



**Codes meiner**  
BACHELORTHESIS

Sonja Schwab Sommersemester 2020

## Vorwort

In diesem Anhang zur Dokumentation meiner Bachelorthesis geht es ausschließlich um die Codes, die ich während der Bearbeitungszeit programmiert habe. Ich erstellte generative Designs als Key Visuals und ein generatives Logo.

Die Seiten habe ich so aufgebaut, dass man links den Code sieht und rechts kurze Erklärungen dazu hat. Die Ergebnisse des Codes zeige ich direkt daneben, damit man sich vorstellen kann wie der Code visualisiert aussieht.

Natürlich sind in diesem Buch hier nicht alle Codes enthalten die ich programmiert habe, denn viele haben entweder nicht funktioniert oder sahen nicht gut aus. In diesem Buch sind trotzdem über 800 Zeilen Code zusammen gekommen die meine Entwicklung meiner Meinung nach ganz gut zeigen.

Ein Grund warum ich dieses Buch machen wollte war, dass es mir eine Hilfe für die Zukunft sein soll, wenn ich wieder programmieren werde. Es soll mir den Einstieg das nächste Mal erleichtern, denn ich merkte während der Bearbeitungszeit, dass ich schon nach ein paar Wochen Pause Einiges vergessen hatte.

Ich programmierte mit der Sprache „p5.js“ in dem dazu entwickelten Online Editor von p5.js. Mit dem untenstehenden Link kommt man direkt zu meinem Ordner mit allen Codes der Thesis. Da viele Codes im bewegten Zustand schöner aussehen und es Spaß macht sie auszuprobieren, kann ich empfehlen sie anzusehen.



<https://editor.p5js.org/sonne51196/collections/hXXjfuXbL>

# Inhalt

## CODES FÜR DIE KEY VISUALS



Foto mit  
Rechtecken  
pixeln

**S. 6-7**



Pixel neu  
einfärben

**S. 8-9**



Foto mit  
Ellipsen  
neu einfärben

**S. 10-11**



Ellipsen über  
Bild zeichnen

**S. 12-13**



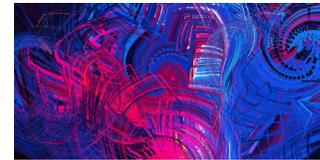
Anwachsende  
Ellipsen

**S. 14-15**



Bewegte  
Ellipsen

**S. 16-17**



Perlin noise  
mit Interak-  
tion

**S. 18-21**



Marble  
Effekt mit  
Ellipsen

**S. 22-43**



Marble  
Effekt mit  
Rechtecken

**S. 44-45**

## CODES FÜR DAS LOGO



Logo mit  
wechselnden  
Elementen

**S. 46-49**



Logo Entwurf  
„Konfetti“

**S. 50-51**



finale Logo

**S. 52-57**

## Foto mit Rechtecken pixeln

```
let img;  
let smallrect, largerec;
```

```
function preload() {  
  img = loadImage('images/disco-kugel.jpg');  
}
```

```
function setup() {  
  createCanvas(800, 800);  
  smallrect = 10;  
  largerec = 70;  
  imageMode(CENTER);  
  noStroke();  
  background(255);  
  frameRate(1200);  
}
```

```
function draw() {  
  let pointillize = map(mouseX, 0, 400, smallrect,  
largerec);  
  let x = floor(random(img.width));  
  let y = floor(random(img.height));  
  let pix = img.get(x, y);  
  fill(pix, 128);  
  rect(x, y, pointillize, pointillize);  
}
```

```
function mousePressed() {  
  background(255);  
}
```

## ERKLÄRUNG DES CODES

Das Foto wird geladen

Ich erstelle eine 800px mal  
800 px große  
„Leinwand“

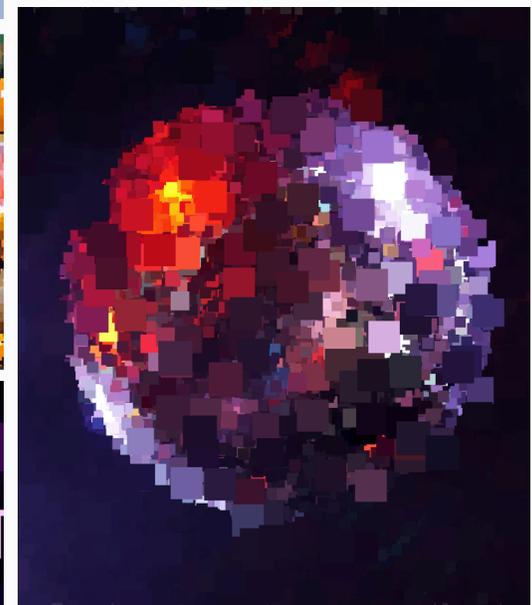
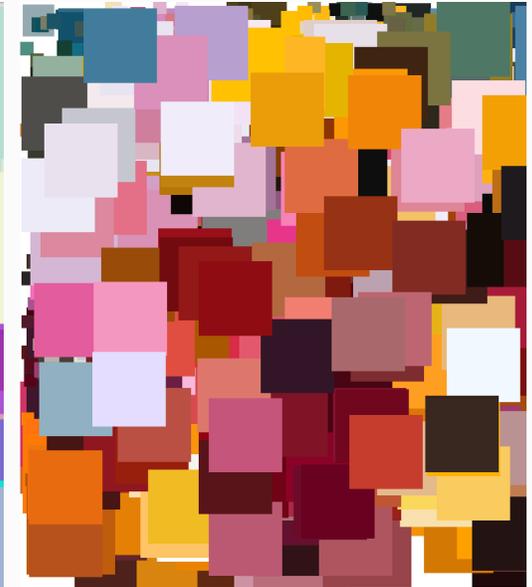
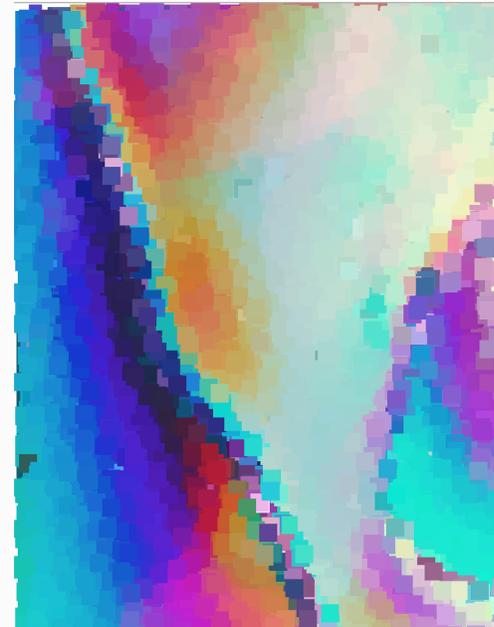
Ich definiere, dass alle klei-  
nen Rechtecke 10px und alle  
Großen 70px haben sollen

Durch Bewegen der Maus  
nach links werden kleine  
Rechtecke gezeichnet und  
nach rechts Größere

auf der ganzen Leinwand  
werden Rechtecke verteilt,  
die die Farbe ihres Pixels in  
der Mitte annehmen

Beim Klick auf die Leinwand  
wird diese wieder weiß und  
neue Pixel werden gezeich-  
net

## ERGEBNISSE



## Alle Pixel neu einfärben

## ERKLÄRUNG DES CODES

```
var img;

var farben = ['#b9dad3', '#20115f', '#0ce8fe', '#5733af', '#ee2559'];

function preload() {
  img = loadImage('data/women-sunglasses-800.jpg');
}

function setup() {
  createCanvas(300, 300);
}

function draw() {
  image(img, 0, 0);
  for (var x = 0; x < width; x++) {
    for (var y = 0; y < height; y++) {
      let r = red(img.get(x, y));
      let g = green(img.get(x, y));
      let b = blue(img.get(x, y));

      let min = 1000;
      let minFarbe = 0;

      for (var f = 0; f < farben.length; f++) {
        let d = dist(r, g, b,
          red(farben[f]),
          green(farben[f]),
          blue(farben[f]));

        if (d < min) {
          min = d;
          minFarbe = f;
        }
      }
      stroke(farben[minFarbe]);
      point(x, y);
    }
  }
}
```

Ich wähle 5 Farben aus dem Originalbild aus

Das Foto wird geladen

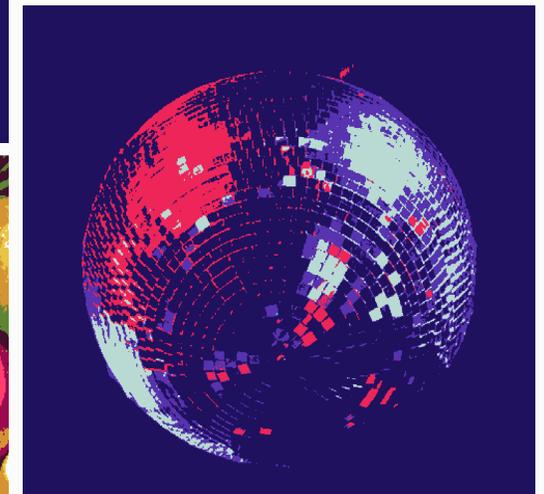
Ich erstelle eine 300px mal 300 px große „Leinwand“

Die RGB Werte jedes Pixels im Foto werden ausgelesen

Jedem Pixel wird eine der vorher festgelegten Farben neu zugeordnet, je nachdem welche Farbe der Ursprungsfarbe am ähnlichsten ist

Punkte mit 1px Durchmesser und den neuen Farben werden auf die Leinwand gezeichnet

## ERGEBNISSE



## Foto mit Ellipsen pixeln

```
var img;  
var farben = ['#20115f', '#ee2559', '#b9dad3',  
             '#5733af'];  
  
function preload() {  
  img = loadImage('data/neon-hallway.jpg');  
}  
  
function setup() {  
  createCanvas(750, 800);  
}  
  
function draw() {  
  background(255);  
  image(img, 0, 0);  
  for (var x = 0; x < width; x+=10) {  
    for (var y = 0; y < height; y+=10) {  
      let r = red(img.get(x, y));  
      let g = green(img.get(x, y));  
      let b = blue(img.get(x, y));  
  
      let min = 1000;  
      let minFarbe = 0;  
  
      for (var f = 0; f < farben.length; f++) {  
        let d = dist(r, g, b,  
                   red(farben[f]),  
                   green(farben[f]),  
                   blue(farben[f]));  
  
        if (d < min) {  
          min = d;  
          minFarbe = f;  
        }  
      }  
      noStroke();  
      fill(farben[minFarbe]);  
      ellipse(x,y,10,10);  
    }  
  }  
  noLoop();  
}
```

## ERKLÄRUNG DES CODES

Ich wähle 4 Farben aus, mit denen ich die Ellipsen im Bild später füllen möchte

Das Foto wird geladen

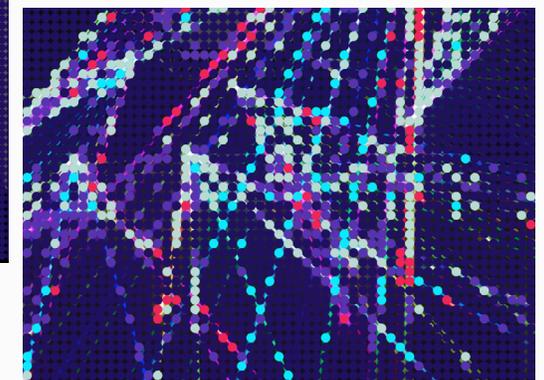
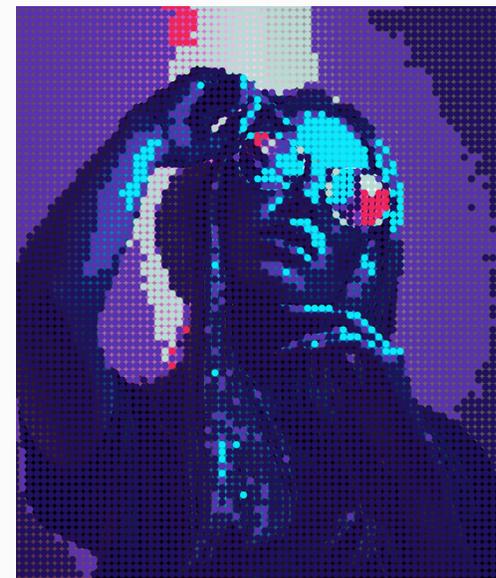
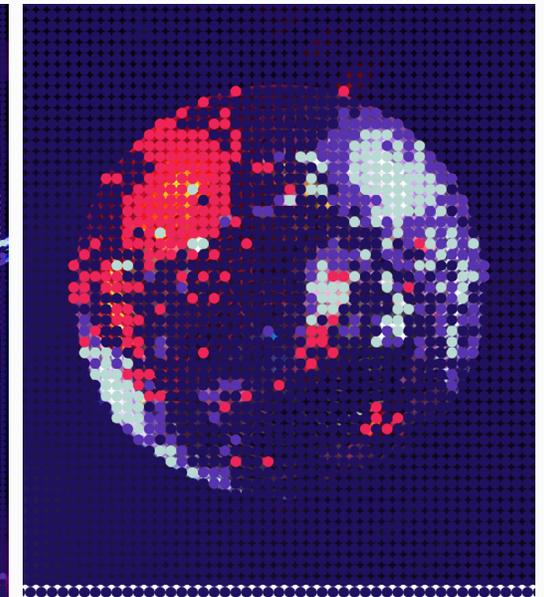
Ich erstelle eine 750px mal 800 px große Leinwand

Die RGB-Werte jedes Pixels im Foto werden ausgelesen

Jedem Pixel wird eine der vorher festgelegten Farben neu zugeordnet, je nachdem welche Farbe der Ursprungsfarbe am ähnlichsten ist

Die ganze Leinwand wird mit 10 px großen Ellipsen gefüllt und diese werden mit den neuen Farben gefüllt

## ERGEBNISSE



## Ellipsen über das Foto zeichnen

```
var img;  
var myArray = [];
```

```
function preload() {  
  img=loadImage('data/schwimmtiere.jpg');  
}
```

```
function setup() {  
  createCanvas(600, 500);  
  img.resize(width, height);  
  for(var i=0; i<10000; i++){  
    myArray.push(new particle());  
  }  
}
```

```
function draw() {  
  image(img,0,0);  
  for(var i=0; i<myArray.length; i++){  
    myArray[i].zeigen();  
  }  
}
```

```
//saveCanvas();  
//noLoop();  
}
```

```
class particle{  
  constructor(){  
    this.px = random(width);  
    this.py = random(height);  
    this.r= random(3,10);  
    this.c=color(img.get(this.px,this.py));  
  }
```

```
  zeigen(){  
    noStroke();  
    fill(this.c);  
    ellipse(this.px,this.py,this.r,this.r);  
  }  
}
```

## ERKLÄRUNG DES CODES

Das Foto wird geladen

Ich erstelle eine 600px mal  
500 px große Leinwand

Ich lasse 1000 „Partikel“  
erstellen

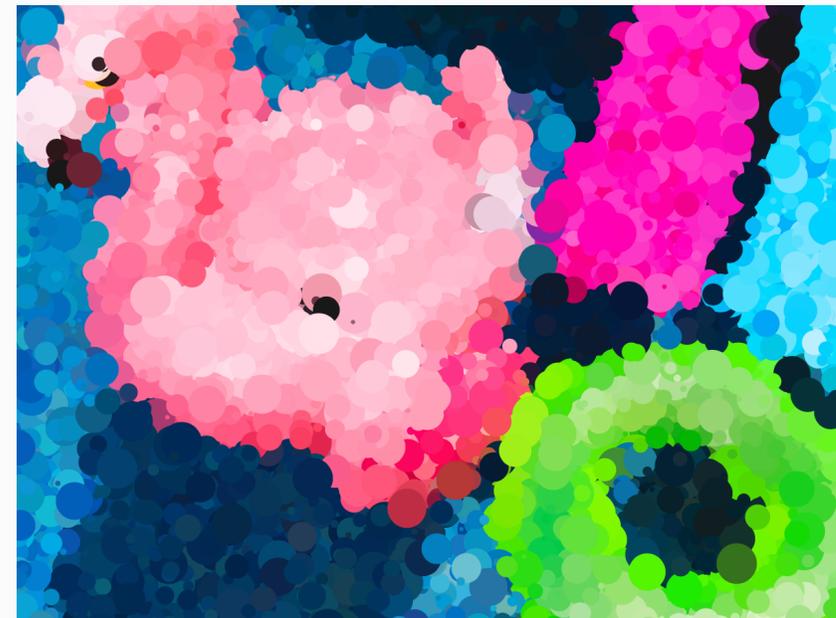
Die „Partikel“ werden auf  
der gesamten Leinwand zu-  
fällig verteilt

Sie bekommen zufällig einen  
Radius zwischen 3px und  
10 px und werden somit  
Ellipsen

Die Ellipsen nehmen die  
Farbe an, die der Pixel in der  
Mitte der Ellipse hat

Durch Variation des Radius  
der Ellipsen kann das Bild  
sehr leicht stark verändert  
werden

## ERGEBNISSE



# Anwachsende Ellipsen

```
var img;  
var myArray = [];  
  
function preload() {  
  img=loadImage('data/disco-kugel.jpg');  
}  
  
function setup() {  
  createCanvas(750,500);  
  frameRate(20);  
  img.resize(width, height);  
  for(var i=0; i<20000; i++){  
    myArray.push(new particle());  
  }  
}  
  
function draw() {  
  //image(img,0,0);  
  for(var i=0; i<myArray.length; i++){  
    myArray[i].zeigen();  
    myArray[i].move();  
  }  
}  
  
class particle{  
  constructor(){  
    this.px = random(width);  
    this.py = random(height);  
    this.r= 5;  
    this.c=color(img.get(this.px,this.py));  
    this.speed=1;  
  }  
  zeigen(){  
    noStroke();  
    fill(this.c);  
    ellipse(this.px,this.py,this.r,this.r);  
    this.r+= 0.3;  
  }  
  move() {  
    this.speed = this.speed*0.9;  
    this.px += random(-this.speed, this.speed);  
    this.py += random(-this.speed, this.speed);  
  }  
}
```

## ERKLÄRUNG DES CODES

Das Foto wird geladen

Ich erstelle eine 750px mal 500 px große Leinwand

Ich lege eine FrameRate von 20 Frames per second fest

Ich lasse 20000 „Partikel“ erstellen

Die Funktionen „zeigen“ und „move“ werden ausgeführt

Die „Partikel“ werden auf der gesamten Leinwand zufällig verteilt

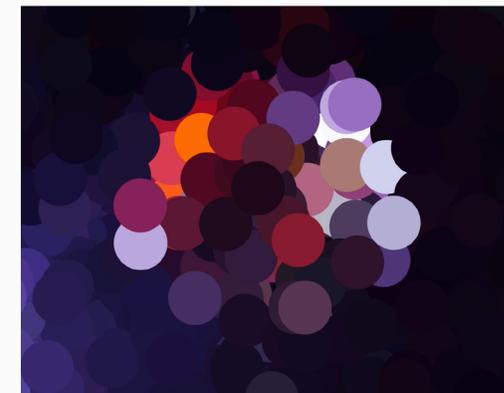
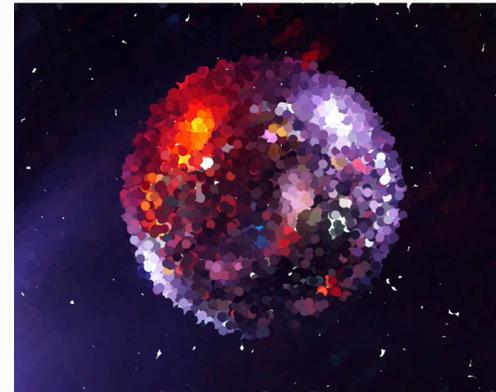
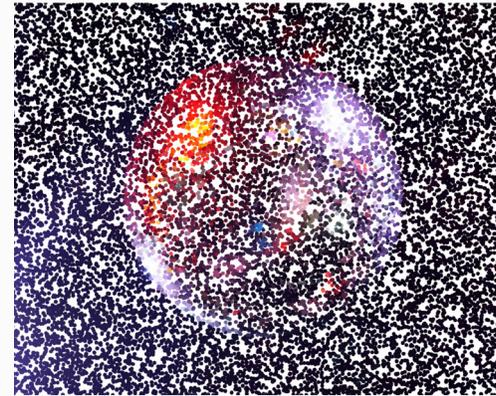
Sie bekommen eine Anfangsgröße von 5px ...

... und nehmen die Farbe an, die der Pixel in der Mitte der Ellipse hat

Der Radius der Ellipsen vergrößert sich gleichmäßig um 0.3

Die Ellipsen wachsen gleichmäßig immer größer an

## ERGEBNISSE



## Bewegte Ellipsen

```
var img;  
var myArray = [];  
  
function preload() {  
  img = loadImage('data/woman-future-  
electro900.jpg');  
}  
  
function setup() {  
  createCanvas(600, 500);  
  frameRate(20);  
  img.resize(width, height);  
  for (var i = 0; i < 6000; i++) {  
    myArray.push(new particle());  
  }  
}  
  
function draw() {  
  for (var i = 0; i < myArray.length; i++) {  
    myArray[i].zeigen();  
    myArray[i].move();  
  }  
}  
  
class particle {  
  constructor() {  
    this.px = random(width);  
    this.py = random(height);  
    this.r = random(3, 20);  
    this.c = color(img.get(this.px, this.py));  
    this.speed = 1;  
  }  
  
  zeigen() {  
    noStroke();  
    fill(this.c);  
    ellipse(this.px, this.py, this.r, this.r);  
  }  
  
  move() {  
    this.speed += 0.025;  
    this.px += random(-this.speed,  
this.speed);  
    this.py += random(-this.speed,  
this.speed);  
  }  
}
```

## ERKLÄRUNG DES CODES

Das Foto wird geladen

Ich erstelle eine 750px mal  
500 px große  
„Leinwand“

20000 „Partikel“ werden  
erstellt

Die „Partikel“ werden auf  
der gesamten Leinwand zu-  
fällig verteilt

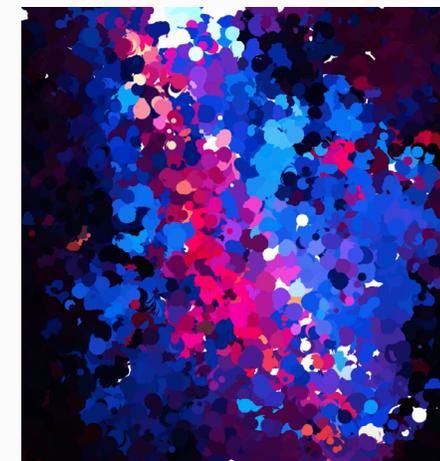
sie bekommen zufällig eine  
Größe zwischen 3px und  
20 px

und nehmen die Farbe an,  
die der Pixel in der Mitte der  
Ellipse hat

Die Geschwindigkeit erhöht  
sich immer mehr um den  
Faktor 0.025

die Ellipsen bewegen sich  
mit zunehmender Ge-  
schwindigkeit und verteilen  
sich immer stärker auf der  
Leinwand

## ERGEBNISSE



## Marble Effekt mit mouse Interaktion

```
var img;  
var status = 0;  
var myArray = [];  
var noiseSize = 0.01;
```

```
function preload() {  
  img = loadImage('data/woman-future-  
electro900.jpg');  
}
```

```
function setup() {  
  createCanvas(800, 600);  
  frameRate(30);  
  img.resize(width, height);  
  image(img, 0, 0);  
  for (var i = 0; i < 8000; i++) {  
    myArray.push(new particle());  
  }  
}
```

```
function draw() {  
  if (status == 1) {  
    for (var i = 0; i < myArray.length; i++) {  
      myArray[i].zeigen();  
      myArray[i].move3();//maus bewegung  
      myArray[i].move();  
    }  
  } else {  
    for (var j = 0; j < myArray.length; j++) {  
      myArray[j].zeigen();  
      myArray[j].move2();//ellipsen werden größer  
    }  
  }  
  if (frameCount == 30) {  
    status = 1;  
  }  
}
```

```
class particle {  
  constructor() {  
    this.px = random(width);  
    this.py = random(height);  
    this.max = random(3,20);  
    this.r = 3;  
    this.c = color(img.get(this.px, this.py));  
  }  
}
```

## ERKLÄRUNG DES CODES

Das Foto wird geladen

Ich erstelle eine 800px mal  
600 px große  
„Leinwand“  
Ich lege eine FrameRate von  
20 fest

20000 „Partikel“ werden  
erstellt

Die Funktionen „zeigen“,  
„move“ und „move3“ werden  
ausgeführt

Die Funktionen move2 lässt  
die Ellipsen zuerst an-  
wachsen, dann werden die  
Funktionen „move3“, „move“,  
„zeigen“ ausgeführt

Die „Partikel“ werden auf  
der gesamten Leinwand zu-  
fällig verteilt

sie bekommen eine An-  
fangsgröße von 3px und  
wachsen dann zufällig ab  
einer Größe zwischen 3px  
und 20px an

und nehmen die Farbe an,  
die der Pixel in der Mitte der  
Ellipse hat

## ERKLÄRUNG DES CODES

```
zeigen() {  
  noStroke();  
  fill(this.c);  
  ellipse(this.px, this.py, this.r, this.r);  
}
```

```
move2() {  
  this.r = frameCount*(this.max-3.0)/30.0 + 3;  
}
```

```
move() {  
  this.px += (noise(this.px * noiseSize,  
this.py * noiseSize) - 0.5) * 3.0;  
  this.py += (noise(this.px * noiseSize + 10,  
this.py * noiseSize + 10) - 0.5) * 3.0;  
  this.r *= 0.99;  
}
```

```
move3() {  
  this.vektor = createVector(mouseX -  
this.px, mouseY - this.py);  
  this.winkel = this.vektor.heading()-0.3;
```

```
this.distanz=dist(this.px, this.py, mouseX, mouseY)+3  
;  
  this.px+/-sin(-this.winkel)*300.0/this.distanz;  
  this.py+/-cos(-this.winkel)*300.0/this.distanz;  
  this.r*=0.996;
```

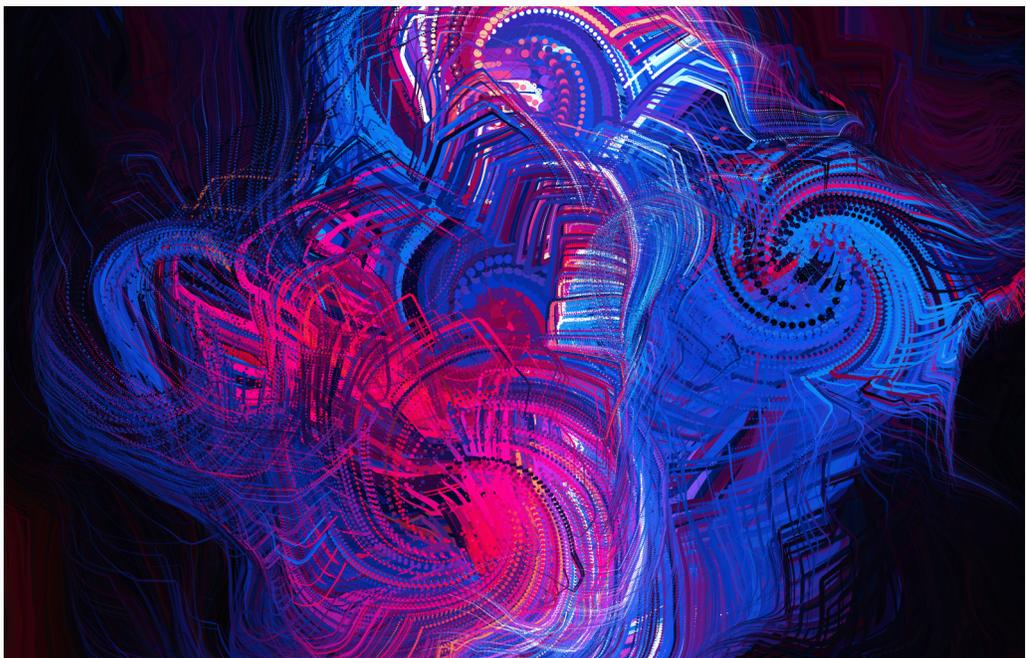
Alle Ellipsen wachsen gleich-  
mäßig bis zum 30. Frame an  
und bewegen sich dann

Die Ellipsen werden durch  
das noise Strömungsfeld  
bewegt. Dieses hängt von  
Faktoren ab, die manuell  
gesteuert werden können.  
Beispielsweise die Schnellig-  
keit oder die Turbulenz, mit  
der sich das Muster ent-  
wickelt.

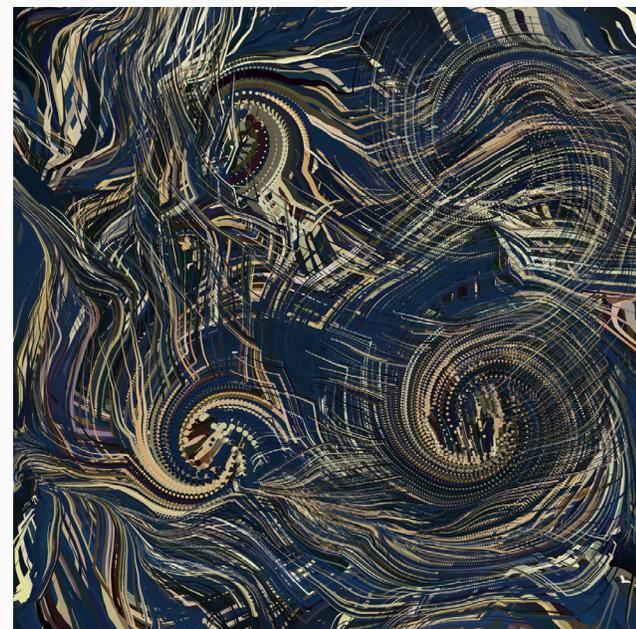
Mit der Maus kann man  
beeinflussen wie sich das  
Muster verändert, indem  
der Abstand zwischen dem  
Mauszeiger und allen Partik-  
eln berechnet wird und sich  
die Ellipsen anfangen spiral-  
förmig vom Mauszeiger weg  
zu drehen

mit der Zeit werden die El-  
lipsen immer um den Faktor  
0.996 kleiner

## Marble Effekt mit mouse Interaktion



## ERGEBNISSE



## Marble Effekt mit Ellipsen

```
var img;
var status = 0;
var myArray = [];
var noiseSize = 0.005;
```

```
function preload() {
  img = loadImage('data/ArtDecoPattern.jpg');
}
```

```
function setup() {
  createCanvas(800, 800);
  frameRate(60);
  img.resize(width, height);
  image(img, 0, 0);
  for (var i = 0; i < 10000; i++) {
    myArray.push(new particle());
    //noiseSeed(6759523);
  }
}
```

```
function draw() {
  if (status == 1) {
    for (var i = 0; i < myArray.length; i++) {
      myArray[i].zeigen();
      myArray[i].move();
    }
  } else {
    for (var j = 0; j < myArray.length; j++) {
      myArray[j].zeigen();
      myArray[j].move2();
    }
  }
  if (frameCount == 30) {
    status = 1;
  }
}
```

```
class particle {
  constructor() {
    this.px = random(width);
    this.py = random(height);
    this.max = random(1, 20);
    this.r = 3;
    this.c = color(img.get(this.px, this.py));
  }
}
```

## ERKLÄRUNG DES CODES

ähnlicher Code wie eine  
Seite weiter vorne

Das Foto wird geladen

Ich erstelle eine 800px mal  
600 px große  
„Leinwand“

Ich lege eine FrameRate von  
20 fest

10000 „Partikel“ werden  
erstellt

Die Funktionen „zeigen“ und  
„move“ werden ausgeführt

Die Funktionen move2 lässt  
die Ellipsen zuerst bis zum  
30. Frame anwachsen, dann  
werden die Funktionen  
„move2“ und wieder „zeigen“  
ausgeführt

Die „Partikel“ werden auf  
der gesamten Leinwand zu-  
fällig verteilt

sie bekommen eine An-  
fangsgröße von 3px und  
wachsen dann zufällig ab  
einer Größe zwischen 1px  
und 20px an

und nehmen die Farbe an,  
die der Pixel in der Mitte der  
Ellipse hat

## ERKLÄRUNG DES CODES

```
zeigen() {
  noStroke();
  fill(this.c);
  ellipse(this.px, this.py, this.r, this.r);
}

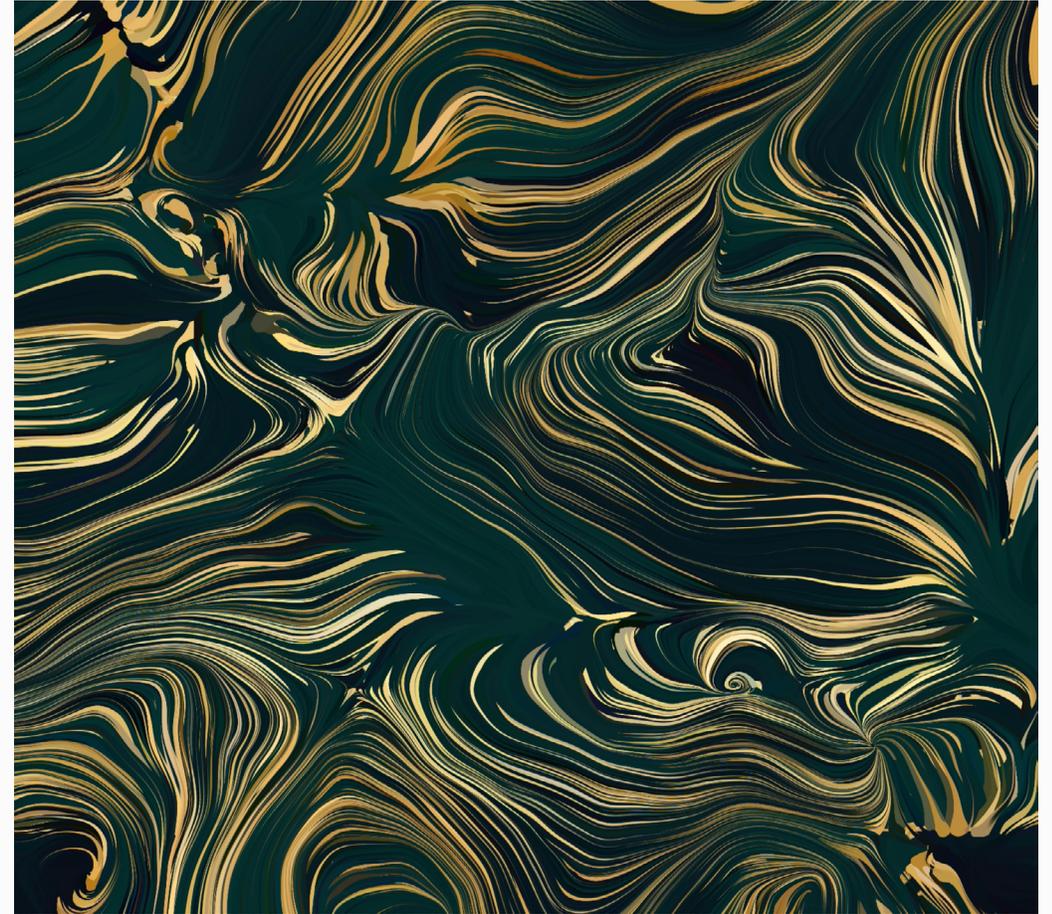
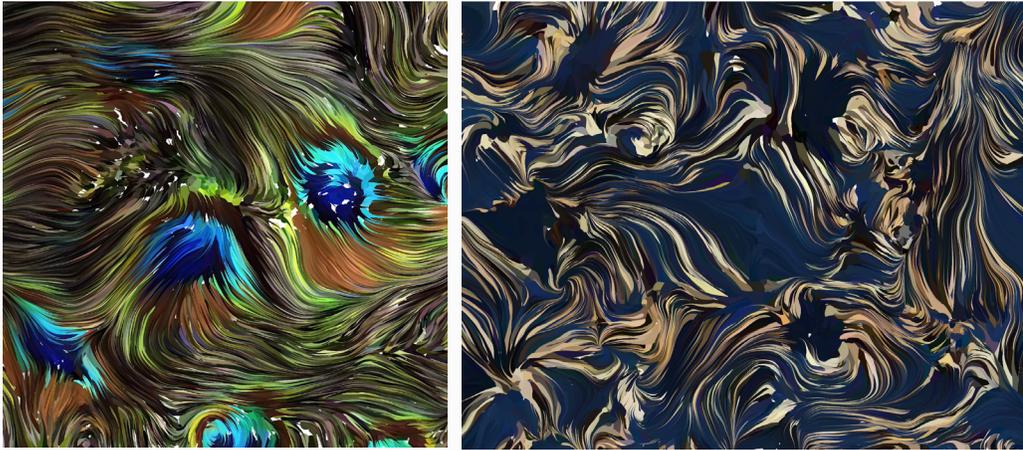
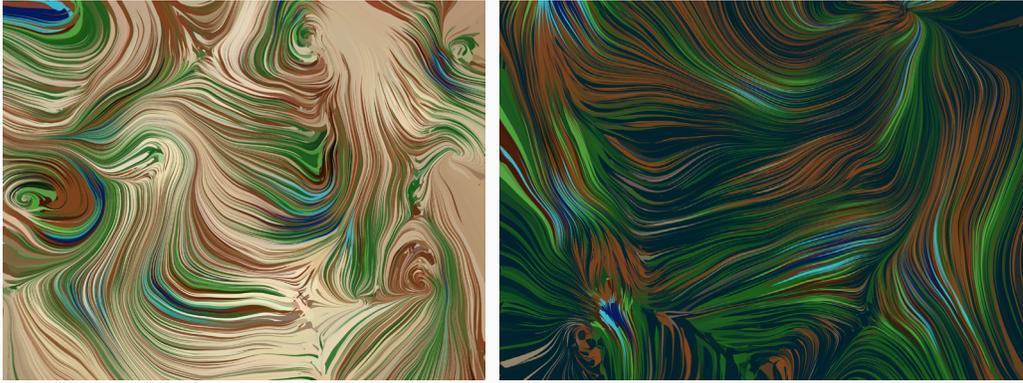
move2() {
  this.r = frameCount * (this.max - 3.0) / 30.0
+ 3;
}

move() {
  this.px += (noise(this.px * noiseSize,
this.py * noiseSize) - 0.5) * 3.0;
  this.py += (noise(this.px * noiseSize + 10,
this.py * noiseSize + 10) - 0.5) * 3.0;
  this.r *= 0.99;
}
```

Alle Ellipsen wachsen gleich-  
mäßig bis zum 30. Frame an  
und bewegen sich dann

Die Ellipsen werden durch  
das noise Strömungsfeld  
bewegt. Dieses hängt von  
Faktoren ab, die manuell  
gesteuert werden können.  
Beispielsweise die Schnellig-  
keit oder die Turbulenz, mit  
der sich das Muster ent-  
wickelt.

mit der Zeit werden die El-  
lipsen immer um den Faktor  
0.99 kleiner



**finales Key Visual „Golden Twenties“**



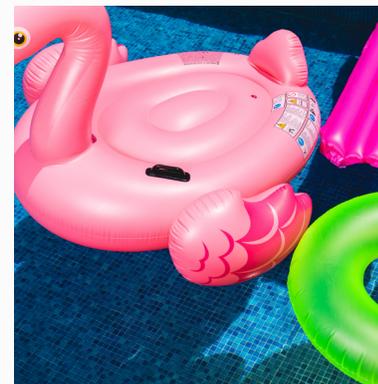
Ursprungsbild



Weitere Varianten die während dem Ausprobieren entstanden bevor ich mich für das finale Design rechts entschied.



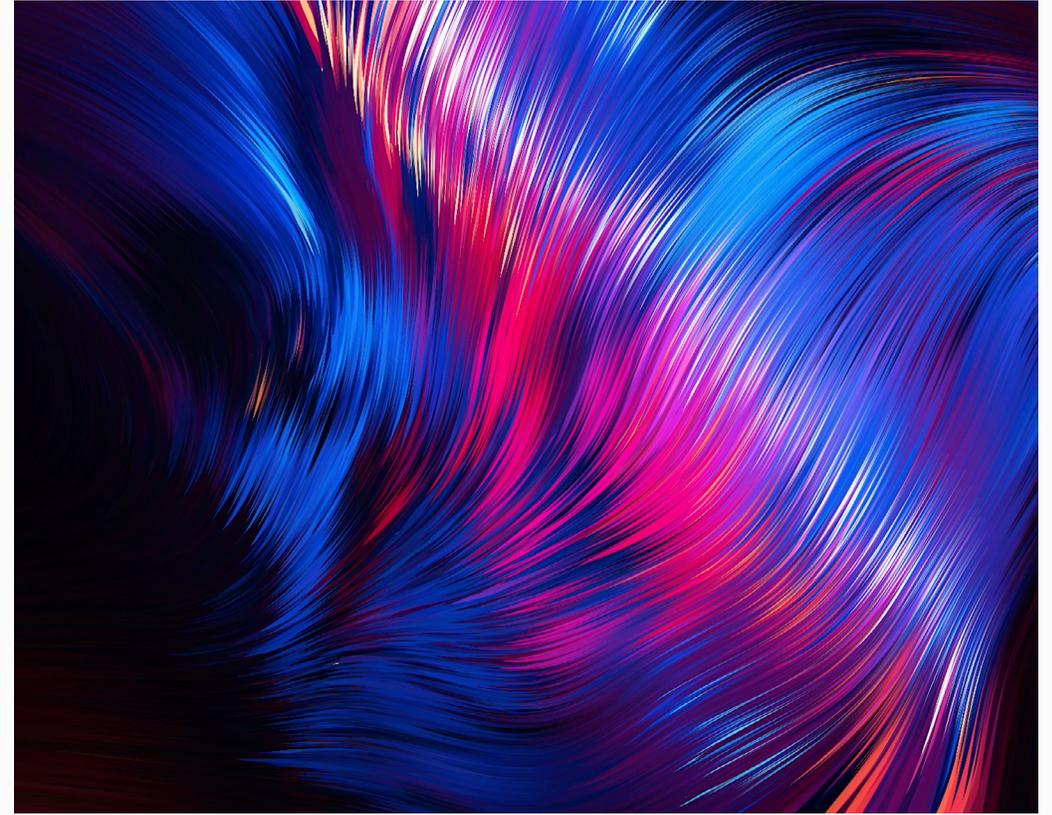
**finale Key Visual „summer vibes“**



Ursprungsbild



Bei diesem motto war ich mir sehr schnell sicher, welches Bild sich gut eignen würde, denn das Bild hat sehr viele Kontraste. Sowohl Farbkontraste als auch Helligkeitskontraste.



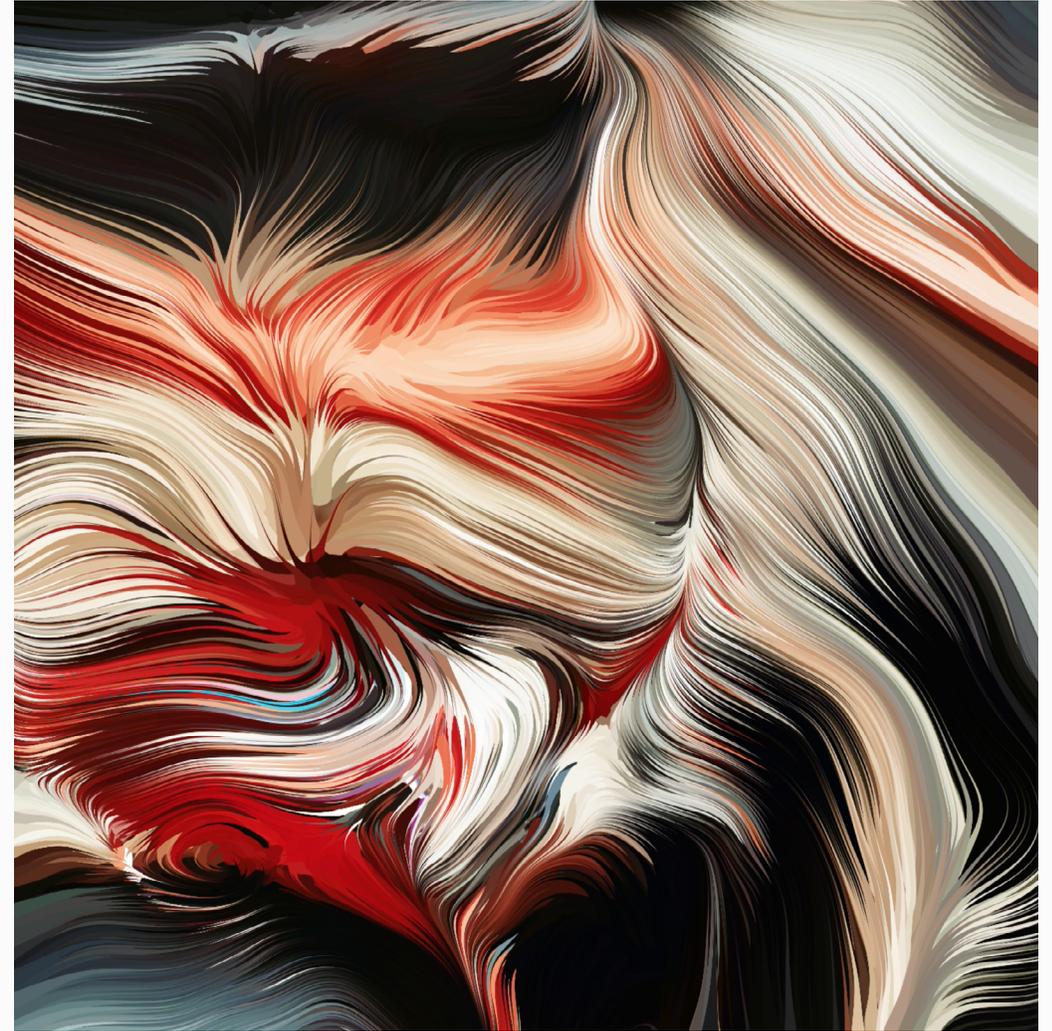
**finales Key Visual „Electro/Techno“**



Ursprungsbild

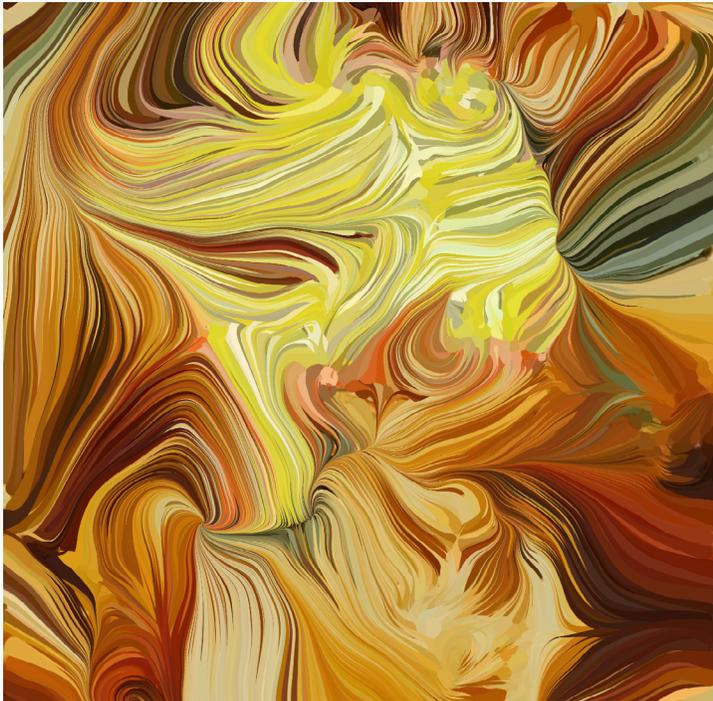


Alle Muster die mich an die Fünziger erinnerten und die ich ausprobierte, sahen leider nicht gut aus, denn sie hatten meistens zu wenige Farben und die Visuals waren langweilig. Dieses Key Visual basiert auf einem Foto eines Lenkrads.

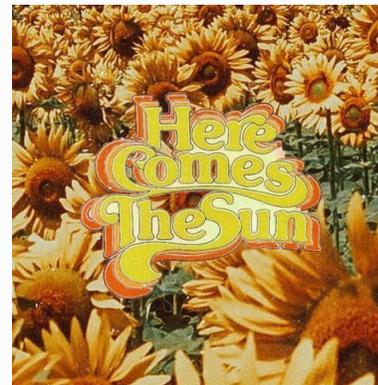


Ursprungsbild

**finales Key Visual „Rocking 50s“**



**finales Key Visual „70s Disco Fever“**



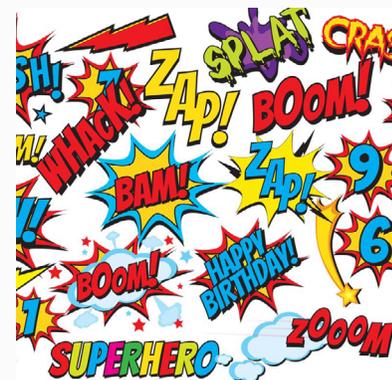
Ursprungsbild



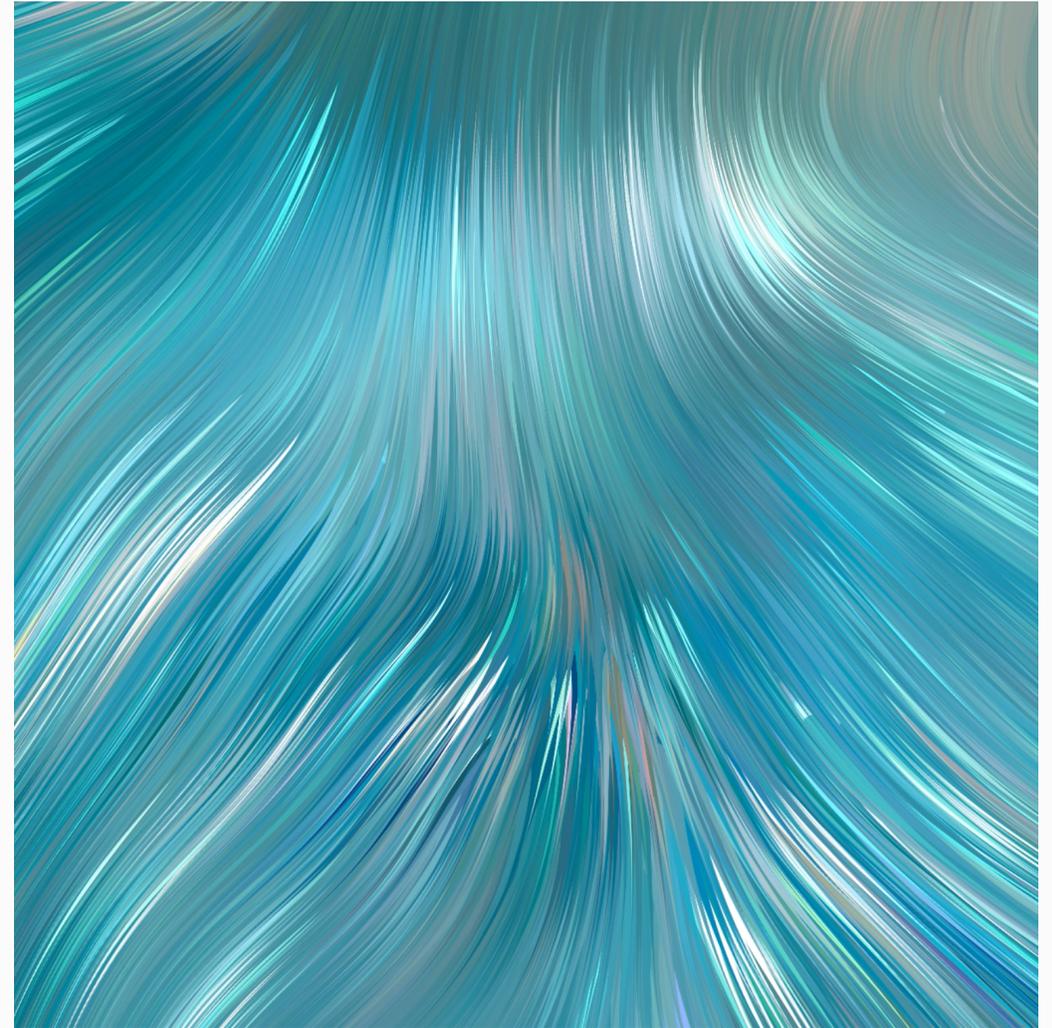
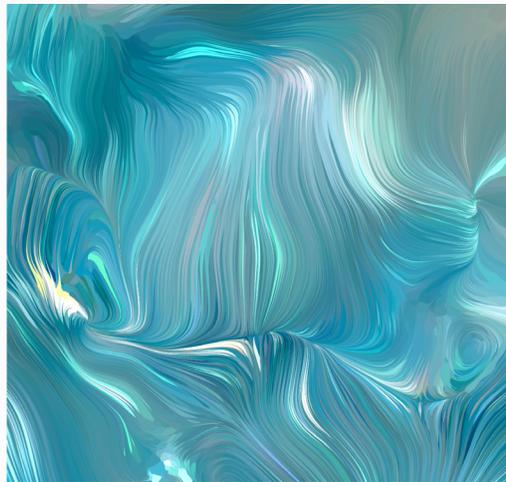
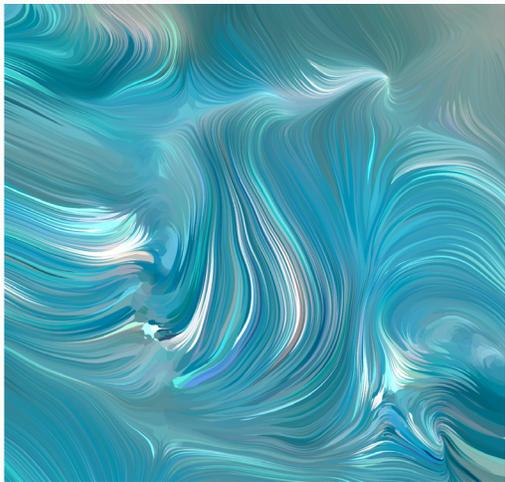
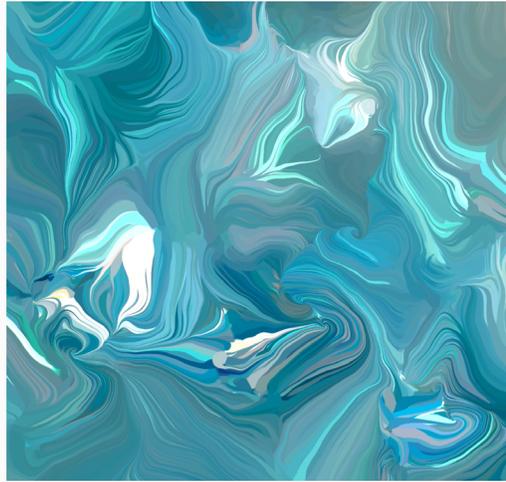
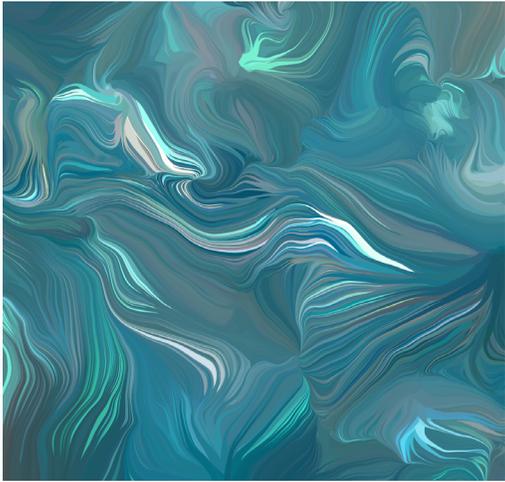
Ein passendes Moodbild für das Superhelden Key Visual zu finden war nicht ganz leicht. Mir fielen zwar Farben ein die diese Welt für mich widerspiegeln, aber ich fand kaum passenden Bilder im Internet. Mein finales Key Visual basiert auf einem Bild, welches sich stark von den anderen Ursprungsbildern unterscheidet, denn es ist kein Foto sondern eher eine Illustration im Comic-Stil.



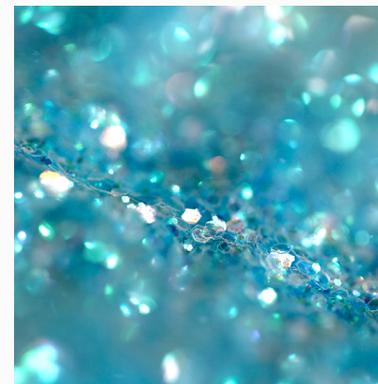
**finales Key Visual „superhero“**



Ursprungsbild



**finales Key Visual „Mermaid“**



Ursprungsbild



**finales Key Visual „Winter Wonderland“**



Ursprungsbild



**finales Key Visual „1001 Nacht“**



Ursprungsbild



**finales Key Visual „Fiesta mexicana“**



Ursprungsbild



## Erster Logo Entwurf zu „ID“

```
let myFont;

function preload() {
  myFont = loadFont('assets/roboto-black.ttf');
  myFont2 = loadFont('assets/roboto-semiB.ttf');
  luftschlange = loadImage('data/luftschlange.png');
}

var myArray = [];
var status = 0;
var subTitel = ['events', 'planning', 'events &
planning', 'nights', 'moments', 'celebrations']

var luftschlange;

function setup() {
  createCanvas(600, 600);

  for (var i = 0; i < 5; i++) {
    myArray.push(new element(int(random(1, 7))));
  }

  luftschlange.resize(width / 20, height / 10);
}

function draw() {
  background(255);

  schriftZeigen();
  for (var i = 0; i < myArray.length; i++) {
    myArray[i].zeigen();
  }
}

function mousePressed() {
  ursprung2();
  status++;
  if (status==7) {
    status=0;
  }
}

function ursprung2() {
  for (var j = 0; j < myArray.length; j++) {
    myArray[j] = new element(int(random(1, 7)));
  }
}
```

## ERKLÄRUNG DES CODES

Um Schriften verwenden zu können, muss man sie hochladen und vorladen

Laden eines Luftschlangen Icons

Ich lege fest welche Untertitel unter dem Logo stehen

Ich erstelle eine 600px mal 600 px große „Leinwand“

in jedem „Durchgang“ werden 5 Elemente ausgesucht und gezeichnet

Die Unterzeile wird angezeigt

bei jedem Klicken mit der Maus ändert sich der Status, d.h. ein neues Design wird generiert

er werden neue Elemente erstellt

## ERKLÄRUNG DES CODES

```
function schriftZeigen() {
  noStroke();
  fill('#2b2985');
  textAlign(CENTER, CENTER);
  textFont(myFont);
  textSize(width * 0.5);
  text('ID', width / 2.0, height * 0.42);
  textSize(width * 0.08);
  textFont(myFont2);
  fill(0);
  text(subTitel[status], width / 2, height * 0.7);
}
```

hier lege ich fest welche (typografischen) Eigenschaften die Schrift der Unterzeile hat und wo sie steht

```
class element {
  constructor(art) {
    this.px = random(height / 4, width - width / 4);
    this.py = random(height / 3, height * 0.62);
    this.angle = random(2 * PI);
    this.art = art;
  }
}
```

Der constructor sucht sich in jedes Mal zufällig 5 neue Elemente aus den Grafiken aus und platziert diese zufällig (in einem eingegrenzten Bereich)

```
zeigen() {
  if (this.art == 1) {
    this.rechteckZeigen();
  }

  if (this.art == 2) {
    this.ellipseZeigen();
  }

  if (this.art == 3) {
    this.dreieckZeigen();
  }

  if (this.art == 4) {
    this.fireworkZeigen();
  }

  if (this.art == 5) {
    this.luftschlangeZeigen();
  }

  if (this.art == 6) {
    this.punkteZeigen();
  }
}
```

## erster Logo Entwurf zu „ID“ (Fortsetzung)

```

punkteZeigen() {
  noStroke();
  fill('#b5aae9');
  ellipse(this.px, this.py, width / 20, height /
20);
}

luftschlangeZeigen() {
  image(luftschlange, this.px -
luftschlange.width, this.py - luftschlange.height /
2);
  image(luftschlange, this.px, this.py);
}

rechteckZeigen() {
  noStroke();
  fill('#78a3c7');
  push();
  translate(this.px, this.py);
  rotate(this.angle);
  rect(-width * 0.05, -height * 0.025, width *
0.1, height * 0.05);
  pop();
}

ellipseZeigen() {
  fill('#db4973');
  noStroke();
  ellipse(this.px, this.py, width / 8, height /
7);
}

dreieckZeigen() {
  fill('#70c9d9');
  noStroke();
  triangle(this.px - width / 20, this.py + height
/ 26,
  this.px + width / 20, this.py + height / 26,
  this.px, this.py - width / 14);
}

fireworkZeigen() {
  stroke('#ff8c46');
  strokeWeight(5);
  strokeCap(ROUND);

  for (var i = 0; i <= 7; i++) {
    push();
    translate(this.px, this.py);
    rotate(i * (2 * PI) / 7.0);
    line(width * 0.025, 0, width * 0.05, 0);
    pop();
  }
}

```

48

## ERKLÄRUNG DES CODES

Größe und Farbe der Ellipse  
werden festgelegt, die Posi-  
tion ist zufällig

das Luftschlangen Icon wird  
geladen, seine Position ist  
zufällig

Größe und Farbe des Recht-  
ecks werden festgelegt und  
dann zufällig positioniert  
und rotiert

Größe und Farbe der Ellipse  
werden festgelegt, die Posi-  
tion ist zufällig

die Farbe und alle drei Eck-  
punkte des Dreiecks werden  
festgelegt

das Feuerwerk Icon wird  
aus 7 Linien erstellt  
durch die Rotation zeigen  
sie alle in die Mitte und  
bilden eine Art Stern

## ERGEBNISSE



49

## Logo Entwurf „Konfetti“

```
var myArray = [];  
var status = 2;  
var konfettiArray = [];
```

```
function setup() {  
  createCanvas(900, 600);  
  smooth(8);  
  for (var j = 0; j < 40; j++) {  
    //konfetti Anzahl  
    //einstellen  
    konfettiArray.push(new konfetti());  
  }  
}
```

```
function draw() {  
  background('#2b2985');  
  for (var j = 0; j < konfettiArray.length; j++) {  
    konfettiArray[j].konfettiZeigen();  
    konfettiArray[j].konfettiBewegen();  
  }  
  
  for (var i = 0; i < myArray.length; i++) {  
    myArray[i].zeigen();  
  }  
}
```

```
class konfetti {  
  constructor() {  
    this.px = random(height / 4, width - width / 4);  
    this.py = random(height);  
    this.winkel = random(2 * PI);  
    this.farbe =  
    color(random(50, 255), random(50, 255), random(50, 255));  
    this.groesse = random(20, 40);  
    this.speed = random(0.5, 2);  
  }  
  
  konfettiZeigen() {  
    push();  
    noStroke();  
    fill(this.farbe);  
    translate(this.px, this.py);  
    rotate(this.winkel);  
    rect(0, 0, width * 0.01 * this.groesse, height *  
    0.04 * this.groesse);  
    pop();  
  }  
  
  konfettiBewegen() {
```

```
    this.winkel += 0.01;  
    this.py += this.speed;  
    if (this.py > height) {  
      this.py = 0;  
    }  
  }  
}
```

```
    this.winkel += 0.01;  
    this.py += this.speed;  
    if (this.py > height) {  
      this.py = 0;  
    }  
  }  
}
```

## ERKLÄRUNG DES CODES

Ich erstelle eine 900px mal  
600 px große „Leinwand“

40 Rechtecke („Konfetti“)  
werden erstellt

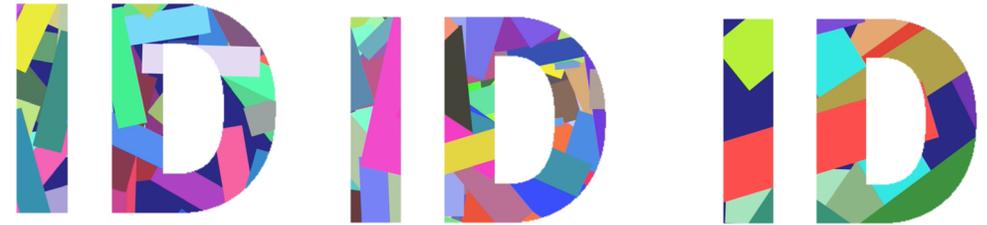
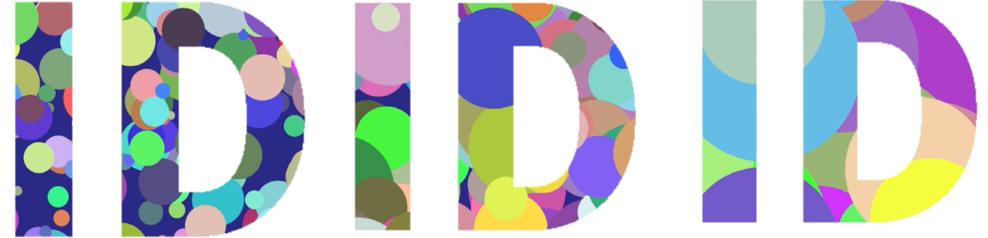
die Funktionen „konfettiZeigen“  
und „konfettiBewegen“  
werden ausgeführt

der constructor erstellt  
alle Konfetti mit zufälliger  
Position, zufälligem Anfangs-  
winkel, einer Größe zwischen  
20 und 40 px, einer zufäl-  
ligen Geschwindigkeit und  
zufälliger Farbe (in einem  
bestimmten Farbspektrum,  
damit die Rechtecke nicht  
zu dunkel werden)

das Konfetti wird angezeigt

das Konfetti fällt nach  
unten und wird wieder oben  
neu gebildet wenn es unten  
angekommen ist

## ERGEBNISSE



## Bewegte Rechtecke als Hintergrund für das generative Logo

```
var myArray = [];
var status = 2;
var konfettiArray = [];
```

```
function setup() {
  createCanvas(900, 600);
  smooth(8);
  for (var j = 0; j < 40; j++) { //konfetti Anzahl
    einstellen
    konfettiArray.push(new konfetti());
  }
}
```

```
function draw() {
  background('#2b2985');
  for (var j = 0; j < konfettiArray.length; j++) {
    konfettiArray[j].konfettiZeigen();
    konfettiArray[j].konfettiBewegen();
  }

  for (var i = 0; i < myArray.length; i++) {
    myArray[i].zeigen();
  }
}
```

```
class konfetti {
  constructor() {
    this.px = random(height / 4, width - width / 4);
    this.py = random(height);
    this.winkel = random(2 * PI);
    this.farbe =
    color(random(50,255),random(50,255),random(50,255));
    this.groesse = random(20,40);
    this.speed = random(0.5,2);
  }
}
```

```
konfettiZeigen() {
  push();
  noStroke();
  fill(this.farbe);
  translate(this.px, this.py);
  rotate(this.winkel);
  rect(0,0, width * 0.01 * this.groesse, height *
  0.04 * this.groesse);
  pop();
}
```

```
konfettiBewegen() {
  this.winkel += 0.01;
  this.py += this.speed;
  if (this.py > height) {
    this.py = 0;
  }
}
```

## ERKLÄRUNG DES CODES

**siehe eine Seite weiter vorne**

Ich erstelle eine 900px mal 600 px große „Leinwand“

40 Rechtecke („Konfetti“) werden erstellt

die Funktionen „konfettiZeigen“ und „konfettiBewegen“ werden ausgeführt

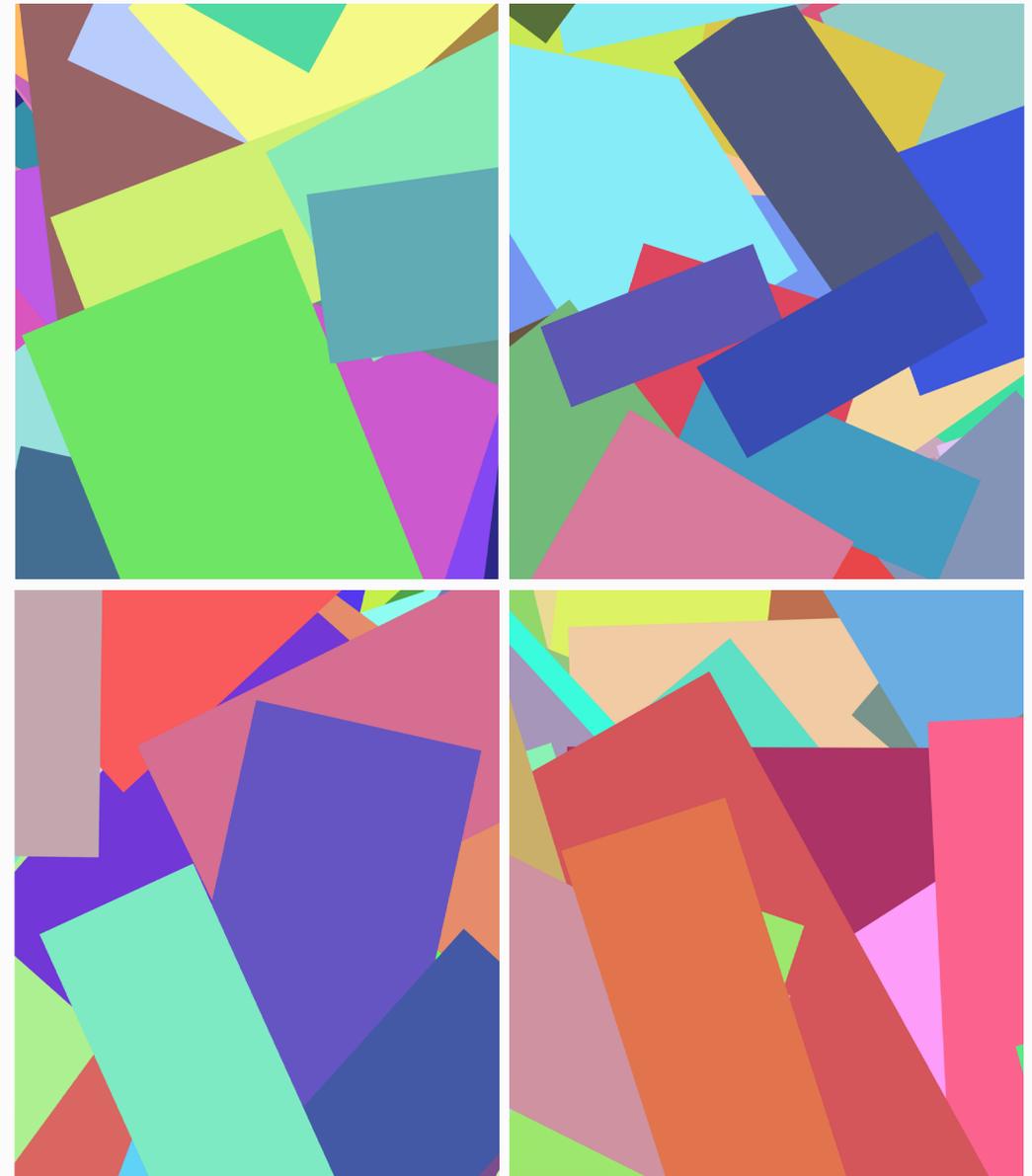
der constructor erstellt alle Konfetti mit zufälliger Position, zufälligem Anfangswinkel, einer Größe zwischen 20 und 40 px, einer zufälligen Geschwindigkeit und zufälliger Farbe (in einem bestimmten Farbspektrum, damit die Rechtecke nicht zu dunkel werden)

das Konfetti wird angezeigt

Nur durch das Ändern der Größe der Rechtecke entstanden die Hintergründe rechts

das Konfetti fällt nach unten und wird wieder oben neu gebildet wenn es unten angekommen ist

## ERGEBNISSE



## Finaler Logo Entwurf joy

```
var myArray = [];  
var status = 2;  
var konfettiArray = [];  
let joySchablone;  
  
function preload() {  
  joySchablone = loadImage('data/joy-  
Schablone.png');  
}  
  
function setup() {  
  createCanvas(900, 800);  
  smooth(8);  
  for (var j = 0; j < 40; j++) { //konfetti  
Anzahl einstellen  
    konfettiArray.push(new konfetti());  
  }  
}  
  
function draw() {  
  background('#2b2985');  
  for (var j = 0; j < konfettiArray.length;  
j++) {  
    konfettiArray[j].konfettiZeigen();  
    konfettiArray[j].konfettiBewegen();  
  }  
  
  for (var i = 0; i < myArray.length; i++) {  
    myArray[i].zeigen();  
  }  
  
  image(joySchablone,0,0);  
  for (var s = 0; s < myArray.length; s++) {  
    myArray[s].zeigen();  
  }  
}
```

## ERKLÄRUNG DES CODES

ähnlicher Code,  
siehe eine Seite  
weiter vorne

Bild einer Schablone des  
Logos „joy“ wird geladen

die Schablone des Logos  
wird drauf gelegt und  
dahinter bewegt sich der  
Hintergrund

## ERKLÄRUNG DES CODES

ähnlicher Code,  
siehe eine Seite  
weiter vorne

```
class konfetti {  
  constructor() {  
    this.px = random(height / 4, width - width  
/ 4);  
    this.py = random(height);  
    this.winkel = random(2 * PI);  
    this.farbe = color(random(50, 255),  
random(50, 255), random(50, 255));  
    this.groesse = random(20, 40);  
    this.speed = random(0.5, 2);  
  }  
  
  konfettiZeigen() {  
    push();  
    noStroke();  
    fill(this.farbe);  
    translate(this.px, this.py);  
    rotate(this.winkel);  
    rect(0, 0, width * 0.01 * this.groesse,  
height * 0.04 * this.groesse);  
    pop();  
  }  
  
  konfettiBewegen() {  
    this.winkel += 0.01;  
    this.py += this.speed;  
    if (this.py > height) {  
      this.py = 0;  
    }  
  }  
}
```



ERGEBNISSE



