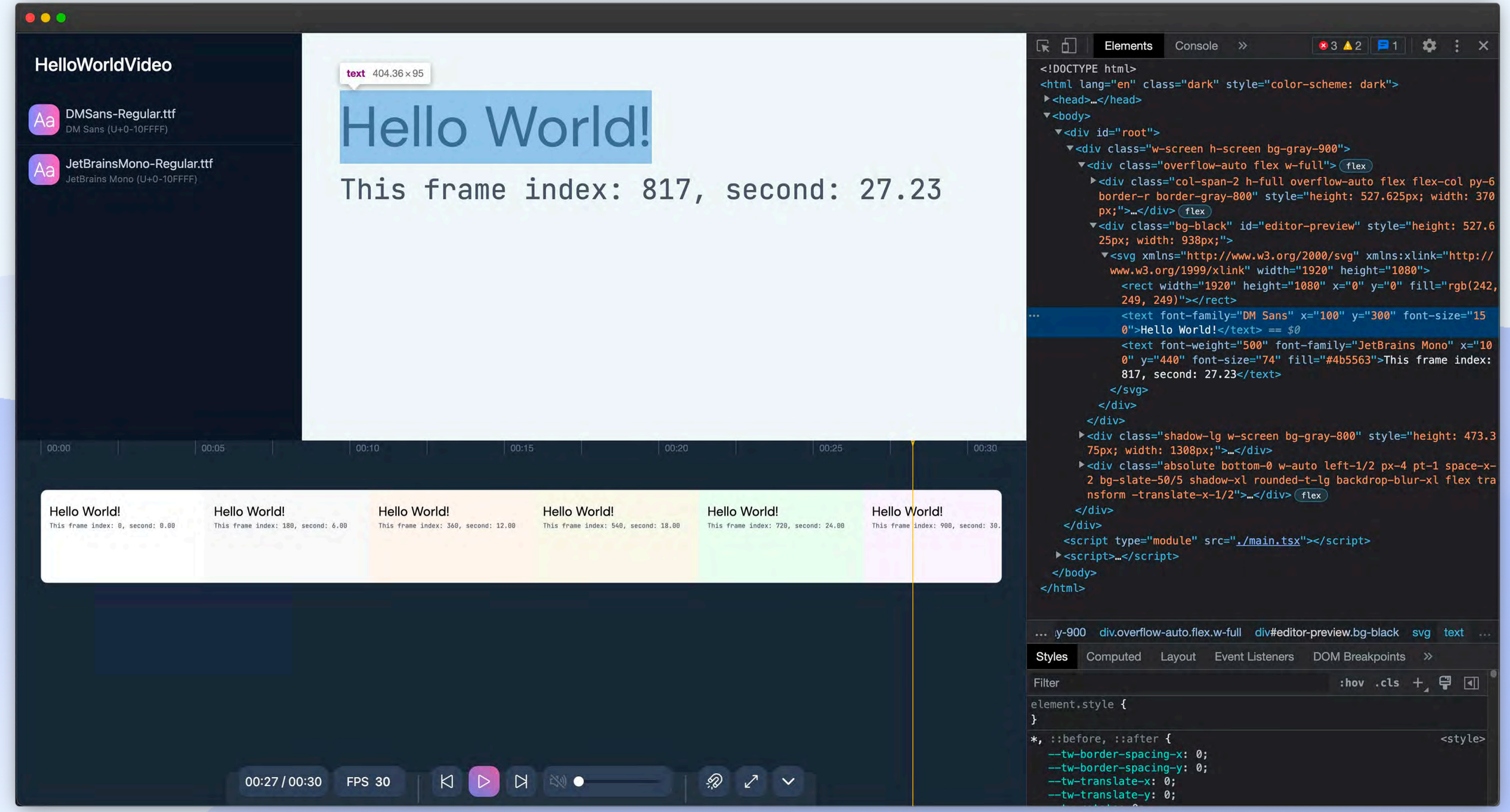
A format we need

# SVG

encoder.rs

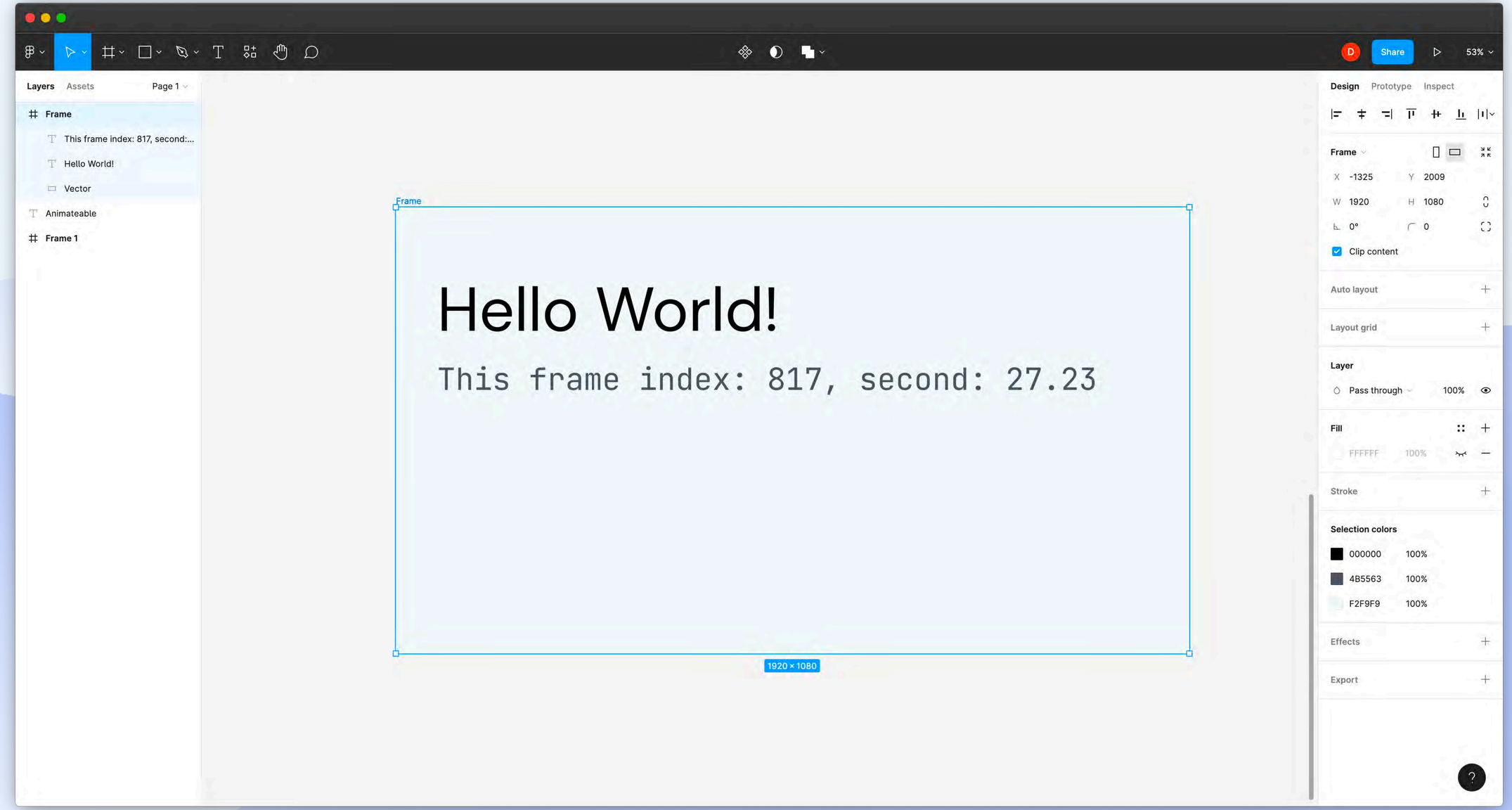
```rust
svgr!(
  <svg
    xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    width={Self::WIDTH}
    height={Self::HEIGHT}
  >
    <rect
      width={Self::WIDTH}
      height={Self::HEIGHT}
      x="0"
      y="0"
      fill={
        frame.animate(fframes::timeline!(
          on 0., val Color::hex("#fff") ⇒ Color::hex("#f8fafc"), &BACKGROUND_EASING,
          on 5., val Color::hex("#f8fafc") ⇒ Color::hex("#fff7ed"), &BACKGROUND_EASING,
          on 10., val Color::hex("#fff7ed") ⇒ Color::hex("#fef2f2"), &BACKGROUND_EASING,
          on 15., val Color::hex("#fef2f2") ⇒ Color::hex("#f7fee7"), &BACKGROUND_EASING,
          on 20., val Color::hex("#f7fee7") ⇒ Color::hex("#ecfdf5"), &BACKGROUND_EASING,
          on 25., val Color::hex("#ecfdf5") ⇒ Color::hex("#faf5ff"), &BACKGROUND_EASING
        ))
      }
    />

    <text font-family="DM Sans" x="100" y="300" font-size="150">
      "Hello World!"
    </text>

    <text font-weight="500" font-family="JetBrains Mono" x="100" y="440" font-size="74" fill="#4b5563">
      {format!("This frame index: {}, second: {:.2}", frame.index, frame.get_current_second())}
    </text>
  </svg>
)
```


localhost
HelloWorldVideo
DMSans-Regular.ttf
DM Sans (U+0-10ffff)
JetBrainsMono-Regular.ttf
JetBrains Mono (U+0-10ffff)
Hello World!
This frame index: 0, second: 0.00
00:00
00:04
00:09
00:13
00:17
00:21
00:26
00:30
Hello World!
This frame index: 0, second: 0.00
Hello World!
This frame index: 150, second: 5.00
Hello World!
This frame index: 300, second: 10.00
Hello World!
This frame index: 450, second: 15.00
Hello World!
This frame index: 600, second: 20.00
Hello World!
This frame index: 750, second: 25.00
Hello World!
This frame index: 900, second: 30.00
00:00 / 00:30
FPS 30


HelloWorldVideo
DMSans-Regular.ttf
DM Sans (U+0-10FFFF)
JetBrainsMono-Regular.ttf
JetBrains Mono (U+0-10FFFF)
text 404.36 × 95
Hello World!
This frame index: 817, second: 27.23
00:00
00:05
00:10
00:15
00:20
00:25
00:30
Hello World!
This frame index: 0, second: 0.00
Hello World!
This frame index: 180, second: 6.00
Hello World!
This frame index: 360, second: 12.00
Hello World!
This frame index: 540, second: 18.00
Hello World!
This frame index: 720, second: 24.00
Hello World!
This frame index: 900, second: 30.
00:27 / 00:30
FPS 30
Elements
Console
<!DOCTYPE html>
<html lang="en" class="dark" style="color-scheme: dark">
<head>…</head>
<body>

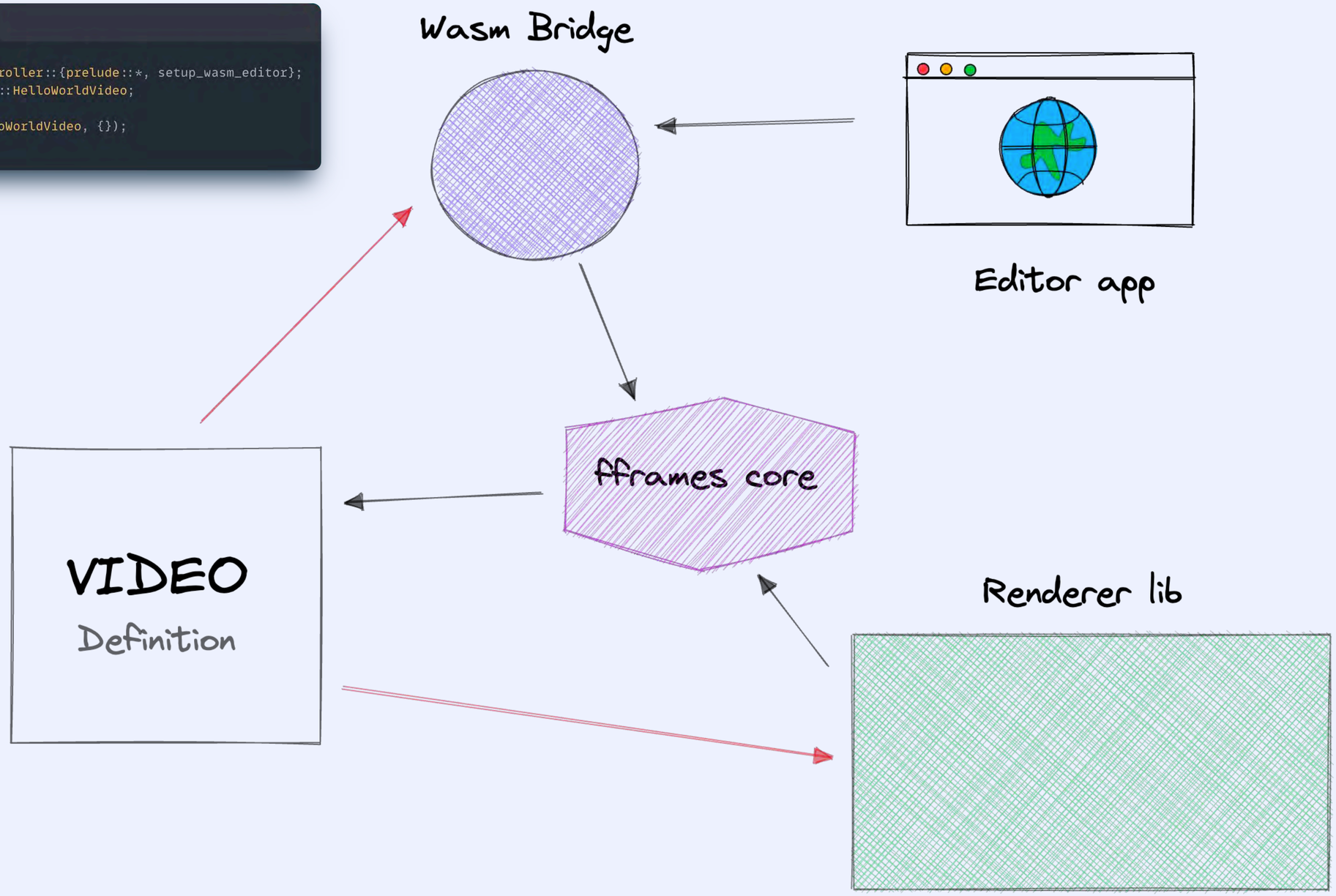
…

<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="1920" height="1080">
<rect width="1920" height="1080" x="0" y="0" fill="rgb(242, 249, 249)"></rect>
<text font-family="DM Sans" x="100" y="300" font-size="150">Hello World!</text> == $0
<text font-weight="500" font-family="JetBrains Mono" x="100" y="440" font-size="74" fill="#4b5563">This frame index: 817, second: 27.23</text>
</svg>
<script type="module" src="./main.tsx"></script>
<script>…</script>
</body>
</html>
Styles
Computed
Layout
Event Listeners
DOM Breakpoints
Filter
:hov .cls
element.style {
}
*, ::before, ::after {
<style>
--tw-border-spacing-x: 0;
--tw-border-spacing-y: 0;
--tw-translate-x: 0;
--tw-translate-y: 0;


Layers
Assets
Page 1
Frame
This frame index: 817, second:...
Hello World!
Vector
Animateable
Frame 1
Share
53%
Design
Prototype
Inspect
Frame
X -1325
Y 2009
W 1920
H 1080
0°
0
Clip content
Auto layout
Layout grid
Layer
Pass through
100%
Fill
FFFFFF
100%
Stroke
Selection colors
000000
100%
4B5563
100%
F2F9F9
100%
Effects
Export
Frame
Hello World!
This frame index: 817, second: 27.23
1920 × 1080

encoder.rs

```rust
use fframes_editor_controller::{prelude::*, setup_wasm_editor};
use hello_world_example::HelloWorldVideo;

setup_wasm_editor!(HelloWorldVideo, {});
```

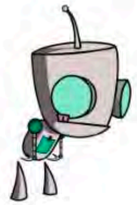


# To be rendered

# 1500 tests

RazrFalcon/**resvg**

An SVG rendering library.

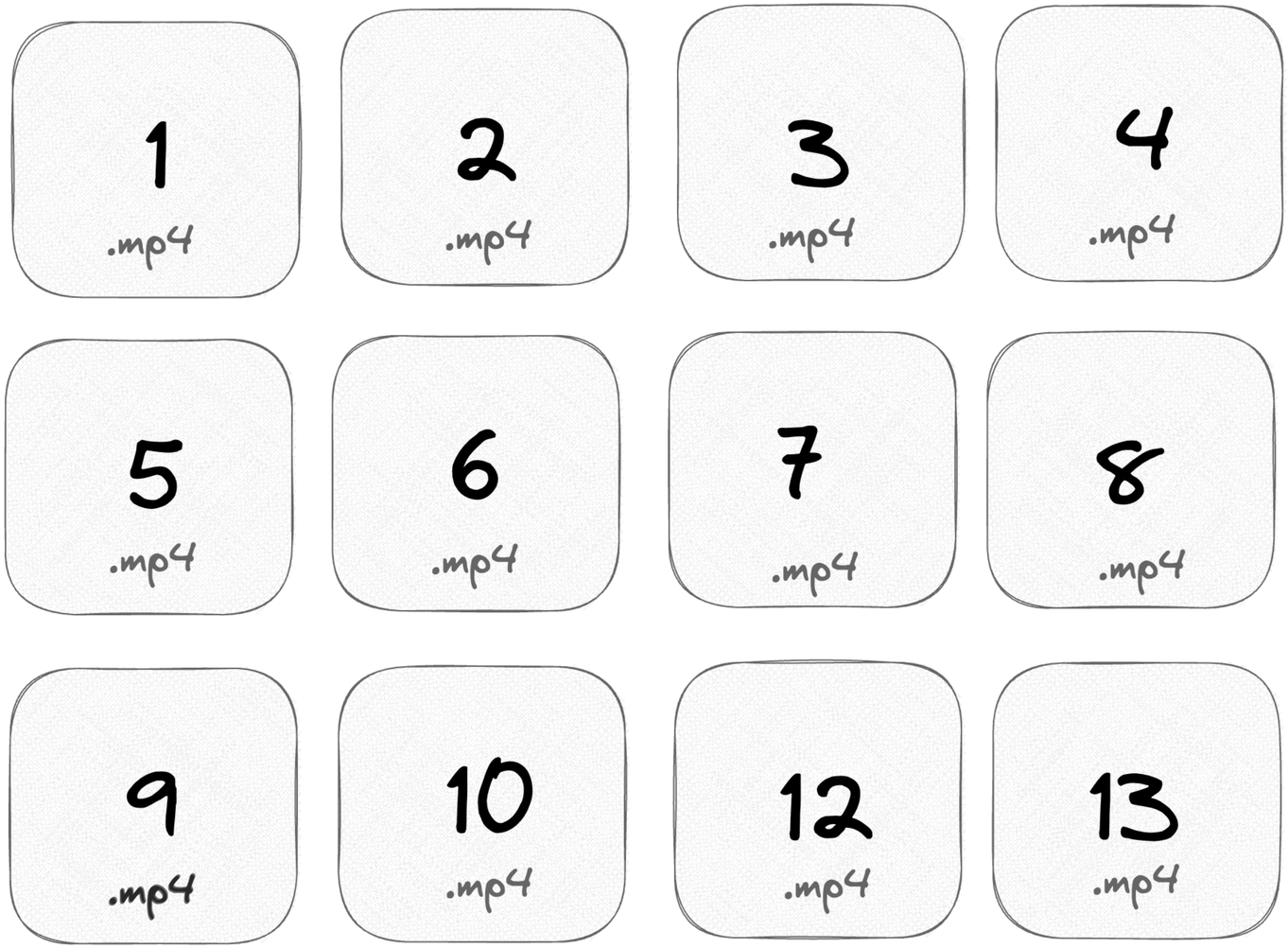


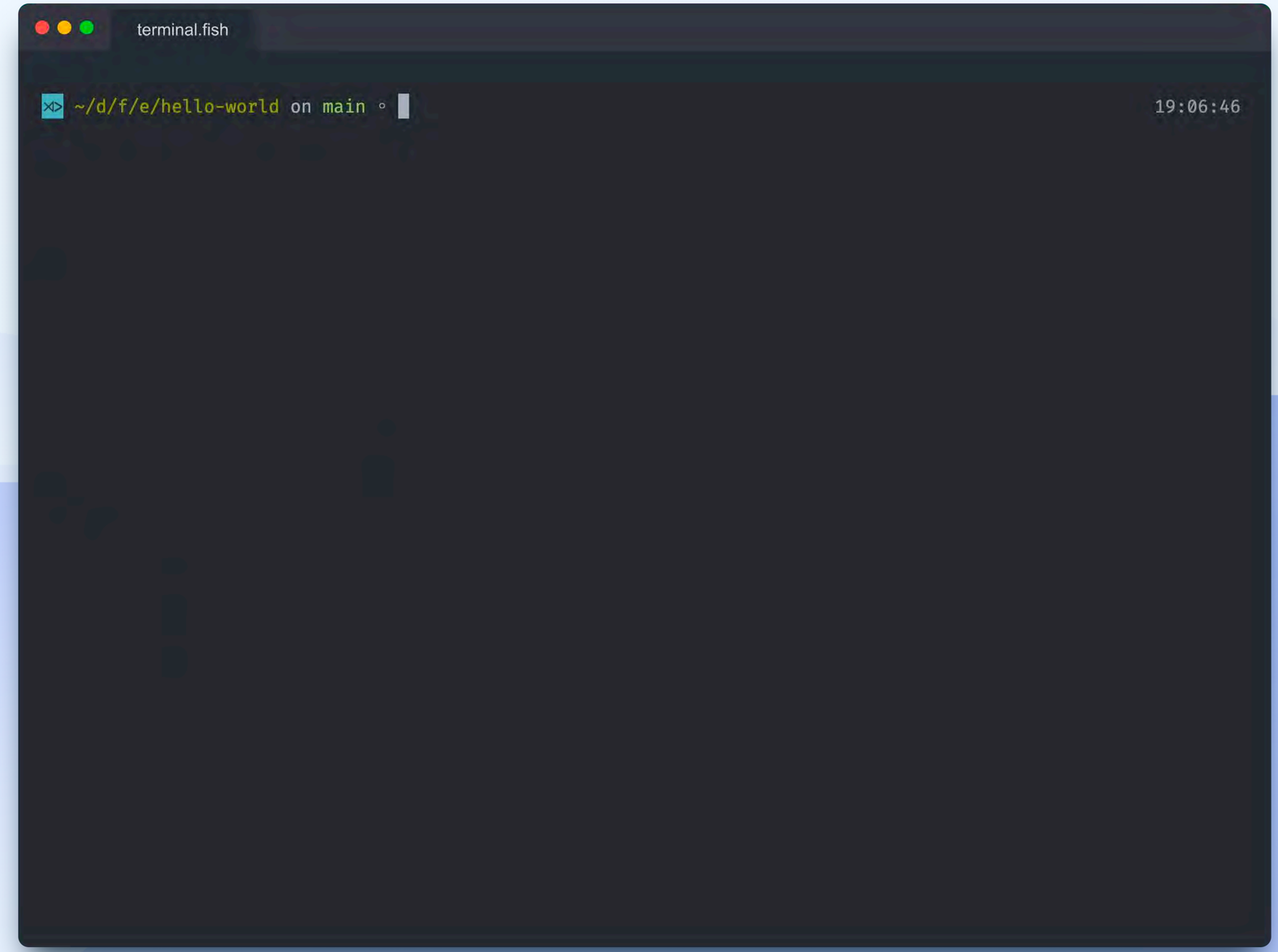
37 Contributors

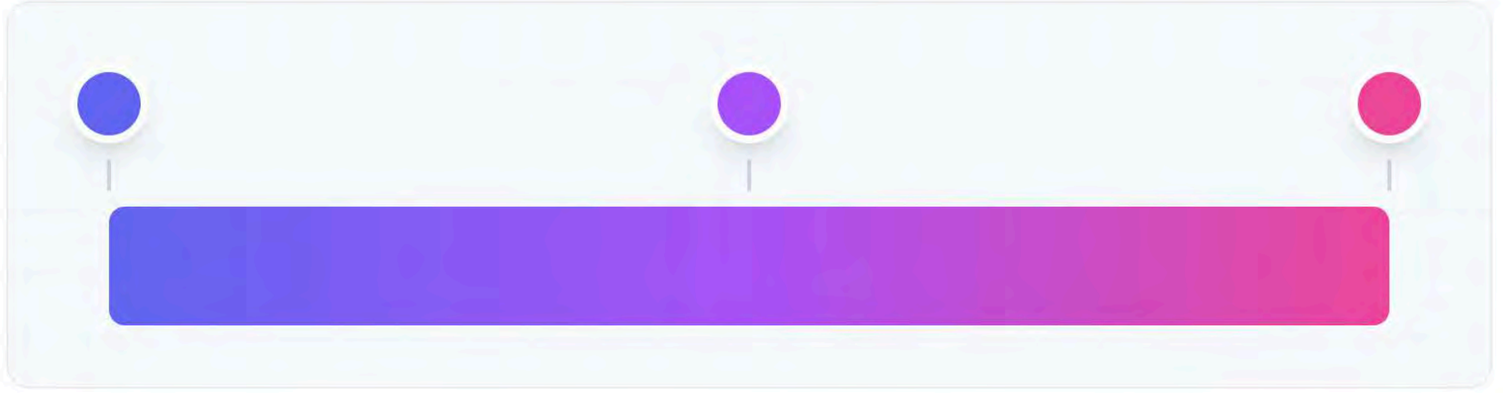
2k Used by

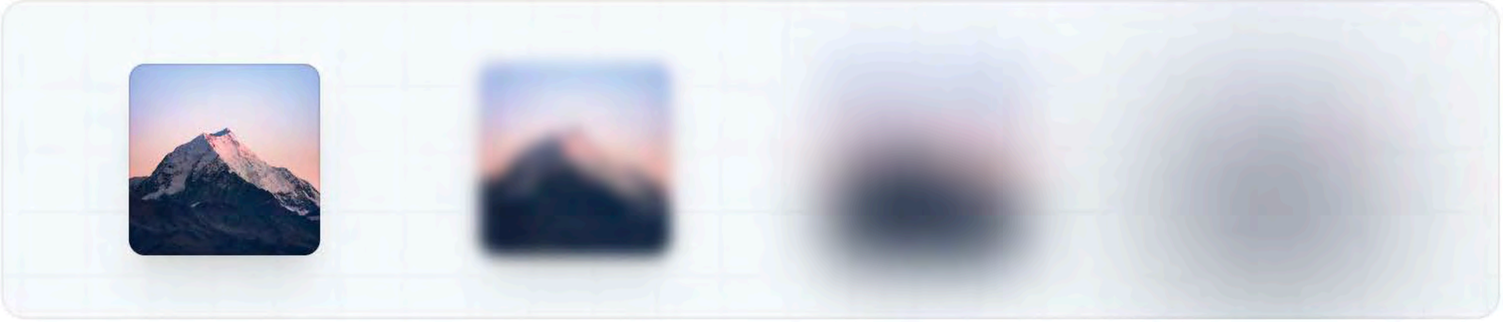
2k Stars

140 Forks


CPU
1
.mp4
2
.mp4
3
.mp4
4
.mp4
5
.mp4
6
.mp4
7
.mp4
8
.mp4
9
.mp4
10
.mp4
12
.mp4
13
.mp4


terminal.fish
~/d/f/e/hello-world on main ◦
19:06:46

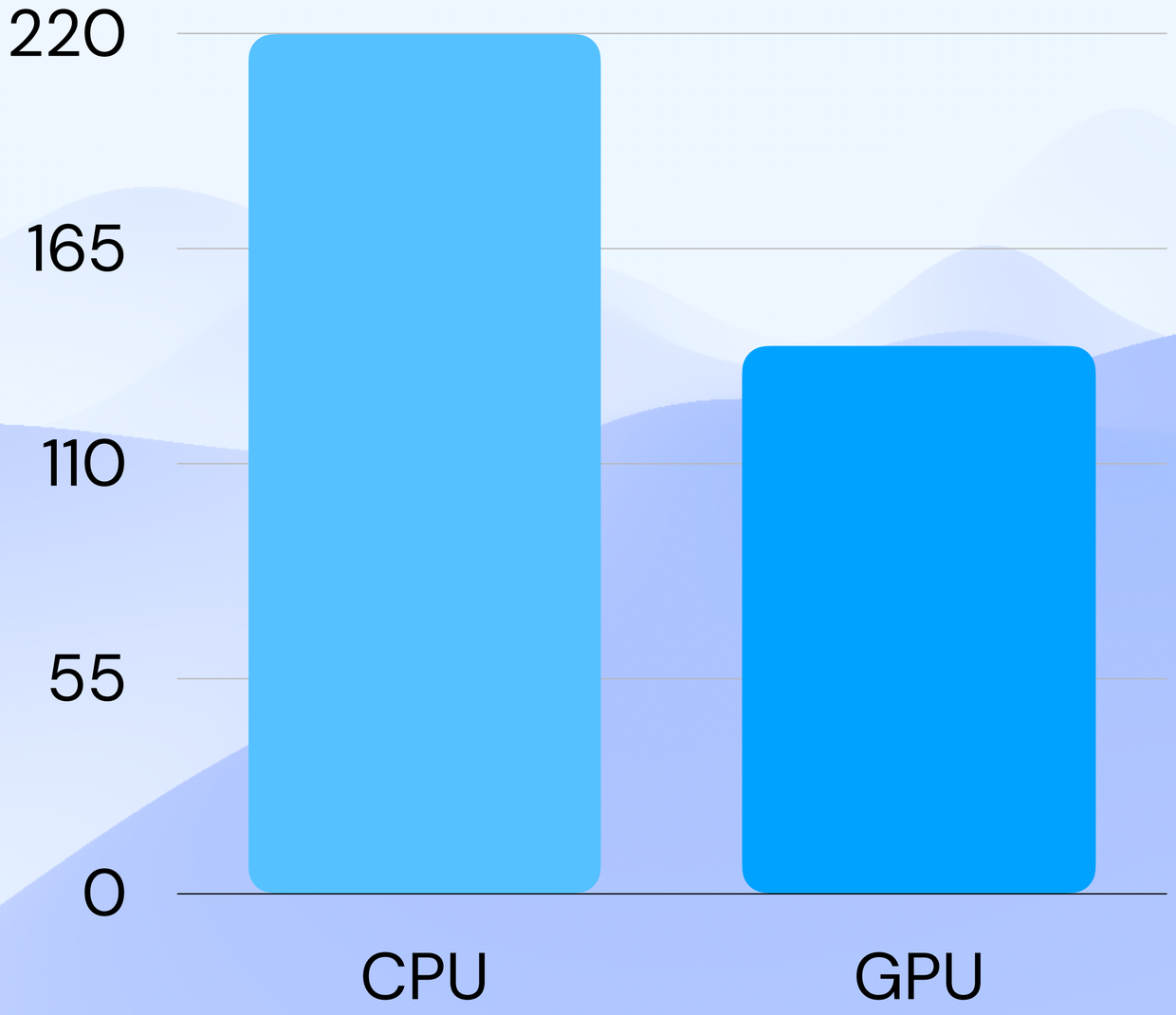




# GPU

useless but gorgeous

# Hello World

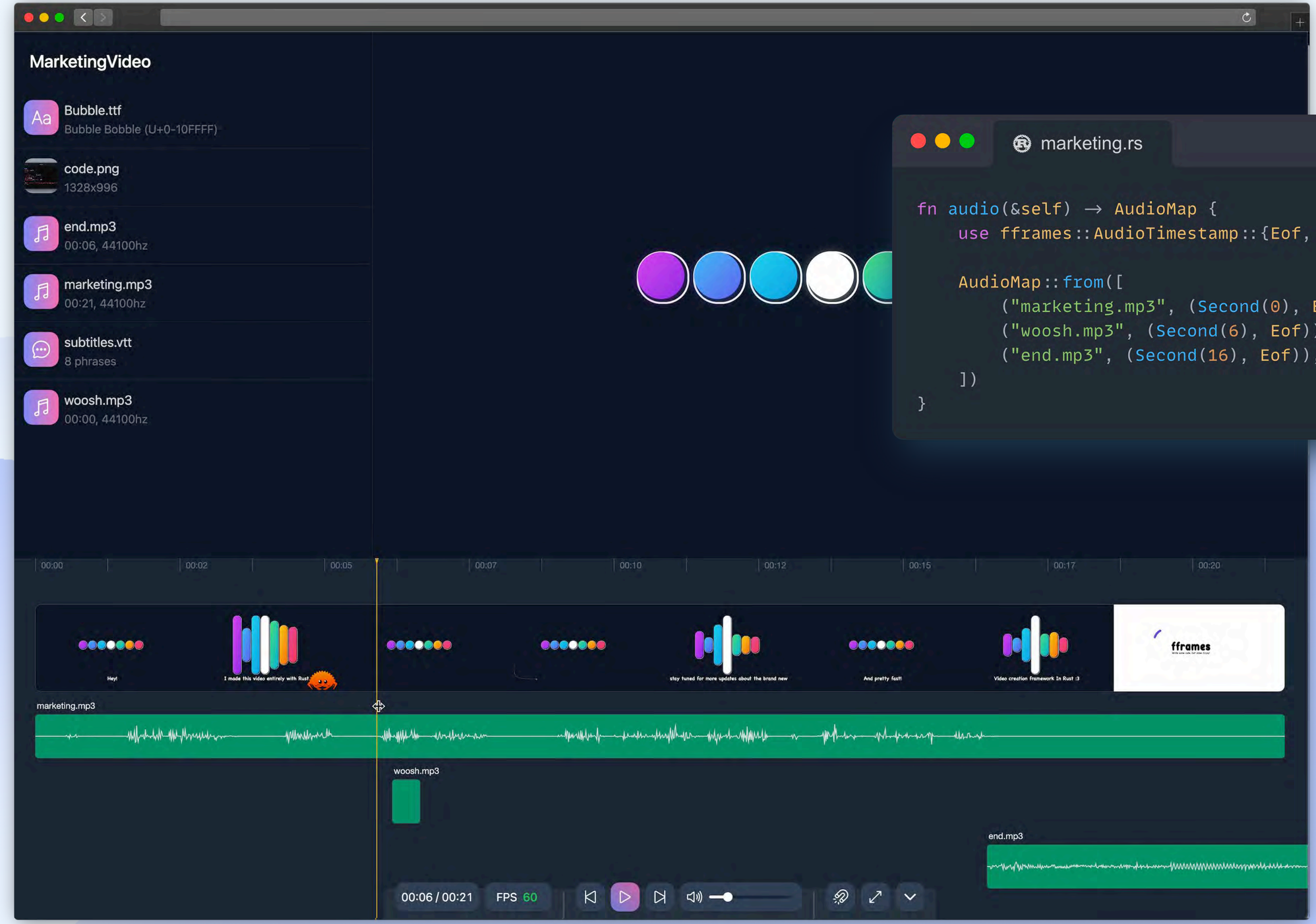


# Audio

**the most important part of a video**

audio_encoder.rs

```rust
// increate the diff by 110haz each second
let diff_hz = 2.0 * PI * 110.0 / 44100.;

for (i, sample) in samples.iter().enumerate() {
    let sample = libm::sin(time * 1000.);

    (*frame).data[0][i] = sample;
    time += diff_hz;
}
```


MarketingVideo
Bubble.ttf
Bubble Bobble (U+0-10FFFF)
code.png
1328x996
end.mp3
00:06, 44100hz
marketing.mp3
00:21, 44100hz
subtitles.vtt
8 phrases
woosh.mp3
00:00, 44100hz
marketing.rs
fn audio(&self) -> AudioMap {
use fframes::AudioTimestamp::{Eof, Second};
AudioMap::from([
("marketing.mp3", (Second(0), Eof)),
("woosh.mp3", (Second(6), Eof)),
("end.mp3", (Second(16), Eof)),
])
}
00:00
00:02
00:05
00:07
00:10
00:12
00:15
00:17
00:20
Hey!
I made this video entirely with Rust
stay tuned for more updates about the brand new
And pretty fast!
Video creation framework In Rust :3
fframes
marketing.mp3
woosh.mp3
end.mp3
00:06 / 00:21
FPS 60

podcast.rs

```
let guest_vis = fframes::audio_data::visualize_audio_frame(
    frame,
    &audio_data::VisualizeFrameInput {
        smooth_level: 2,
        ctx,
        audio: ctx.get_audio_data("guest.mp3"),
        sample_size: audio_data::SampleSize::S32,
        window: None,
    },
);

svgr!(
  <svg>
  {
    guest_vis.iter().enumerate().map(|(i, fr)|  {
      // smooth raw frequencies to get decibels
      let db = 10.0 * libm::log10f(*fr);
      let db = db.max(10.0);

      svgr!(
        <rect
          y={(950) as f32 - db / 2.0}
          x={800 + (i * 20)}
          fill="#E7D850"
          height={db}
          width="16"
          rx="4"
          ry="4"
        />
      )})
   }
  </svg>
)
```

# Videos

are interesting

# fframes

Write some code. Get video. Enjoy!

# Beta

starts right now

# Discord

put your @githubName to the **#Beta** chat

https://fframes.studio/

# Thank you for watching!

BY THEGOOSE