



## Pointillismus

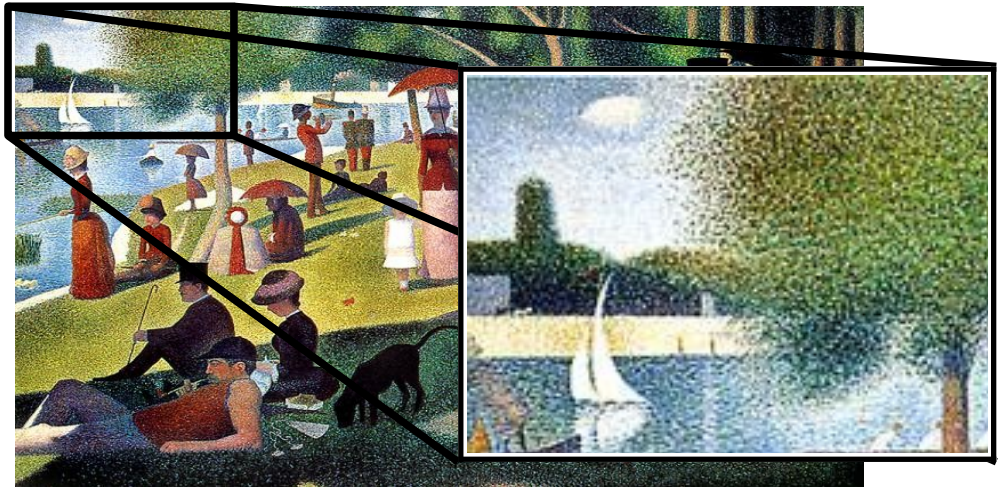
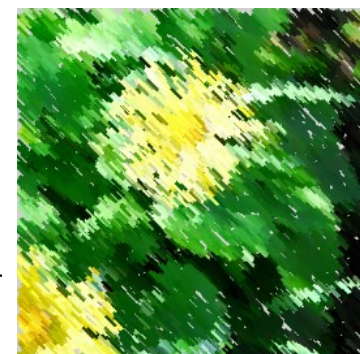


Abb. 1: Georges Seurat - Un dimanche après-midi à l'Île de la Grande Jatte  
Wikimedia Commons (Lizenz: Public Domain), URL:  
[https://commons.wikimedia.org/wiki/File:Georges\\_Seurat\\_-\\_Un\\_dimanche\\_apr%C3%A8s-midi\\_%C3%A0\\_l'%27%C3%8ELe\\_de\\_la\\_Grande\\_Jatte.jpg](https://commons.wikimedia.org/wiki/File:Georges_Seurat_-_Un_dimanche_apr%C3%A8s-midi_%C3%A0_l'%27%C3%8ELe_de_la_Grande_Jatte.jpg)

Pointillismus bezeichnet eine Stilrichtung in der Malerei. Sie hatte in den Jahren zwischen **1889** und **1910** ihre Blütezeit. Der Pointillismus wird dem Post-Impressionismus zugeordnet.

Typisch für den Pointillismus ist der streng geometrisch durchkomponierte, oft ornamental wirkende Bildaufbau. Die Farben werden dabei in kleinen Punkten auf die Leinwand gesetzt. Dabei nutzten pointillistische Maler nur reine Farben. Farbmischungen entstehen erst auf dem Malgrund selbst.

Man kann einen ähnlichen Effekt erreichen, wenn man in einem Bild zufällige Pixel auswählt und dann ausgefüllte Kreise einer bestimmten Größe an diese Stelle zeichnet.



Gleiches kann man statt mit Punkten auch mit kurzen schrägen Strichen machen. Auch hier erhält man schöne, künstlerische Effekte.

Dabei kann man die Anzahl und den Durchmesser der Punkte bzw. die Breite und Länge der Striche festlegen.



## Zufallszahlen in Java

Um die Pixel zufällig auszuwählen, benötigt man Zufallszahlen. In Java gibt es eine Klasse `Random`, um Zufallszahlen zu erzeugen. Möchte man Sie benutzen, muss man sie importieren und dann ein `Random`-Objekt erzeugen. Bei diesem gibt es verschiedene Methoden, um reelle oder ganzzahlige Zufallszahlen zu erzeugen.

An Anfang der Datei:

```
import java.util.Random;
```

In der Methode, in der Zufallszahlen benötigt werden:

```
Random zufallszahlen = new Random();
int x = zufallszahlen.nextInt(20);      // zwischen 0 und 19
double y = zufallszahlen.nextDouble(); // zwischen 0.0 und 0.9999
```

Dokumentation: <https://docs.oracle.com/javase/8/docs/api/java/util/Random.html>

## Zeichnen mit der IMP-Klasse `Picture`

Um auf einem `Picture`-Objekt zu zeichnen, muss der Name des Objektes genannt und dann die entsprechende Methode aufgerufen werden:


```
objektname.methodenname(...) // Punktoperator
```

Zum Beispiel:

```
Picture meinBild = new Picture(300,200);
meinBild.stroke(250,0,0); // Stiftfarbe rot
meinBild.strokeWeight(3); // Stiftbreite 3
meinBild.line(10,10,30,10); // Linie von (10,10) nach (30,10)
meinBild.noStroke(); // kein Stift/Rand
meinBild.fill(0,250,0); // Füllfarbe grün
meinBild.ellipse(20,20,10,10); // Kreisfläche an (20,20) mit d=10
```

## Aufgabe

1. Erstelle in BlueJ eine Klasse `Kunst`. Dekлариere dort die Methode

 `public Picture tupfen(Picture originalbild, int groesse, int anzahl)`

Lies die Pixel aus dem Originalbild aus. Erzeuge ein neues Bild von der Größe des Originalbildes.

Wähle eine zufällige x-Koordinate und eine zufällige y-Koordinate auf dem Bild aus. Setze die Füllfarbe auf die Farbe des Pixels im Originalbild und zeichne im neuen Bild eine Kreisfläche (ohne Rand) an der ausgewählten Stelle mit dem Durchmesser `groesse`.

Wiederhole das in einer Schleife `anzahl` mal.

Teste deine Methode.

2. Dekлариere analog eine Methode `stricheln` mit den Parametern `originalbild`, `dicke`, `laenge` und `anzahl`.






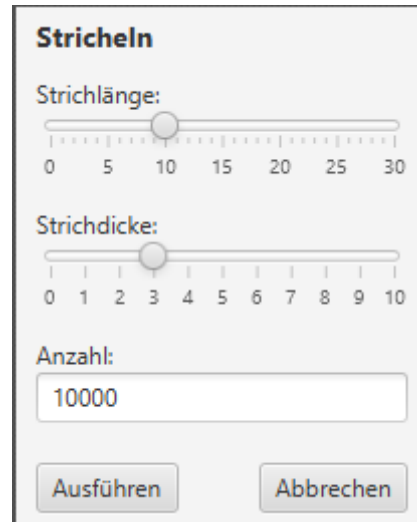
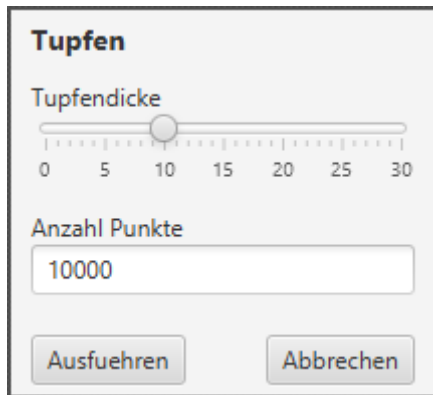
Zeichne die gewünschte Anzahl von Strichen auf das neue Bild. Wähle auch hier die Positionen zufällig und benutze die Farbe des Originalbildes an dieser Stelle. Die Stiftbreite muss auf `dicke` eingestellt werden.


Es sieht schöner aus, wenn die Striche schräg sind. Überlege dir, wie du den Endpunkt einer schrägen Strecke berechnen kannst, deren Länge über den Parameter verändert werden kann.

## ERWEITERUNG: STRICHELN / TUPFEN



3. Füge in der GUI im Menü Kunst zwei neue Menüpunkte hinzu. Führe in den dazugehörigen Action-Methoden zunächst die Methoden tupfen bzw. stricheln mit von dir gewählten Parameterwerten aus.  
  

4. Erstelle je ein Eingabefenster für die Parameter des Strichelns bzw. des Tupfens. Verwende dabei für die Buttons die gleichen Action-Methoden wie bei der Farbanpassung, so dass die Action-Methoden weiter verwendet werden können.  




-  Binde diese in die GUI ein, so dass bei jeder Änderung der Parameterwerte das Bild angepasst wird.