# Unterrichtsstunde Funktionen

## Übersicht:

In dieser Unterrichtsstunde soll den Schülern das Konzept von Funktionen beigebracht werden. Es wird beigebracht, was Funktionen sind, wie Funktionen definiert werden, was Rückgabewerte sind und worfür Funktionen verwendet werden können.

## Lernziel:

Nach der Stunde sollte jede/r eine eigene Funktion schreiben und aufrufen können. Konzepte wie Rückgabewert und Parameter sollten verstanden sein. Zusätzlich sollten Schüler verstehen, was Funktionen für Vorteile bieten um ihren eigenen Code mit funktionen strukturieren zu können.

## Zeitplan:

1. Was sind Funktionen? (5min)

2. Arten von Funktionen (10min)

3. Übungen (35min)

4. Vorteile von Funktionen (10min)

5. Übungen (30min)

## Theorie:

### **1. Was sind Funktionen?**

* Erwähnung bereits bekannter Funktionen:

|  |
| --- |
| void setup() { *// "Definition"* }  ellipse(250, 250, 50, 50); *// "Aufruf"* |

* Funktionen sind **Blöcke von Code**, die **aufgerufen** werden können
* Der Befehl ellipse(...); ist ein Aufruf einer Funktion, die Processing für uns definiert hat. Setup ist eine spezielle Funktion, die von Processing am Anfang des Programms aufgerufen wird.
* Neben den eingebauten Funktionen wie ellipse oder rect können wir auch unsere eigenen Funktionen schreiben und aufrufen!
* eigene Funktionen müssen **in setup oder in draw aufgerufen** werden

### 2. Arten von Funktionen

* Einfache Funktion

|  |
| --- |
| void setup() {  blueRect(); }  void blueRect() {  fill(0, 0, 255);  rect(20, 20, 50, 50); } |

* Funktion mit Parametern

|  |
| --- |
| void setup() {  blueRect(20, 20); }  void blueRect(float x, float y) {  fill(0, 0, 255);  rect(x, y, 50, 50); } |

* Funktion mit Rückgabewert

|  |
| --- |
| void setup() {  int fuenfmalfuenf = quadrat(5);   print(fuenfmalfuenf); }  int quadrat(int x) {  return x \* x; } |

### Wichtig: Funktionen mit Rückgabewert können wie Werte verwendet werden!

* **Typ am Anfang**, **“return” am Ende**
* Können auch mehrere returns haben!

### 3. Vorteile von Funktionen

* **Lesbarkeit**

Codeblöcken können Namen gegeben werden, sodass man sich besser im Programm zurecht findet und schneller erkennt, welcher Programmcode was macht

* **Wiederverwendbarkeit**

Codeblöcke, die man immer wieder an verschiedenen Stellen ausführen will, muss man nicht immer wieder neu schreiben: Man definiert sie in einer Funktion und verwendet diese wieder. Man spart sich Schreibarbeit und muss weniger tippen.

* **Struktur**

Programm wird in kleinere Unterprogramme runtergebrochen. Somit kann man größere Programme bauen: Indem viele kleine Programme zusammenarbeiten.