# **Unterrichtsstunde Animation und Interaktion**

## Übersicht:

Bisher kenn+n die Schüler Möglichkeiten, statische Programme zu erstellen, die einmalig auf das Sketch-Fenster zeichnen können. In dieser Unterrichtsstunde soll den Schülern gezeigt werden, wie sich Sketches animieren lassen und wie man interaktive Sketches erstellen lassen.

## Lernziel:

Nach der Stunde sollte jede/r das Konzept von draw- und setup-Funktionen verstanden haben, und wie sich mit diesen animierte Programme erstellen lassen. Als Minimalziel sollten alle Schüler am Ende der Stunde ein animiertes Programm geschrieben haben.

## Zeitplan:

1. Einführung Setup/Draw (5 min)
2. Erste Animation (5 min)
3. Übungsaufgaben (35 min)
4. Erstes Interaktives Programm (10 min)
5. Übungsaufgaben Rest (35 min)

## Theorie:

### 1. Einführung Setup/Draw *(introduction.pde)*

* Einführung der Funktionen setup und draw:   
  In **Setup** kommt der Code, der ein einziges Mal **am Anfang** des Programms ausgeführt werden soll. In **Draw** kommt der Code, der **immer wieder** (60 mal pro Sekunde) ausgeführt werden soll.  
  Setup = Initialisieren, Draw = Animieren
* Ab jetzt sollten Befehle für Programme immer entweder in setup in draw stehen!
* Anmerkung: Der Syntax von Funktionen (void, etc…) ist noch nicht bekannt, darauf wird erst nächste Stunde eingegangen.

### 2. Erste Animation *(firstAnimation.pde)*

* Vergleich mit einem Film: jedes mal wenn draw ausgeführt wird, ist es wie ein Frame in einem Film: wenn es jedes Mal das gleiche ist, herrscht Standbild
* Damit sich etwas bewegt, müssen in draw immer zwei Dinge passieren:

1. Es muss etwas gezeichnet werden
2. Es muss etwas bewegt werden

-> am Beispiel *firstAnimation.pde* vorprogrammieren

* **Frage:** Wo kommt der background-Aufruf hin? In draw, oder in setup?
* **Antwort:** Je nachdem: Wenn **background in setup** steht, wird es nur einmal ausgeführt, heißt alles vom alten draw bleibt stehen. Wenn **background in draw** steht, wird der Sketch vor jedem draw quasi zurückgesetzt.Je nach Anwendungsfall will man manchmal das eine, manchmal das andere.
* Übungsaufgaben 1 und 2

### 3. Erstes interaktives Programm *(firstInteractive.pde)*

* Jetzt wo man weiß, wie man Programme animiert, kann man die Animation abhängig vom Benutzer machen!
* Processing bietet viele Variablen, mit denen man Benutzereingaben machen kann:
* mouseX und mouseY für Mausposition
* key für die letzte Taste, die gedrückt wurde
* width und height für die Breite/Höhe des Sketches
* Zusätzlich gibt es neben draw und setup noch weitere Funktionen, die Processing dynamisch aufruft, z.B.:
* mousePressed()
* keyPressed()
* Übungsaufgaben Rest

## Übungen:

(Teil 1: Animation)

### Übung 1:

Erstelle deine erste eigene Animation!

a) Animiere einen Kreis, der von oben nach unten fällt.  
b) Versuche auch, die Geschwindigkeit, mit der der Kreis fällt zu animieren.

c) Bonus: Programmiere, dass der Kreis liegen bleibt wenn er die untere Kante erreicht hat!

### Übung 2:

Animiere Farben: Zeichne in deinem Sketch ein Rechteck, das zuerst schwarz ist und mit der Zeit immer heller wird. *(Tipp:* fill(int); füllt eine Form aus).

(Teil 2: Interaktion)

### Übung 3:

Zeichne einen Kreis, der immer in der Mitte des Sketches ist, egal bei welcher Fenstergröße. (*Tipp*: Schreibe den Befehl surface.setResizable(true); in setup, damit sich die Fenstergröße des Sketches verstellen lässt!)

### Übung 4:

a) Erstelle ein mini-Paint-Programm, das eine Linie hinter deiner Maus zeichnet. (*Tipp:* die Variablen pmouseX und pmouseY speichern die vorherige Position des Mauszeigers ab. Das p steht für “previous”)

b) Erweiterte das Programm so, dass nur gezeichnet wird, wenn die Maus gedrückt ist.