











Use Directive to attach images

WebWorkers.....

Emit a new event when processing is done





```

1 export class ColorFromImgDirective implements OnChanges {
2   @Input()
3   src: string;
4
5   @Output()
6   newColor = new EventEmitter();
7
8   worker: Worker;
9   image: Partial<CanvasImage>;
10  private imgSrc = "";
11  constructor() {
12    this.worker = new Worker(new URL("./quantize.worker", import.meta.url));
13    this.worker.onmessage = ({ data }) => {
14      this.image.removeCanvas();
15      this.newColor.emit(data);
16    };
17  }
18  ngOnChanges({ src }: SimpleChanges) {
19    if (src.currentValue === "assets/imgs/default.svg") {
20      if (src.firstChange) {
21        this.imgSrc = src.currentValue;
22        this.getPaletteFromUrl(this.imgSrc);
23      } else {
24        const incomingVal = new URL(src.currentValue);
25        const currentVal = new URL(src.previousValue);
26        if (currentVal.pathname !== incomingVal.pathname) {
27          this.imgSrc = src.currentValue;
28          this.getPaletteFromUrl(this.imgSrc);
29        }
30      }
31    }
32  }
33
34  getPalette(
35    sourceImage: HTMLImageElement,
36    colorCount?: number,
37    quality?: number
38  ) {
39    if (
40      typeof colorCount === "undefined" ||
41      colorCount < 2 ||
42      colorCount > 256
43    ) {
44      colorCount = 10;
45    }
46    if (typeof quality === "undefined" || quality < 1) {
47      quality = 10;
48    }
49
50    // Create custom CanvasImage object
51    this.image = canvasImage(sourceImage);
52    const imageData = this.image.getImageData();
53    const pixels = imageData.data;
54    const pixelCount = this.image.getPixelCount();
55    this.worker.postMessage([pixels, pixelCount, colorCount, quality]);
56  }
57
58  getPaletteFromUrl(imageUrl: string, quality = 10) {
59    new Promise((resolve, reject) => {
60      const sourceImage = new Image();
61      sourceImage.crossOrigin = "Anonymous";
62      sourceImage.addEventListener("load", () => {
63        const palette = this.getPalette(sourceImage, 5, quality);
64        resolve([palette, imageUrl]);
65      });
66      sourceImage.src = imageUrl;
67      sourceImage.addEventListener("error", reject.bind(this));
68    });
69  }
70 }

```

