An Introduction: „The most inefficient & imperformant Video-Rendering-Engine possible“

Julius Sebastian Figge

9. November 2022

1. Code
2. Demo

Code[1]

Listing on document load

```
1          document.addEventListener('DOMContentLoaded', () => {
2            canvas = document.querySelector('#playback-canvas');
3            ctx = canvas.getContext('2d');

5            if (!window.MediaStreamTrackProcessor) {
6              document.querySelector('#not-supported-warning')
                   .style.display = 'unset';
7            }
8          })
```

[1] SourceCode: [Fig22b]

Listing extractFrames()

```
1       const extractFrames = async (src,videoScalingFactor) => {
2         const track = await getVideoTrack(src);
3
4         setFrameRateFromVideoTrack(track);
5
6         const processor = new MediaStreamTrackProcessor(track);
7         const reader = processor.readable.getReader();
8
9         await readAndProcessFrames(reader,videoScalingFactor);
10
11        //TODO reduce framerate beforehands -> this is imperformant
              as hell
12        const usableKeyFrameAmount = reduceFrameRateForKeyFrames();
13        keyFrameListToStyleDeclaration(usableKeyFrameAmount);
14
15        animateKeyFrames();
16      };
```

[2]SourceCode: [Fig22b]

Listing getVideoTrack()

```
1       const getVideoTrack = async (src) => {
2         //TODO introduce video upload?
3         const video = document.querySelector("#playback-video");
4         video.crossOrigin = "anonymous";
5         video.src = src;
6         document.body.append(video);
7         await video.play();
8         const [track] = video.captureStream().getVideoTracks();
9         video.onended = () => track.stop();
10        return track;
11      };
```

[3]SourceCode: [Fig22b]

Listing readAndProcessFrames()

```
1   const readAndProcessFrames = async (reader,videoScalingFactor)
        => {
2     await reader.read().then(async ({done, value}) => {
3       if (!value) return;
4       const calculatedWidth = value.codedWidth /
            videoScalingFactor;
5       const calculatedHeight = value.codedHeight /
            videoScalingFactor;
6       const bitmap = await createImageBitmap(value, {
7         resizeWidth: calculatedWidth,
8         resizeHeight: calculatedHeight,
9         resizeQuality: "pixelated"
10      });
11      extractPixelsFromFrame(bitmap);
12      value.close();
13      if (done) return;
14      await readAndProcessFrames(reader,videoScalingFactor);
15    });
16  };
```

Code[5]

Listing extractPixelsFromFrame()

```
1           const extractPixelsFromFrame = bitmap => {
2             const {width: w, height: h} = bitmap;
3             const canvas = new OffscreenCanvas(w, h);
4             const ctx = canvas.getContext('2d');

6             ctx.drawImage(bitmap, 0, 0);
7             const pixels = ctx.getImageData(0, 0, w, h).data;

9             boxShadowKeyFramesList.push(
                  mapPixelsToBoxShadowDeclaration(w, h, pixels));
10          };
```

[5] SourceCode: [Fig22b]

Listing mapPixelsToBoxShadowDeclaration()

```
1  const mapPixelsToBoxShadowDeclaration = (width, height, pixels) => {
2    const boxShadowPixels = [];
3
4    //extract r,g,b,a values from array -> r1,g1,b1,a1,r2,g2,b2,a2
5    [...Array(Math.ceil(pixels.length / 4)).keys()]
6    .forEach(i => {
7      const [r, g, b, a] = pixels.slice(i * 4, (i + 1) * 4);
8
9      const x = i % width;
10     const y = Math.floor(i / width);
11
12     boxShadowPixels.push(`${x}px ${y}px rgb(${r},${g},${b},${a})`);
13   })
14
15   return boxShadowPixels.join(',') + ';';
16 };
```

Code[7]

Listing reduceFrameRateForKeyFrames()

```
1          const reduceFrameRateForKeyFrames = () => {
2            const skipRatio = Math.ceil(framerate / DESIRED_FRAMERATE)
                 ;
3            return boxShadowKeyFramesList.filter((value, index) => (
                 index % skipRatio === 0));
4          };
```

[7] SourceCode: [Fig22b]

Code[8]

Listing keyFrameListToStyleDeclaration()

```
1       const keyFrameListToStyleDeclaration = (
            usableAmountOfKeyFrames) => {
2         let totalFramesLength = usableAmountOfKeyFrames.length;
3         usableAmountOfKeyFrames = mapShadowPixelsToBoxShadow(
              usableAmountOfKeyFrames, totalFramesLength);
4         addBoxShadowFramesAsKeyframesToStyles(
              usableAmountOfKeyFrames);
5         setCssAnimationLength(usableAmountOfKeyFrames);
6       };
```

[8] SourceCode: [Fig22b]

Code[9]

Listing mapShadowPixelsToBoxShadow()

```
function mapShadowPixelsToBoxShadow(usableAmountOfKeyFrames,
    totalFramesLength) {
  return usableAmountOfKeyFrames
  .map((element, index) => {
    let keyFramePercentage = parseFloat(String(index /
        totalFramesLength)).toFixed(2);
    return keyFramePercentage + '% { box-shadow: ' + element
        + '}'
  });
}
```

[9] SourceCode: [Fig22b]

Listing addBoxShadowFramesAsKeyframesToStyles()

```
1  const addBoxShadowFramesAsKeyframesToStyles =
       usableAmountOfKeyFrames => {
2    const cssMovieKeyframes = document.createElement('style');
3    const keyFramesDeclaration = '@keyframes css-movie {' +
       usableAmountOfKeyFrames.join(' ') + '}';
4    const rules = document.createTextNode(keyFramesDeclaration
       );
5    cssMovieKeyframes.appendChild(rules);
6    document.getElementsByTagName("head")[0].appendChild(
       cssMovieKeyframes);
7  };
```

[10] SourceCode: [Fig22b]

Code[11]

Listing animateKeyFrames()

```
1          const animateKeyFrames = () => {
2            document.querySelector('.css-video').style.animation = `
                 css-movie ${cssVideoAnimationLength}s steps(1, end)`;
3          };
```

[11] SourceCode: [Fig22b]

Listing animateKeyFrames()

```
1          <h1 id="not-supported-warning" style="color: red; display:
               none">Your browser is not supported! Please refer to
               the
2          <a href="./README.md">Readme</a> and use a recent Chromium
               based Browser.</h1>

4          <div class="wrapper">
5            <div class="videos">
6              <canvas id="playback-canvas"></canvas>
7              <video id="playback-video"></video>
8              <div class="css-video">
9            </div>
10         </div>
11         <div class="buttons">
12           <button id="playback-button" onclick="extractFrames('$URL
                 ',12)">$EXAMPLE_VIDEO</button>
13         </div>
```

[12]SourceCode: [Fig22b]

Demo

LiveDemo![13]

[13] [Fig22a]

End - Questions?

[Fig22a] Julius Figge. *css-to-video*. 2022. URL: https://video-to-css.jeujeus.de/ (besucht am 09. 11. 2022).

[Fig22b] Julius Figge. *Sourcecode Repo*. 2022. URL: https://github.com/JeuJeus/video-to-css (besucht am 09. 11. 2022).