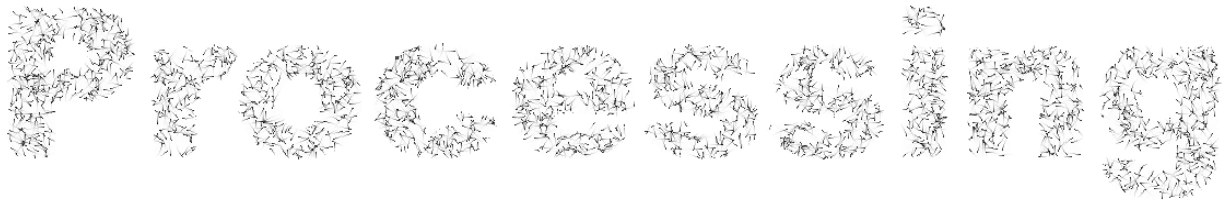
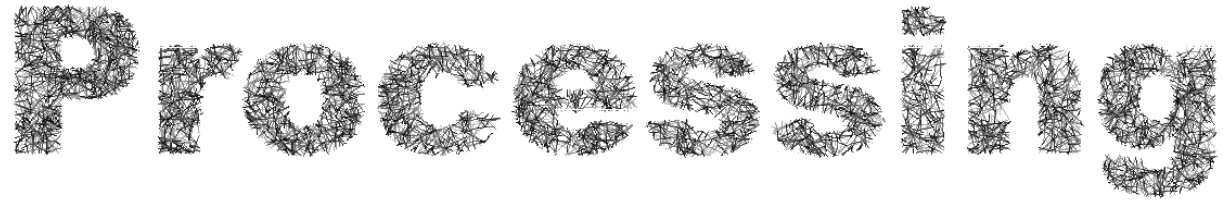
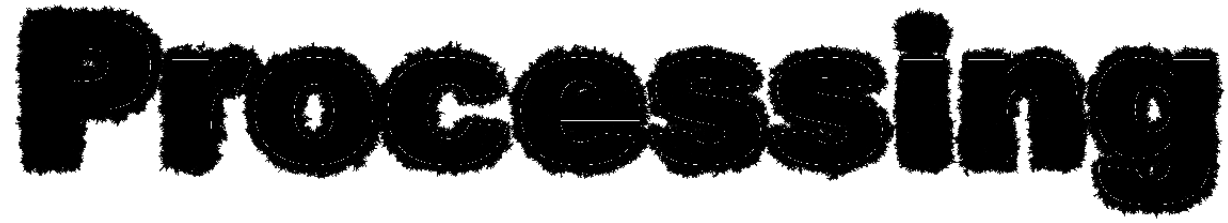






Computerkunst mit Processing

Eine Einführung zum Erstellen von Grafiken mit Processing.

Speaker: Patrick Hess / Com2u

Processing ist eine an JAVA angelehnte Programmiersprache in der man schnell Visualisierungen erstellen kann. Das wollen wir im Workshop tun.

Ich biete einen Workshop zu Processing an.
Dabei zeige ich wie man Processing bekommt, installiert und wo man gute Tutorials findet.
An ein paar Beispielen zeige ich die Grundlegende Programmierung mit Processing.
Dann machen wir alle zusammen ein keines Programmierprojekt um eine Computergrafik/Kunst zu erstellen.
Im Anschluss haben die Teilnehmer Zeit für Fragen und um das Beispiel selbst zu erweitern.
Eigener Laptop notwendig. (Laptop, Netzteil, Adminrechte)
Keine Programmierkenntnisse notwendig. JAVA Kenntnisse hilfreich.
Wer Processing schon installiert hat und im Club WLAN verbunden ist, hat es einfacher.

Bei Interesse kann danach frei weiter programmiert werden. Ich stehe für Fragen zur Verfügung. Evtl. gibt es später auch weitere Workshops im Club.

Wo finde ich Processing?

https://processing.org/download/

Welche Processing Tutorials sind gut?

https://processing.org/examples/ : Allgemeine Beispiele direkt von Processing. Animationen und Grafiken laufen meist als Vorschau auch im Browser.

https://www.openprocessing.org/ : Viele bunte Beispiele immer gleich mit einem Vorschaubild

http://www.generative-gestaltung.de/code : Beispiele aus einem Gestaltungsbuch

https://www.youtube.com/user/shiffman/playlists?sort=dd&shelf_id=6&view=50 : Video Tutorial zu Processing natürlichen Grafiken

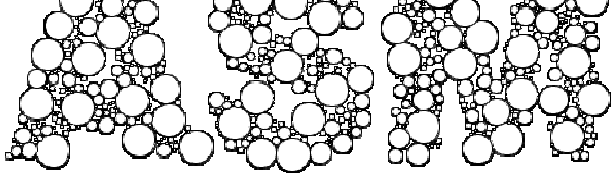
https://www.youtube.com/user/shiffman/playlists?shelf_id=2&view=50&sort=dd : Video Tutorial zu Processing Grundlagen

https://docs.google.com/viewer?a=v&pid=sites&srcid=cG91Z2hrZWVwc2llZGF5Lm9yZ3xhcnR0ZWNofGd4OjFkZDg2NTdlNmNhODk2MGE : Ebook zu Processing

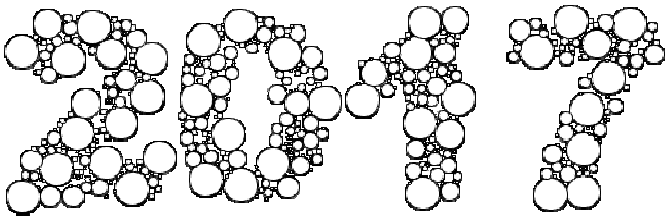
http://passthrough.fw-notify.net/static/908846/downloader.html

```
ArrayList<Shape>myShapes;
PImage typo;
int shapeCount=1000;
int maxSize=25;

class Shape {
  PVector position;
  int size;
  Shape (float _x, float _y, int _size) {
     position    = new PVector (_x, _y);
     size = _size;
  }

  void show() {
     ellipse(position.x, position.y, size, size);
  }
}

void setup() {
  size(500, 500);
  stroke(0);
  background(255);
  typo = loadImage("background.jpg");
  myShapes = new ArrayList<Shape>();
  float x, y;
  int size;
  color pixel;
  for (int i = 0; i < shapeCount; i++) {
    do {
      x = random(width);
      y = random(height);
      pixel = typo.get((int) x, (int) y);
      size = (int) nearestDistance((int) x, (int) y) * 2;
      // size= (int) random(maxSize);
    } while (brightness(pixel) > 90 || size < 2);
    if (size > maxSize) {
      size = maxSize;
    }
    myShapes.add(new Shape(x, y, size));
  }
}

void draw() {
  stroke(0);
  for (Shape s : myShapes) {
    s.show();
  }
}

float nearestDistance(int x, int y){
  float disance = 9999;
  float col=0;
  for (Shape s : myShapes) {
     col = dist(x, y, s.position.x, s.position.y)-s.size/2;
     if (col < disance){
       disance = col;
     }
  }
  return disance;
}
```

```
ArrayList<Shape>myShapes;
int shapeCount=1000;
int maxSize=25;

class Shape {
  PVector position;
  int size;

  Shape (float _x, float _y, int _size) {
      position    = new PVector (_x, _y);
      size = _size;
  }

  void show() {
      ellipse(position.x, position.y, size, size);
  }

}

void setup() {
  size(500, 500);
  stroke(0);
  background(255);
  myShapes = new ArrayList<Shape>();
  float x, y;
  for (int i = 0; i < shapeCount; i++) {
    x = random(width);
    y = random(height);
    myShapes.add(new Shape(x, y, maxSize));
  }
}

void draw() {
  stroke(0);
  for (Shape s : myShapes) {
    s.show();
  }
}
```



```
PImage typo;
```



```
void setup() {
  size(500, 500);
  stroke(0);
  background(255);
  typo = loadImage("background.jpg");
  myShapes = new ArrayList<Shape>();
  float x, y;
  int size;
  color pixel;
  for (int i = 0; i < shapeCount; i++) {
    do {
      x = random(width);
      y = random(height);
      pixel = typo.get((int) x, (int) y);
      size = (int) nearestDistance((int) x, (int) y) * 2;
      // size= (int) random(maxSize);
    } while (brightness(pixel) > 90 || size < 2);
    if (size > maxSize) {
      size = maxSize;
    }
    myShapes.add(new Shape(x, y, size));
  }
}
```

```
float nearestDistance(int x, int y){
  float disance = 9999;
  float col=0;
  for (Shape s : myShapes) {
      col = dist(x, y, s.position.x, s.position.y)-s.size/2;
      if (col < disance){
         disance = col;
      }
  }
  return disance;
}
```