g;rls' <digital> /camps/

empfohlen für Sekundarstufe I

Stefanie Umlauft

PROGRAMMIEREN IN OPEN PROCESSING

Grafische Animationen digital gestalten

INFORMATIK, BILDENDE KUNST

GEFÖRDERT DURCH
Baden-Württemberg
MINISTERIUM FÜR WIRTSCHAFT, ARBEIT UND WOHNUNGSBAU
VDI Württembergischer Ingenieurverein
VDI TecCity
Stadt Böblingen Raum für Taten und Talente
Klett MINT
FRAUEN IN MINT BERUFEN

Rund um das Projekt gibt es vielfältige Materialien, Sie finden diese hier

Vorwort

LIEBE LESERIN, LIEBER LESER,

Digitalisierung wird für die berufliche Zukunft eine tragende Rolle spielen. In diesem Bereich entstehen bereits heute neue, kreative und herausfordernde Berufsfelder und -bilder, die für alle junge Menschen von Relevanz sind und insbesondere für Mädchen und junge Frauen neue Chancen und berufliche Perspektiven eröffnen. Daher fördete das Ministerium für Wirtschaft, Arbeit und Wohnungsbau Baden-Württemberg das Modellprojekt „Girls' Digital Camps" von 2018 bis 2020 in fünf Wirtschaftsregionen in Baden-Württemberg. Die in der Modellphase entwickelten, erfolgreich evaluierten Konzepte sollen in einer Transferphase ab 2021 weiterentwickelt, verstetigt und über ganz Baden-Württemberg ausgerollt werden.

Wie können wir möglichst viele Mädchen für die Beschäftigung mit IT-Themen begeistern? Dieser Frage sind wir, Natalie Spahr und Angelika Baur, gemeinsam auf den Grund gegangen. Schnell waren Unterstützer*innen aus Unternehmen, Universitäten und Hochschulen gefunden, die die Vision, vielen Mädchen den Weg in attraktive Digitale Berufsfelder aufzuzeigen, teilten. Und so startete das erste Girls' Digital Camp im März 2019 an vier Schulen gleichzeitig als freiwillige AG am Nachmittag.

Insgesamt wurden über 250 Mädchen der Klassenstufen 6 bis 9 erreicht. Die Workshops boten Themenmodule mit einer Dauer von einer bis drei Doppelstunden an, um die Vielfalt der IT-Anwendungen aufzuzeigen. Bei der Entwicklung der Angebote für die Girls' Digital Camps wurde großer Wert darauf gelegt, die digitalen Themen mit der Lebens- und Erfahrungswelt der Mädchen zu verknüpfen, um so einen möglichst hohen Motivationsanreiz zu bieten.

Ein großer Dank gilt dem unermüdlichen Einsatz unserer AG-Leiter*innen, die ihre Konzepte auf Basis ihrer Erfahrungen im Vermitteln digitaler Kompetenzen mit viel Kreativität ausgearbeitet haben. Die Unterrichtseinheiten haben die Mädchen begeistert.

Viel Spaß bei der Umsetzung im Unterricht

wünschen Ihnen

Natalie Spahr
VDI-Württembergischer Ingenieurverein e. V.

Angelika Baur
Gleichstellungsbeauftragte der Stadt Böblingen

IMPRESSUM

1. Auflage 2021

Herausgeberinnen: Angelika Baur, Natalie Spahr

Autorin: Stefanie Umlauft

Herstellung: Klett MINT GmbH

Bildquellen: Girls' Digital Camps Region Stuttgart

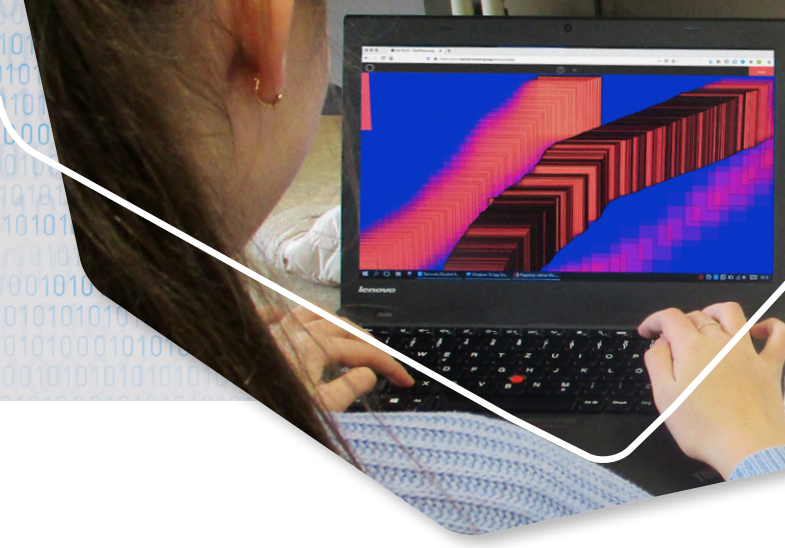
Layout und Satz: Tanja Bregulla, Aachen

Das Modellprojekt Girls' Digital Camps wird vom Ministerium für Wirtschaft, Arbeit und Wohnungsbau Baden-Württemberg gefördert.

Eine Zusammenarbeit des Vereins deutscher Ingenieure e. V.Stuttgart, der MINT Region Böblingen und der Klett MINT GmbH

Stefanie Umlauft

PROGRAMMIEREN IN OPEN PROCESSING



BEZUG ZUM BILDUNGSPLAN

Fach: Informatik, Bildende Kunst
Klassen: 7/8/9 Sek I
Gymnasium Baden-Württemberg

KOMPETENZBEREICH

In diesem Modul erlernen die Schüler*innen den Umgang mit der Programmiersprache P5js und deren Anwendung in der Programmierumgebung Open Processing. Sie können eigene visuelle Medienprodukte konzipieren, entwickeln und realisieren. Dabei werden Kenntnisse über das Einsatzgebiet des RGB-Farbraums, algorithmische Grundprinzipien sowie Funktionen des Programmierens wie z. B. „loop“ und „setup“ vermittelt.

ZIELE

Die Schüler*innen sollen in der Programmierumgebung Open Processing mehrere Formen in unterschiedlichen Farben erstellen können und dabei die algorithmischen Grundprinzipien des Programmierens anwenden.

LEITFRAGEN

- Was ist die Programmierumgebung Open Processing?
- Was kann ich in Open Processing programmieren?
- Wie funktioniert die Programmiersprache P5js?

AUSBLICK FÜR FOLGESTUNDEN

Die Schüler*innen erweitern ihr Wissen durch das Erlernen weiterer Programmcodes.

METHODIK

- Je nach Ausstattung an Computern ist eine Erarbeitung in Gruppen möglich, aber nicht empfehlenswert.
- Sozialform: Plenums- und Einzelarbeit im Wechsel

VORBEREITUNG

Geplant werden zwei Unterrichtsstunden (je 45 Minuten). Vorkenntnisse der Lehrkraft im Umgang mit Open Processing sind erforderlich, ebenso ein eigener Account der Lehrkraft, auf dem die Schüler*innen die Inhalte nachschauen können. Als Hilfe kann eine vorab erstellte Übersicht über die Funktionen von Open Processing bereit gestellt werden.

MATERIALLISTE

- Pro Schüler*in ein Desktop PC oder Laptop mit Tastatur, Maus/Touchpad und Internetzugang über einen modernen Browser
- Pro Schüler*in eine aktive E-Mail-Adresse

Lösungen der Arbeitsaufträge

Aufgabe 1: An der Position des Mauszeigers werden Kreise gezeichnet.

Aufgabe 2: In der Programmzeile *ellipse(mouseX, mouseY, 20, 20);* müssen die beiden Zahlen durch schülerindividuelle Werte ersetzt werden.

Aufgabe 3: Der Funktionsaufruf *ellipse();* wird durch *rect(mouseX, mouseY, schülerindividueller Wert, schülerindividueller Wert);* ersetzt.

Aufgabe 4: Schülerindividuelle Lösung.

Aufgabe 5: Grüntöne: *fill(0, random(0,255), 0);* Zufallsfarbe: *fill(random(0,255), random(0,255), random(0,255));*

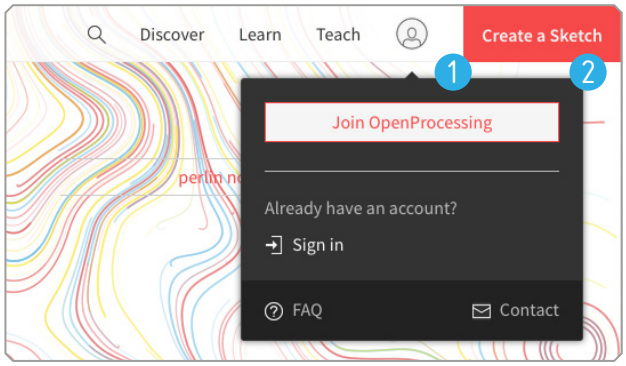
Aufgabe 6: Der Funktionsaufruf *rect()* lautet: *rect(random(0, windowWidth), random(0,windowHeight), schülerindividueller Wert, schülerindividueller Wert);*

ZUR SACHE

Open Processing ist eine Online-Programmierumgebung, mit welcher grafische Ausgaben erzeugt werden. Programmiert wird in der Sprache P5js, einer vereinfachten Version von JavaScript. Open Processing ist über www.openprocessing.org erreichbar, wo direkt im Browserfenster programmiert werden kann.

Open Processing Konto einrichten und starten

Damit die Schüler*innen mit Open Processing arbeiten können, müssen sich die Lernenden je ein eigenes Benutzerkonto anlegen. Auf der Startseite von Open Processing ist der Button **Join** 1 zu finden, ein Klick auf diesen öffnet das Anmeldeformular. Dort müssen ein Name (oder ein Pseudonym), eine E-Mail-Adresse und ein Passwort angegeben werden. Ein weiterer Klick auf **Join** bestätigt die Angaben und leitet die Benutzenden auf die eigene Profilseite weiter. Durch einen Klick auf den Button **Create a Sketch** 2 wird eine neue Zeichnung angelegt, welche bereits einigen Beispielcode enthält.



© Open Processing

Das Bildschirm-Koordinatensystem und die Einheit Pixel

In Open Processing werden grafische Ausgaben programmiert, weshalb es wichtig ist, dass die Lernenden das Koordinatensystem des Bildschirms kennen. Open Processing legt den Mittelpunkt des Koordinatensystems standardmäßig auf die linke obere Ecke des Bildschirms. Von dort aus wird die x- und y-Koordinate eines Punktes in Pixeln angegeben. Alle Größenangaben wie die Bildschirmgröße oder die Maße einer Form werden ebenfalls in Pixeln definiert.

Der RGB-Farbraum

In Open Processing wird standardmäßig mit dem RGB-Farbraum gearbeitet, welcher ein additiver Farbraum ist, der auf der Mischung der drei Grundfarben Rot, Grün und Blau basiert. Der Anteil der Grundfarben an der späteren Mischfarbe wird mit Werten zwischen 0 und 255 angegeben. Ein pures Rot entspricht daher dem Wert (255, 0, 0), Türkis (0, 255, 255) und ein mittleres Grau (100, 100, 100).

Systemvariablen

Systemvariablen sind Variablen, deren Wert automatisch festgelegt wird. Zum Beispiel enthält die Variable *windowWidth* die Breite des genutzten Bildschirms, *windowHeight* gibt hingegen seine Höhe an.

Funktionen

Jedes Programm in Open Processing besteht aus den Funktionen *function setup()* und *function loop()*. Die *setup()* Funktion wird einmal zum Start des Programms ausgeführt, anschließend wird die *loop()* Funktion bis zum Stopp des Programmes dauerhaft ausgeführt. Open Processing stellt eine Menge vorgefertigter Funktionen bereit, die in *setup()* und *loop()* aufgerufen werden können. Die Funktionsaufrufe bestehen immer aus dem Funktionsnamen und eventuell aus Parametern, die weitere Eigenschaften definieren. Ein Funktionsaufruf wird immer mit einem Semikolon beendet.
funktionsname(parameter1, parameter2, …);

Im Beispielcode von Open Processing werden in der *setup()*Funktion folgende Funktionen aufgerufen:

- *createCanvas(windowWidth, windowHeight);* legt die Größe der Zeichenfläche fest.
- *background(100);* definiert die Hintergrundfarbe (Werte 0–255, 0 entspricht schwarz, 255 weiß)

TIPP

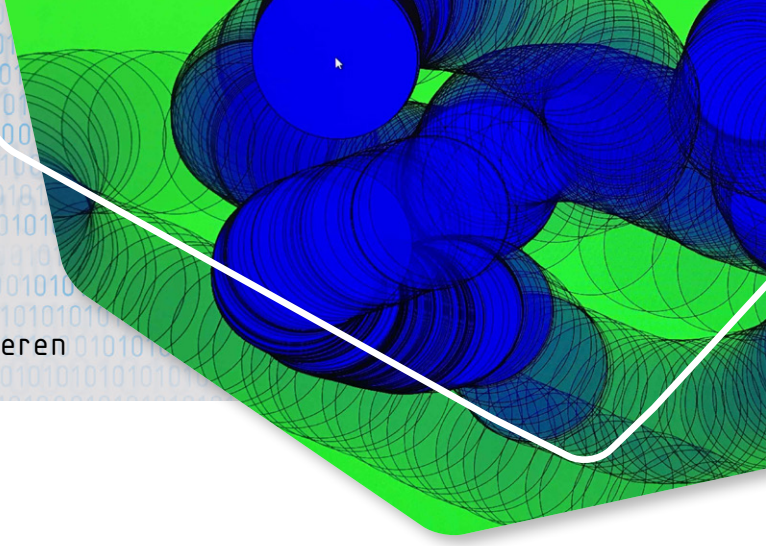
Link

Dokumentation mit allen Funktionen und Systemvariablen
www.p5js.org/reference

Arbeitsblatt

WIE FUNKTIONIERT OPEN PROCESSING?

Bunte Formen in Open Processing programmieren



Wie programmiere ich in Open Processing?

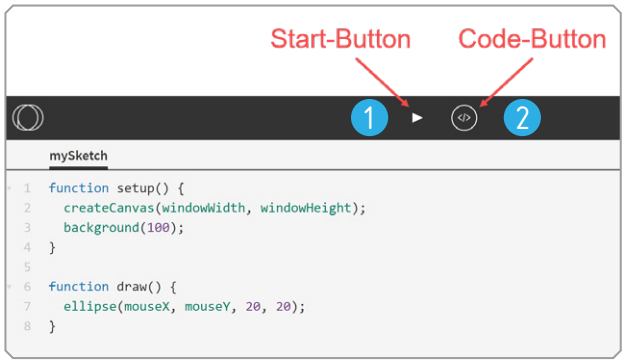
Open Processing ist eine Online-Programmierumgebung, in der du bewegte Bilder erstellen kannst. Lege zuerst ein Benutzerkonto auf der Seite www.openprocessing.org an. Gehe auf deine Profilseite und klicke dort auf den Button „Create a new Sketch“, um die Programmierumgebung zu öffnen.

1. Das Standardprogramm erkunden

Du siehst nun einige vorgefertigte Codezeilen. Klicke auf den **Start-Button** 1, um den Code auszuführen und so das Programm zu starten. Bewege anschließend den Mauszeiger über den Bildschirm und beobachte, was passiert.

Meine Beobachtung:

Wechsle danach durch einen Klick auf den **Code-Button** 2 wieder zu deinem Programmcode.



Die Programmierumgebung mit Beispielcode. Im oberen Bereich des Fensters befinden sich der Start- und der Code-Button

2. Größe der Ellipse ändern

Die Funktion *ellipse(mouseX, mouseY, 20, 20)* zeichnet eine Ellipse. Die Werte in den Klammern geben an, an welcher Position und mit welchem Durchmesser die Ellipse gezeichnet wird:
ellipse(x-Koordinate, y-Koordinate, Durchmesser x, Durchmesser y);

In den Systemvariablen *mouseX* und *mouseY* speichert der PC automatisch die aktuelle Position des Mauszeigers.

Ändere den Durchmesser der Ellipse auf verschiedene Werte deiner Wahl und schaue, was passiert, wenn du das Programm startest.

Meine Beipiele:

Schon gewusst?

Der Computer ignoriert Leerzeichen und Absätze beim Verarbeiten deines Codes. Er erkennt zusammengehörige Codeabschnitte an den Klammern { } und das Ende eines Funktionsaufrufs am Semikolon. Achte darauf, keine Klammern oder Semikolons zu vergessen.

3. Form ändern
Zeichne anstelle der Ellipse ein Rechteck. Die Funktion dafür lautet:
rect(x-Koordinate, y-Koordinate, Breite, Höhe);

Mein Code:

4. In Farbe zeichnen
Wenn du in Open Processing eine Farbe festlegst, werden alle darauffolgenden Objekte in dieser Farbe gezeichnet. Die Funktion, mit der du eine Farbe festlegen kannst, lautet:
fill(Rotwert, Grünwert, Blauwert);

mySketch

```
1  function setup() {
2    createCanvas(windowWidth, windowHeight);
3    background(100);
4  }
5
6  function draw() {
7    fill(30, 125, 240);
8    rect(mouseX, mouseY, 30, 30);
9  }
```

Der Funktionsaufruf von fill() wird immer vor dem Funktionsaufruf zum Zeichnen eines Objekts in den Code eingefügt.

Füge die Funktion *fill()* in deinen Programmcode ein (das Bild zeigt dir die richtige Stelle im Code). Probiere dann verschiede Farbwerte aus und beobachte, welche Farbe das Rechteck annimmt.

Meine Farbbeispiele:

5. Zufallsfarben
Anstelle von festgelegten Farbwerten, kannst du die Ellipse oder das Rechteck auch in einer zufälligen Farbe einfärben. Die folgende Funktion erzeugt einen zufälligen Wert innerhalb eines Wertebereichs:
random(Minimalwert, Maximalwert);

Um eine Zufallsfarbe zu erhalten, kombiniere die Funktionen *fill()* und *random()*. Ersetze den ersten Wert der Funktion *fill()* durch die Funktion *random (0,255)*. Deine *fill()*-Funktion sollte dann so aussehen:
fill(random(0,255), 0, 0);

Starte dein Programm und beobachte, was passiert:

Versuche anschließend, zuerst nur zufällige Grüntöne zu erhalten und dann, in einer komplett zufälligen Farbe zu zeichnen.

Mein Code:

6. Zufällige Position
Bisher wurden die Ellipse und das Rechteck immer an die Position deines Mauszeigers gezeichnet. Versuche, Rechtecke an einer zufälligen Position auf deinem Bildschirm zu zeichnen.
TIPP In der Systemvariablen *windowWidth* speichert der PC die Bildschirmbreite, *windowHeight* enthält die Bildschirmhöhe.

Mein Code: