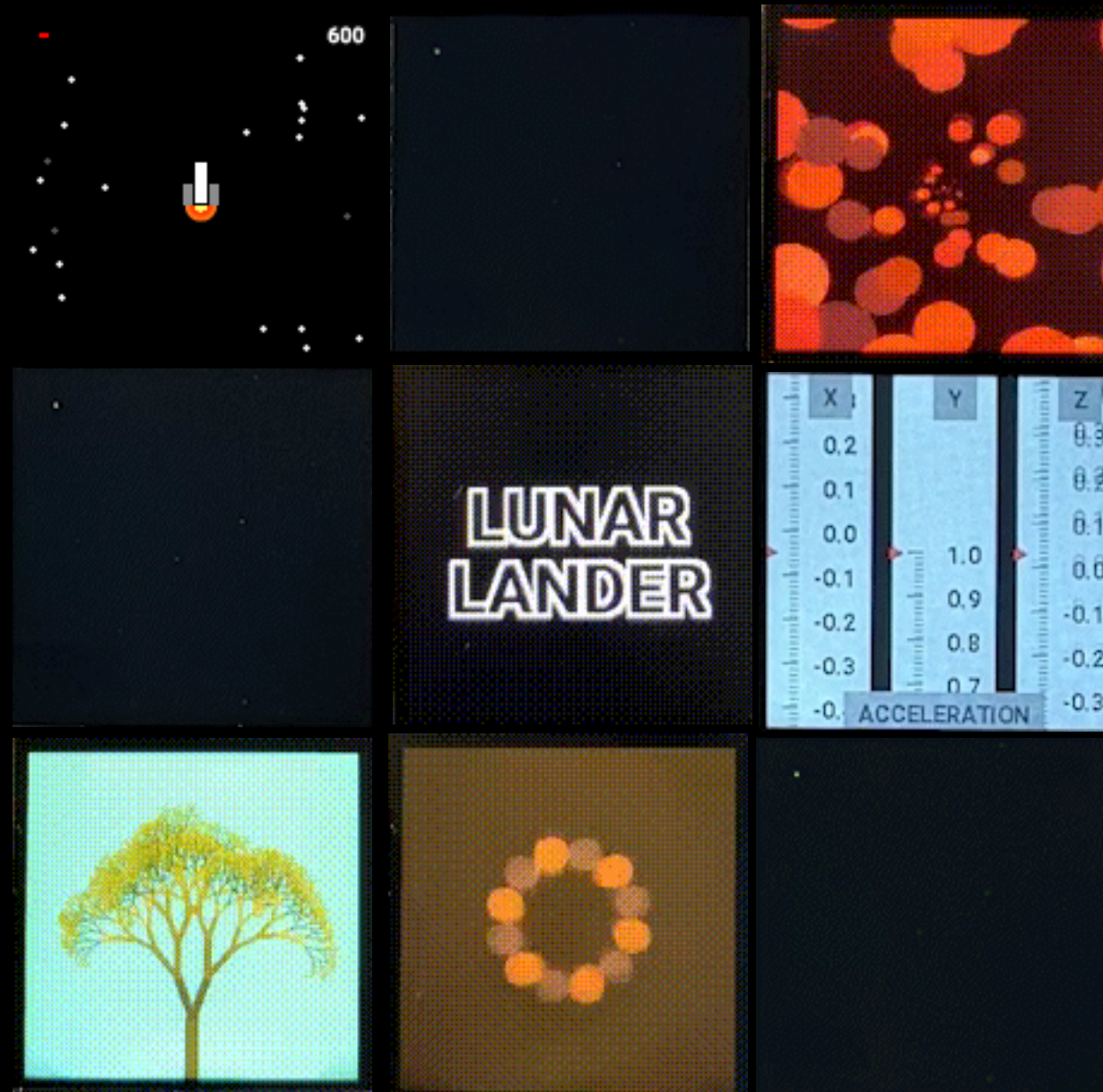


INFORMATIKMAGIE

KREATIVER EINSTIEG IN DIE FASZINIERENDE
WELT DES PROGRAMMIERENS



*«Ich habe in kurzer Zeit schon
grosse Fortschritte gemacht.»*

Lena, 1. Oberstufe

Die Oxocard Galaxy hat nicht nur die Schülerinnen und Schüler, sondern auch die **Lehrpersonen begeistert**. Die einfache Handhabung und die umfangreichen Möglichkeiten zur Programmierung haben es uns ermöglicht, den **Unterricht interaktiver und praxisnaher** zu gestalten. Die Lernenden waren **motiviert**, ihre eigenen Projekte zu realisieren, und wir konnten ihre Fortschritte verfolgen und unterstützen.

Lehrperson, Sek I

«Mir gefällt, dass ich sehr schnell etwas Cooles programmieren kann. Weil die Programmierungen bereits „fertig“ sind, funktioniert es auch. So kann ich es abändern und dabei viel über das Programmieren lernen.»

Nico, 2. Oberstufe

Zusammenfassend kann man festhalten, dass die neue Oxocard Galaxy bei den Schülerinnen und Schülern gut ankam. Es wurde von vielen zurückgemeldet, dass sie es spannend fanden zu sehen, wie ein Programm-Code aussieht, und dass sie selber «coden» konnten. Bei den ersten Tests (vor den Sommerferien) gab es noch Verbindungsprobleme, da die Karte über WLAN mit dem Editor verbunden werden musste. Beim zweiten Durchgang (vor den Herbstferien) gab es diese Probleme nicht mehr. Diejenigen, die die klassische Oxocard schon kannten, fanden diese neue Oxocard natürlich «cooler» und aufregender, jedoch auch komplizierter. (Das hat aber natürlich auch damit zu tun, dass die klassische Oxocard für ein jüngeres Zielpublikum konzipiert ist).

Im Anschluss an den Test wurde ein kurzes Feedback eingeholt, in dem die Schülerinnen und Schüler auch eine Bewertung abgeben konnten. Vor den Sommerferien lag die Bewertung bei ca. 3.8 Sternen (von maximal 5 Sternen). Im zweiten Test (als die Verbindung zum Editor nicht mehr notwendigerweise über WLAN geschehen musste) stieg sie die Bewertung auf 4.1 Sterne.

Die neue Oxocard Galaxy (und sicherlich auch die anderen Oxocards: Science und Artwork) sind meiner Meinung nach für die Jugendlichen sehr attraktiv und gut durchdacht. Sie eignen sich sehr gut, um ihnen das Programmieren (und die Informatik im Allgemeinen) näherzubringen. Auch das Tutorial ist gut erstellt und

Lehrperson, Sek II

*«Es ist cool, hinter die
Programmcodes zu sehen. Mit
der Oxocard Galaxy kann ich
das textbasierte
Programmieren sehr einfach
lernen. Es hat für alle Levels
etwas dabei – als Anfänger oder
auch als Crack.»*

Tim, 2. Oberstufe

Oxocard mini

Lernplattform mit Display und Sensoren

Die Oxocard kommt auf einer kreditkarten-großen Platine daher und richtet sich an Programmierneinsteiger ab 14 Jahren. Sie basiert auf einem ESP32 und ist ausgestattet mit einem Display, einem Lautsprecher und einem kreuzförmig angeordneten Eingabe-Pad. Insgesamt gibt es drei Versionen, die sich leicht unterscheiden: Art-work, Galaxy und Science. Während Artwork und Galaxy nur unterschiedlich dekoriert, aber sonst baugleich sind und einen Beschleunigungssensor enthalten, ist Science zusätzlich mit diversen Umweltsensoren ausgestattet.

Obwohl die Oxocard über kein Gehäuse verfügt, sieht sie ansprechend aus, da die Leiterplatte mit künstlerischen Bildern bedruckt ist. Aufgrund einer fehlenden Batterie muss sie

jedoch immer über USB mit einer Stromquelle verbunden sein.

Außer zwei vorinstallierten Spielen und einigen Demo-Skripten, die die technischen Möglichkeiten des Gerätes vorführen, hat die Oxocard anfangs noch nicht viel zu bieten. Das ändert sich allerdings, wenn man das kleine Gerät mit dem Internet verbunden hat und mit dem Online-Code-Editor koppelt (Link siehe Kurz-URL, alternativ kann auch eine Verbindung per USB-Kabel hergestellt werden). Mittels einer Python ähnlichen Programmiersprache lassen sich hier selber Programme entwickeln. Es handelt sich um eine vereinfachte Version von Python.

In der Online-IDE hat man Zugriff auf den Code von allen Spielen und Demo-Skripten. So kann man sich

durch die Beispiele arbeiten und mit Änderungen spielerisch herausfinden, was welcher Befehl macht und wie man bestehenden Code anpassen kann. Man arbeitet hier in einer sehr „realistischen“ Entwicklungsumgebung, von der der Sprung in eine „richtige“ IDE nicht schwerfallen sollte.

Ich konnte bislang kein Python. Trotzdem konnte ich innerhalb eines Tages eine einfache Version von Snake auf einer Oxocard programmieren. Dank der Dokumentation fühlte ich mich immer angenehm an die Hand genommen. Wäre ich heute 14 und hätte auf diese Weise die Möglichkeit, direkt in eine richtige Programmiersprache einzusteigen, wäre ich Feuer und Flamme für die Oxocards. —das

► make-magazin.de/xvjh

| | |
|------------|---|
| Hersteller | Oxon |
| URL | https://www.oxocard.ch |
| Preis | 59 CHF (Artwork und Galaxy), 89 CHF (Science) |



Informatikmagie

Kreativer Einstieg in die faszinierende Welt des Programmierens

Das Buch dient als Lernbegleitung für die Oxo-cards, die wir hier auf der gleichen Seite vorstellen. Es geht einen anderen Weg als viele herkömmlichen Bücher rund ums Programmieren für Einsteiger. Der Autor erklärt anhand von gut gewählten Praxisbeispielen, wie die Python-ähnlichen Skripte für die Oxocards funktionieren und was passiert, wenn man bestimmte Parameter respektive Konstanten ändert.

Das exerziert er schön und leicht verständlich an 13 gut gewählten Beispielen durch, die sich alle rund ums Display und teilweise auch um die verbauten Sensoren drehen. So werden drei Uhren durchprobiert, Bälle, Würmer und Feuer animiert, Schneeflocken und Bäume gezeichnet, Neigungen gemessen und Spannungen angezeigt. Erst nach diesen Kapiteln erläutert Thomas Garaio die Grundelemente der Sprache, etwa welche Arten von Variablen es gibt, wie sie im Speicher abgelegt sind, wie Schleifen und Bedingungen funktionieren und was Klassen und Listen sind.

Der Vorteil dieses Ansatzes: Erst wird man begeistert und inspiriert und dann folgt die trockene Theorie. Man wünscht sich, mehr Autoren würden diesen Weg wählen.

Ein wenig Kritik gibt es dennoch: Der browserbasierte Editor erscheint in einem dunklen Schema, was dazu führt, dass manche Abbildungen im Buch nur schwer zu erkennen sind. Bei der Erklärung von Fließkommazahlen und der Exponentialschreibweise sind im Druck leider die Exponenten verrutscht: aus 2^1 wurde 21 und



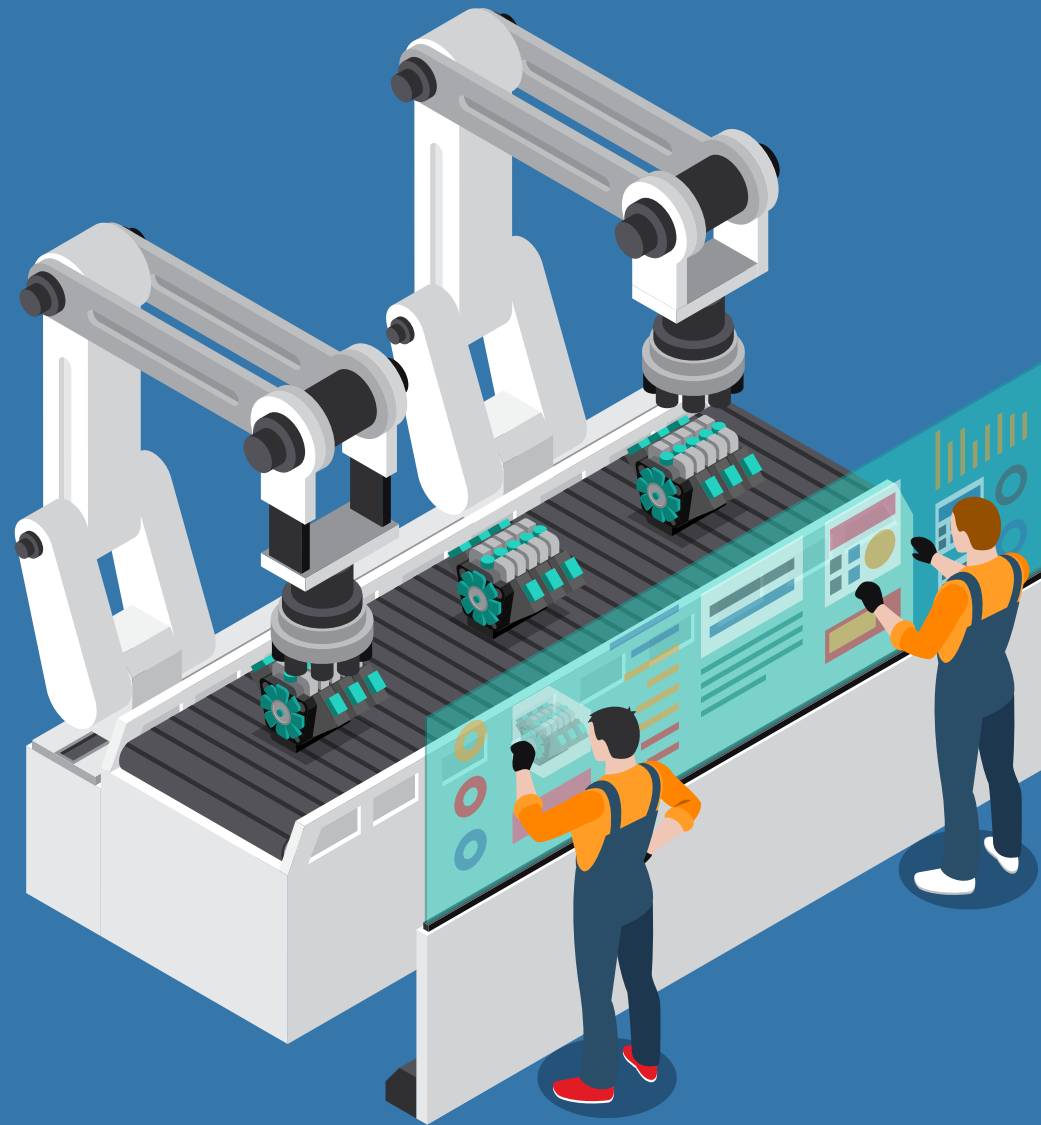
aus 10^1 wurde 101. Da steht man als Neuling auf dem Schlauch. Dennoch ist **das Buch eine Empfehlung.** —dab

| | |
|--------|----------------------|
| Autor | Thomas Garaio |
| Verlag | hep Verlag (Schweiz) |
| Umfang | 176 Seiten |
| ISBN | 978-3-0355-2230-3 |
| Preis | 40 € |



**Es gibt drei Gründe, warum Programmieren
so wichtig ist**

Erstens: Computer sind das wichtigste Werkzeug der modernen Gesellschaft



Programmieren eröffnet dir eine neue Dimension, den Computer an deine Bedürfnisse anzupassen

Zweitens: Computer sind heute allgegenwärtig



Zu verstehen, wie all diese Geräte prinzipiell funktionieren, macht dich mündig und kompetent

Drittens:

Programmierer:innen entwickeln aussergewöhnliche Denkmuster

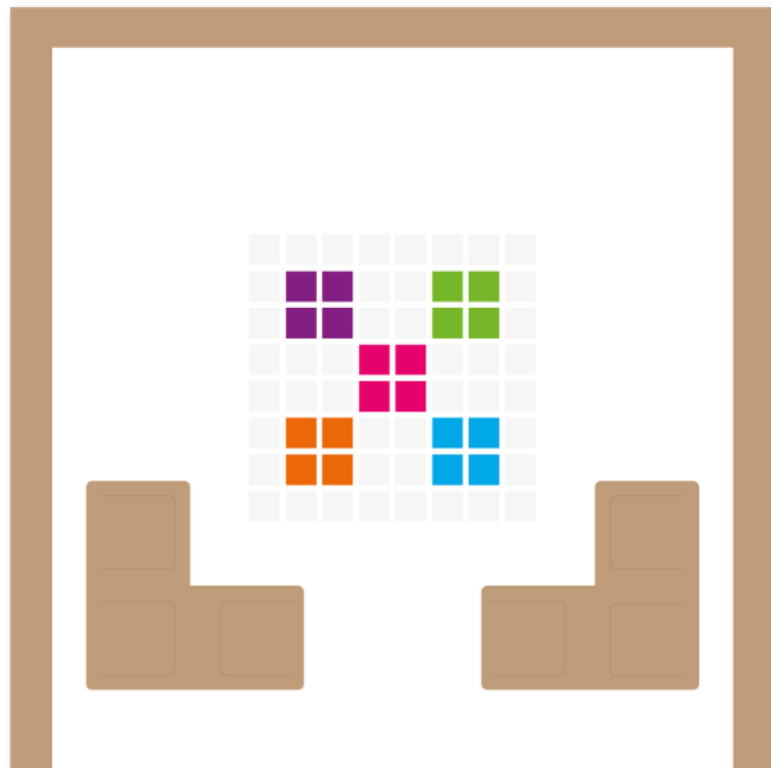


Es fördert nachweislich deine analytischen Fähigkeiten, komplexe Probleme durch Unterteilung in kleinere zu reduzieren und Muster zu erkennen.

Dies ist bekannt unter dem Begriff Computational Thinking [CT]

Oxocard Lernsysteme

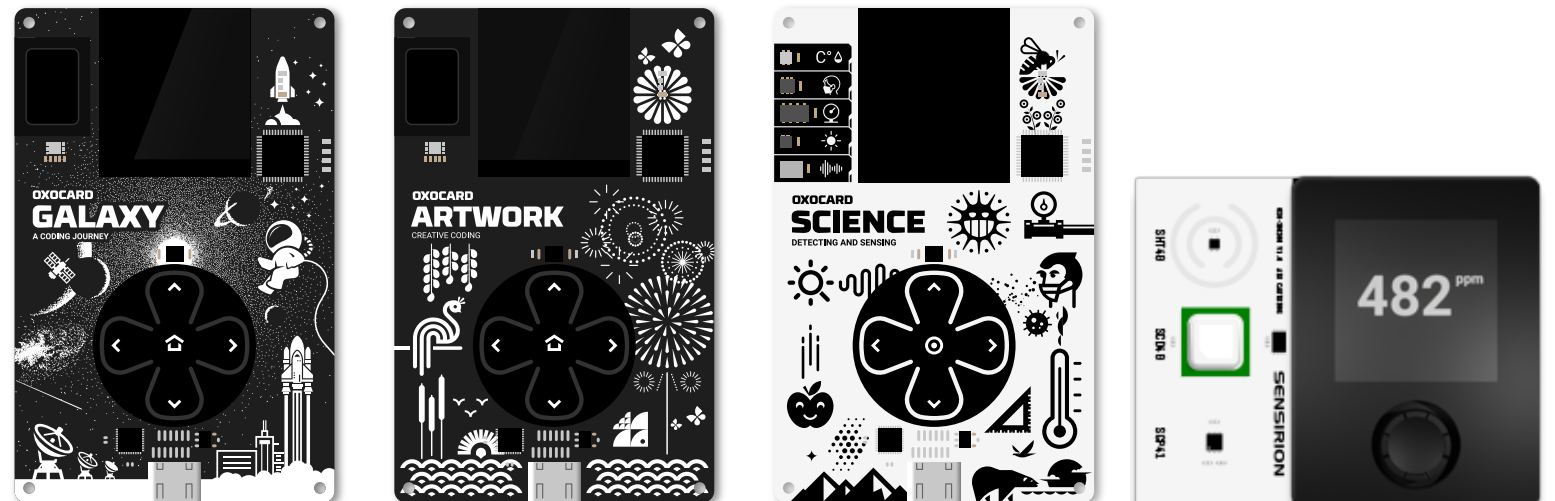
Oxocard Blockly



Niederschwelliger Einstieg mit
blockbasierter Programmierung

Ab 10 Jahren

Oxocard Mini Serie

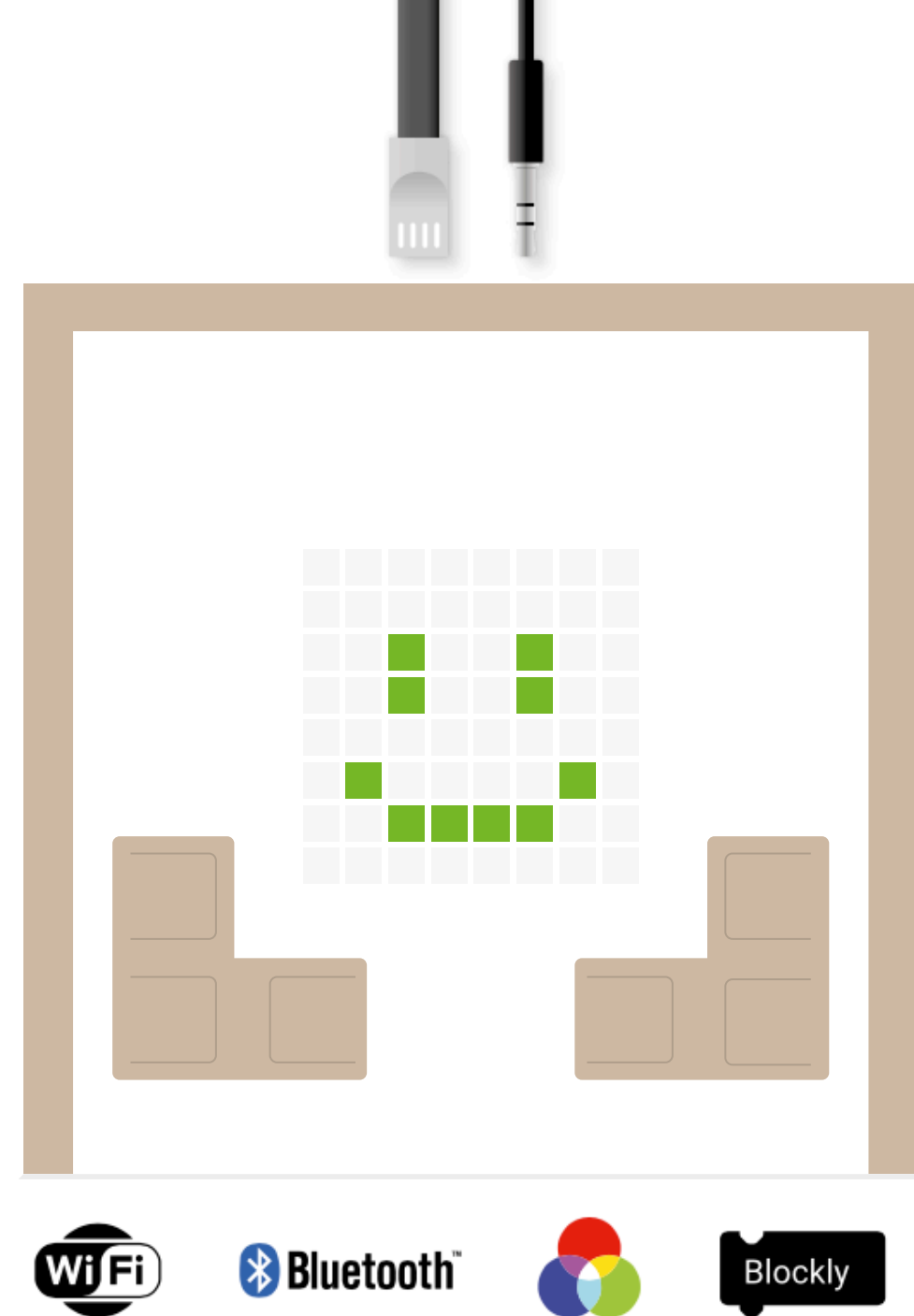


Textbasiertes Programmieren auf Basis
vorhandener Beispiele

Ab 14 Jahren

OXOCARD BLOCKLY

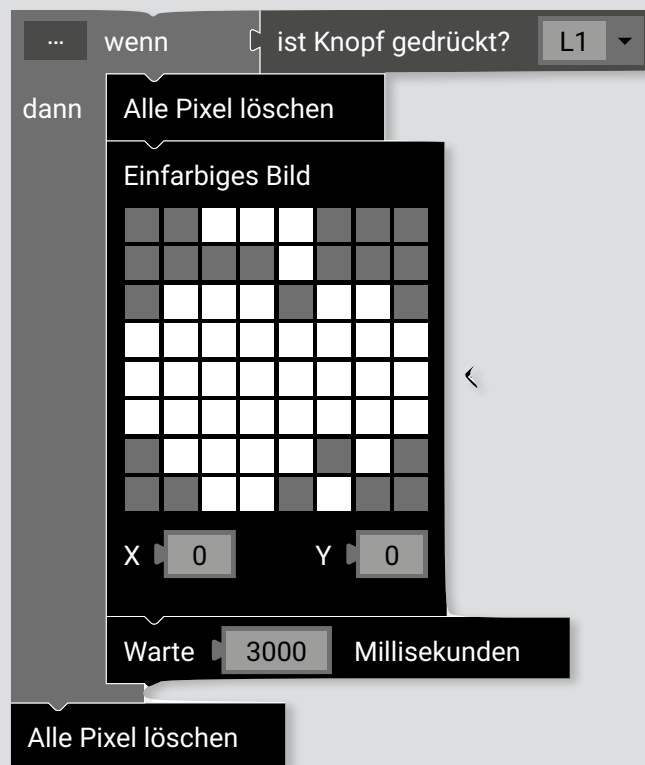
Spielend programmieren lernen
mit der mehrfach ausgezeichneten
Oxocard und der visuellen
Programmiersprache Blockly.



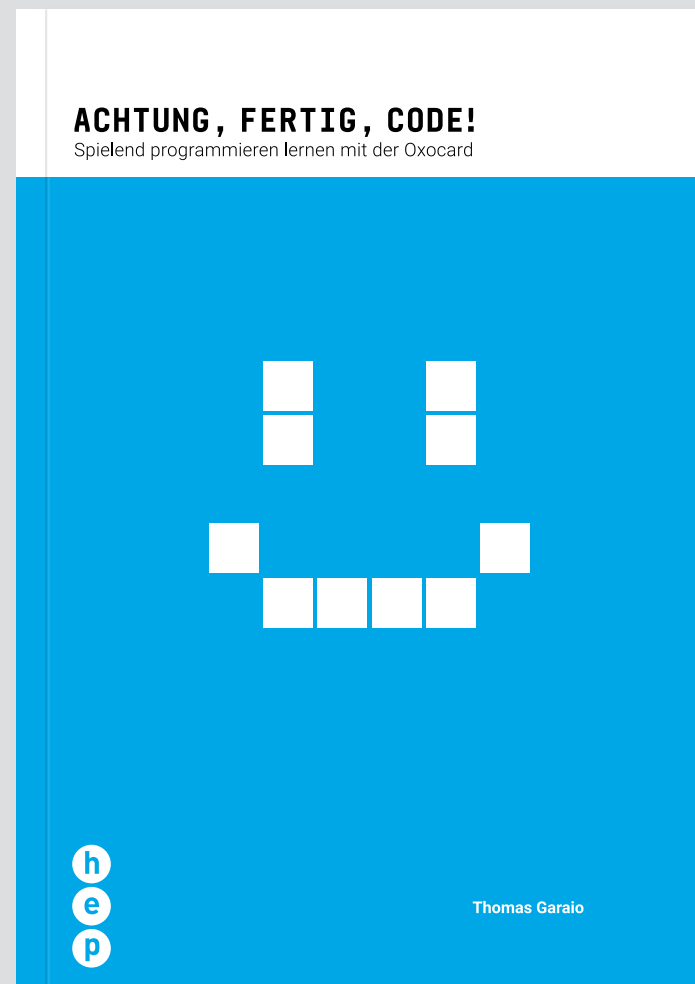
worlddidac
A W A R D 2 0 1 8

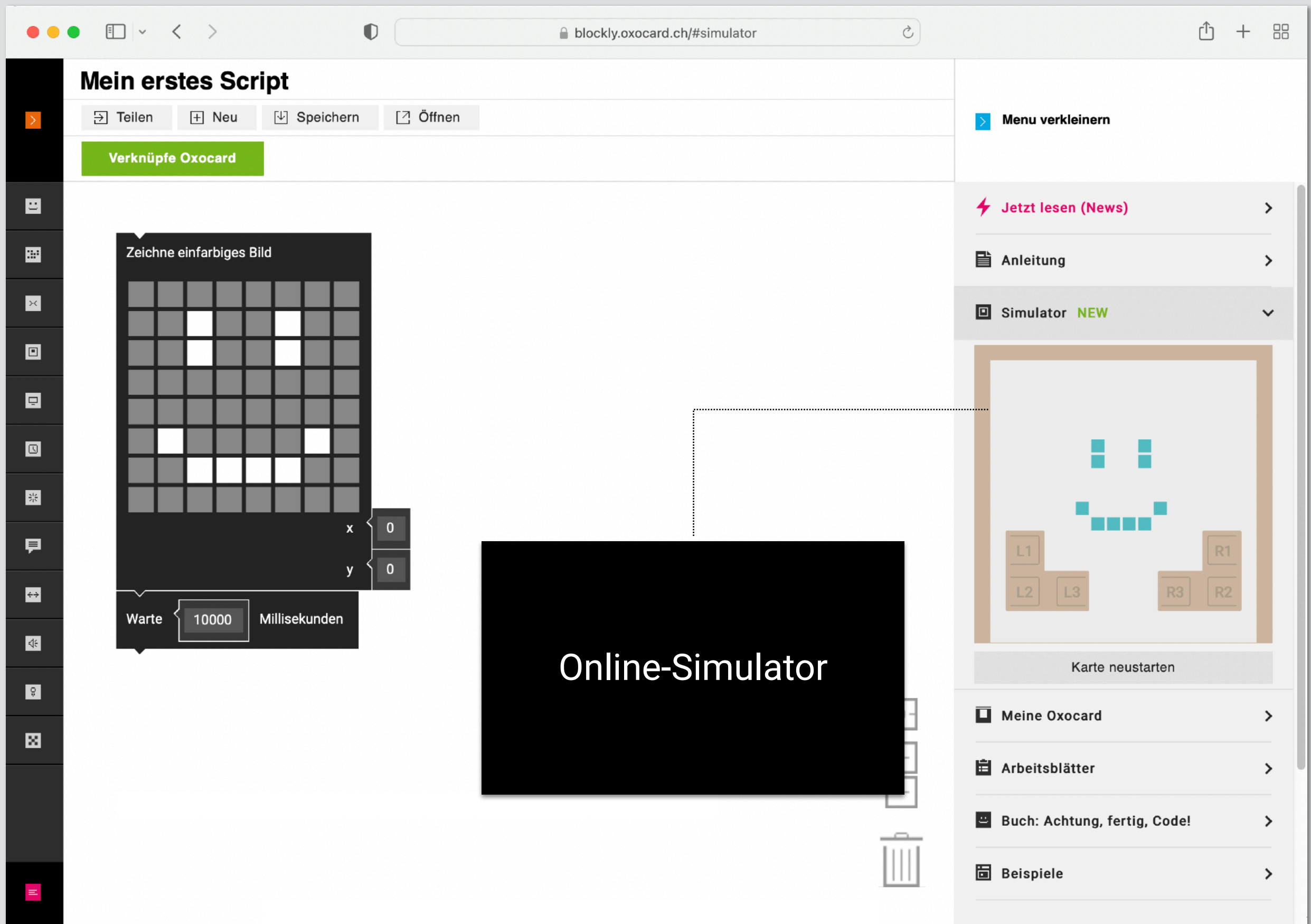
BELMA
Best European Learning Materials Awards

Blockbasierte Programmierung



Schweizer Lehrmittel





Oxocard Mini Serie

Die Oxocard Mini Karten sind programmierbare, kreditkartengrosse Computer mit einer umfangreichen Softwarebibliothek.

GALAXY



ARTWORK



SCIENCE



CONNECT



Spielfertige Games und ausgewählte Grafik-Simulationen laden zum Experimentieren ein

Algorithmische Kunst für Einsteiger - lerne in kurzer Zeit, wie visuelle

Entdecke und erforsche deine Umwelt mit dreizehn leistungsfähigen Sensorwerten

Plug and Play Elektronik: Cartridge einstecken und sofort ausprobieren

Audio
Lautsprecher mit Verstärker für
sphärischen Klang, der
dich von den Socken haut

Bombastischer Screen

240 × 240 Pixel = 57'600 RGB Pixel!

3D Beschleunigungssensor

Buttons
Fünf taktile Buttons

Leistung für coole Apps
8 MB Flash, 2MB RAM und
Dual-Core Chip

 **NANOPY - OS**



OXOCARD
GALAXY & ARTWORK



13 SENSOR-WERTE

Temperature
Humidity
Pressure
VoC
eCO₂
Ethanol
IAQ
Brightness
RGB-Light
Infrared
Audio frequencies
Decibel
3D-Acceleration

Buttons

Fünf taktile Buttons

Bombastischer Screen

240 × 240 Pixel = 57'600 RGB Pixel!

Leistung für coole Apps

8 MB Flash, 2MB RAM und
Dual-Core Chip



OXOCARD
SCIENCE



OXOCARD CONNECT

USB-C

Bombastischer Screen

240 × 240 Pixel = 57'600 RGB Pixel!

Leistung für coole Apps

8 MB Flash, 2MB RAM und
Dual-Core Chip

 **NANOPY - OS**

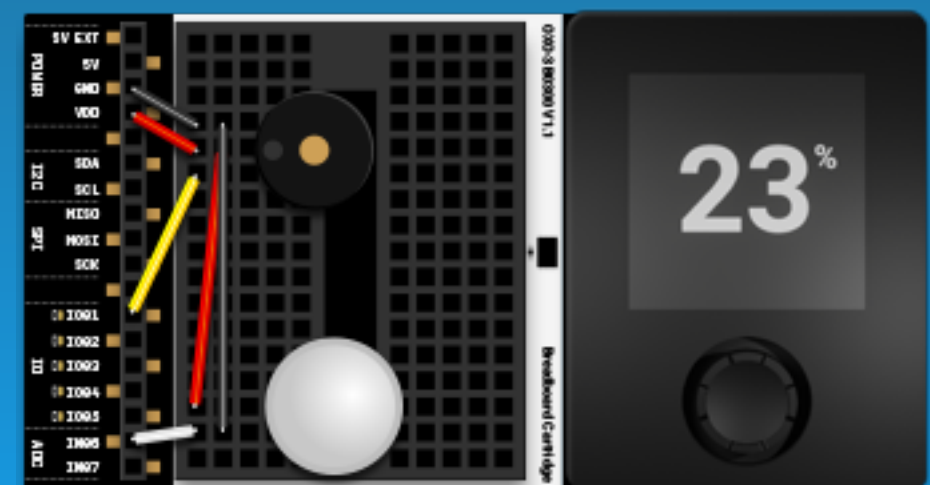
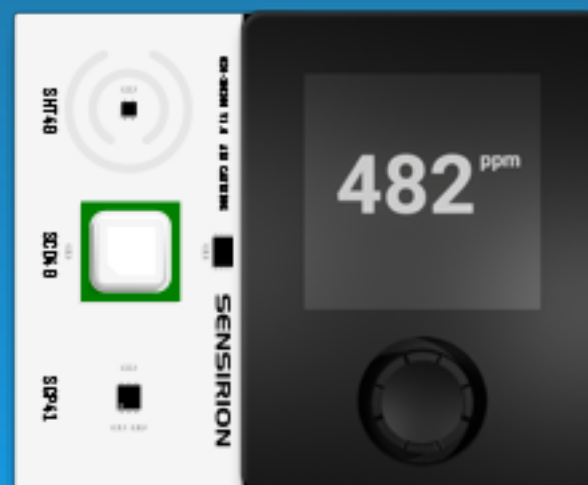
4+1 Daumen-Joystick

16 Pin Cartridge Connector

Plug&Play Elektronik-
Adapter

Erweiterungen

Elektronik, Sensoren,
Aktoren und mehr



NanoPy OS - Das "iPod"-UI für IoT-Geräte

«START»

Diese Funktion startet das zuvor installierte Programm.

«Games»

In dieser Liste finden sich die vorinstallierten Spiele, für die im Editor auch der Source-Code zur Verfügung steht.

«Demos»

Alle Karten verfügen über eine Liste kleiner Computeranimationen, die man anklicken und ausführen kann. Sobald man die Karte mit dem Editor verbunden hat, lassen sich diese Programme auch anpassen.

«WiFi»

Mit dieser Funktion wird die Karte in einen Hotspot-Modus versetzt. Sie dient dazu, die WiFi-Einstellungen des lokalen Netzwerks zu hinterlegen. Ohne diese kann die Karte keine Verbindung mit dem Oxocard-Server vornehmen und sie lässt sich nicht programmieren.



Nutze Schieberegler um Code zu ändern

INNOVATION

The screenshot displays the Nanopy web editor interface. The top navigation bar includes the Nanopy logo and a menu icon. The main area is divided into two panels. The left panel contains a code editor with a Python script for an alien character. The right panel, titled 'Konstanten:', features two sliders: 'ALIEN_COLOR' (a color picker) and 'COUNT_EDGES' (a range slider). A red dashed box highlights the 'ALIEN_COLOR' constant in the code and its corresponding slider in the panel.

```
1 const ALIEN_COLOR = 56 # HUE
2 const COUNT_EDGES = 10 # 5 .. 50
3
4 class Alien:
5     pos:vector
6     size:int
7     radius:int
8     faceColor:color
9     neckColor:color
10    offsets:int[COUNT_EDGES]
11
12    def init(x,y,s):
13        pos.x = x
14        pos.y = y
15        size = s
16        radius = size/4
17        faceColor.hsv(ALIEN_COLOR,255,255)
18        neckColor.hsv(ALIEN_COLOR,255,150)
19        for i in sizeof(offsets):
20            offsets[i] = radius/2-random(0,radius)
21
22
23    def drawEye1():
24        push()
25        noStroke()
26        translate(pos.x,pos.y-size/8)
27        fill(255,255,255)
28        drawCircle(0,0,size/10)
29        fill(0,0,0)
30        drawCircle(0,0,size/16)
31        pop()
32
33    def drawEye2():
34        push()
```

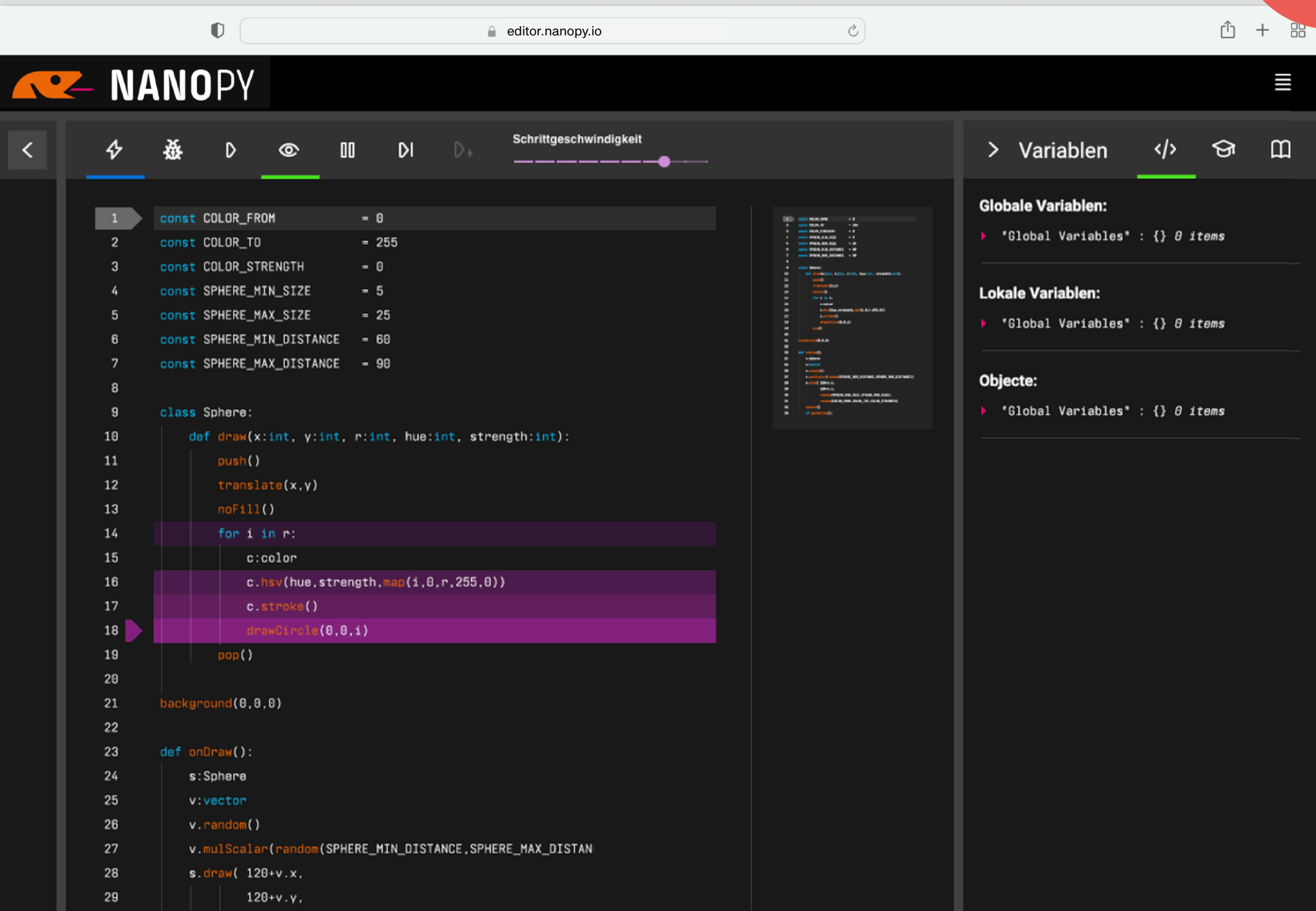
Konstanten:

ALIEN_COLOR

COUNT_EDGES

Beobachte das Programm während es läuft

INNOVATION



The screenshot displays the Nanopy web editor interface. The top navigation bar includes the Nanopy logo and a hamburger menu. The main editor area is divided into three sections: a code editor on the left, a console/output window in the middle, and a variables/objects panel on the right.

Code Editor: The code is written in Python and defines a `Sphere` class with a `draw` method. The `draw` method uses `push()`, `translate(x,y)`, `noFill()`, `for i in r:` loop, `c:color`, `c.hsv(hue, strength, map(i, 0, r, 255, 0))`, `c.stroke()`, `drawCircle(0, 0, i)`, and `pop()`. The `onDraw` method creates a `Sphere` object `s` and calls `s.draw(120+v.x, 120+v.y, ...)`.

Console/Output Window: This window shows the execution of the code, including the creation of the `Sphere` object and the execution of the `draw` method.

Variables/Objects Panel: This panel displays the state of the program during execution. It shows global variables and local variables, all of which are currently empty ({}).

Debugging Tools: The interface includes a toolbar with icons for running, stepping through code, and other debugging functions. A progress bar labeled "Schrittgeschwindigkeit" (Step Speed) is also visible.

Python, C/C++ und NanoPy im Vergleich

Python:

```
for i in range(0,10):  
    print("Hello World!")
```

C/C++:

```
#include <iostream>  
  
int main() {  
    for(int i;i<10;i++) {  
        std::cout << "Hello World!";  
    }  
    return 0;  
}
```

NanoPy:

```
for i in 10  
    print "Hello World!"
```

NanoPy vereint die Einfachheit von Python mit der Leistungsfähigkeit maschinennaher Sprachen.

- ✓ Python-Syntax
- ✓ Statisch typisiert
- ✓ Statisches
Speichermanagement
- ✓ Integrierter Debugger
- ✓ Byte-Code-Interpreter mit HAL
für verschiedene Embedded
Controller

Ereignisgesteuerte Programmierung

```
def onTimer():  
    print("timer")  
  
def onClick():  
    b = getButtons()  
    if b.down:  
        setInterval(100)  
        print("start interval")  
    if b.up:  
        stopInterval()  
        print("stop interval")
```

```
on=false  
  
def onDraw():  
    clear()  
    if on:  
        drawCircle(120,120,80)  
    update()  
  
def onClick():  
    on = not on  
    delay(500)
```

Einfaches Programmiermodell bei dem Ereignisprozeduren durch das Betriebssystem automatisch aufgerufen werden.

- ✓ Keine Hardwarenahe Programmierung notwendig
- ✓ Einfaches API
- ✓ Kompakter, robuster Code

Umfangreicher Funktionsumfang

Computer graphics

2D: pixels, lines, circles, ellipses, rectangles, polygons; RGB and HSV colour space; translation, rotation, scaling, clipping

Texts: drawText, textWidth, textLeading, textFont

3D: cube, sphere, cuboids, cylinder, prism, any 3D shapes (vertices/triangles); 3D projection with a camera and a light source; rotation/translation of the world or the camera, scaling, RGB and HSV colour space, camera

Mathematics

sin/cos/tan/asin/acos/atan

sqrt, exp, log, avg, min, max, ceil, floor, lerp, round

vector (2d): normalize, add, sub, addScalar, subScalar, mulScalar, fromAngle, random, magnitude, distance, dot, limit

vector3D: normalize, add, sub, addScalar, subScalar, mulScalar, fromAngle, random, magnitude, distance, dot, cross

random, randomSeed

noise, noise2D, noise3D (Perlin)

Sensors

getButtons
getAcceleration
Internet date and time
Milliseconds

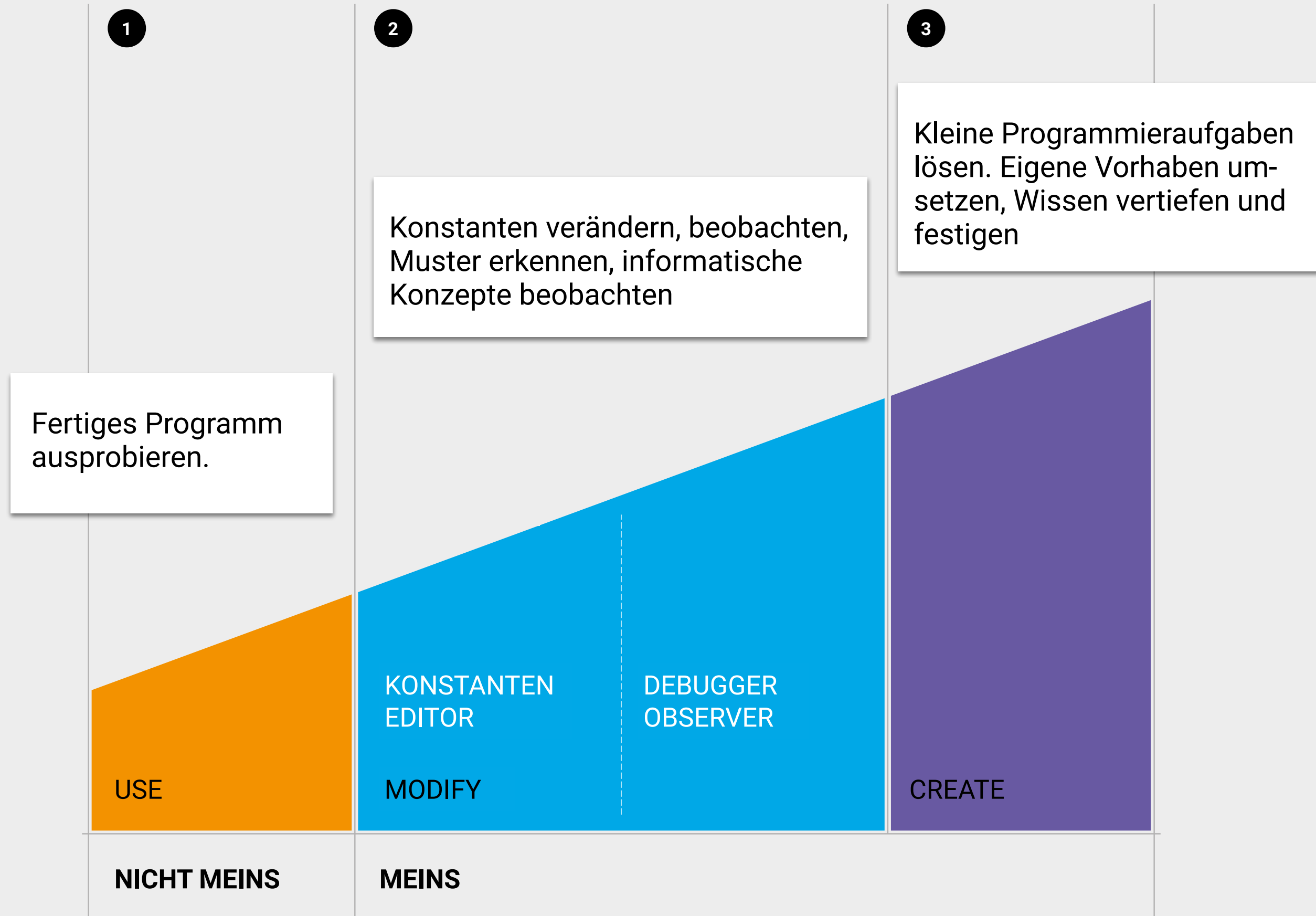
Science only:

getAcceleration
getTemperature, getHumidity, getPressure, getAmbientRGB, getAmbientIR, getAmbientLUX, getCO2, getIAQ, getTVOC, getETOH, getMicrophoneAmplitude, getMicrophoneDecibel, getMicrophoneFrequency

System

File IO (open, close, read/write byte/int/long/float, rename, delete, file size)
GPIO (init, write, read GPIO/ADC)
turnOff, sleep, setAutostart

LERNEN AM BEISPIEL



5.11 Natur: Meer

Das Meer hat – zumindest an einem schönen Tag – einen beruhigenden Effekt auf viele Menschen. Rhythmisch schlagen die Wellen ans Ufer. Auch dieser Wellengang lässt sich natürlich simulieren.

Wenn wir einen Stein in ein Becken werfen, schlägt dieser Wellen ins Wasser. Dabei sind diese zunächst noch hoch, schwächen sich aber ab, je weiter weg sie sich vom Aufschlagspunkt bewegen. Wenn wir zwei Steine reinwerfen, überlagern sich die beiden Wellen. Das nennt man Interferenz. Wenn zwei Wellen interferieren, können sich komplexe Muster bilden. An manchen Stellen verstärken sich die Wellen, an anderen Stellen gleichen sie sich aus. Vielleicht hast du das auch schon auf der Wasseroberfläche beobachtet.

In unserem Beispiel betrachten wir die Wellen, wie sie sich von der Seite präsentieren.



WAVE_HEIGHT
Höhe der Wellen

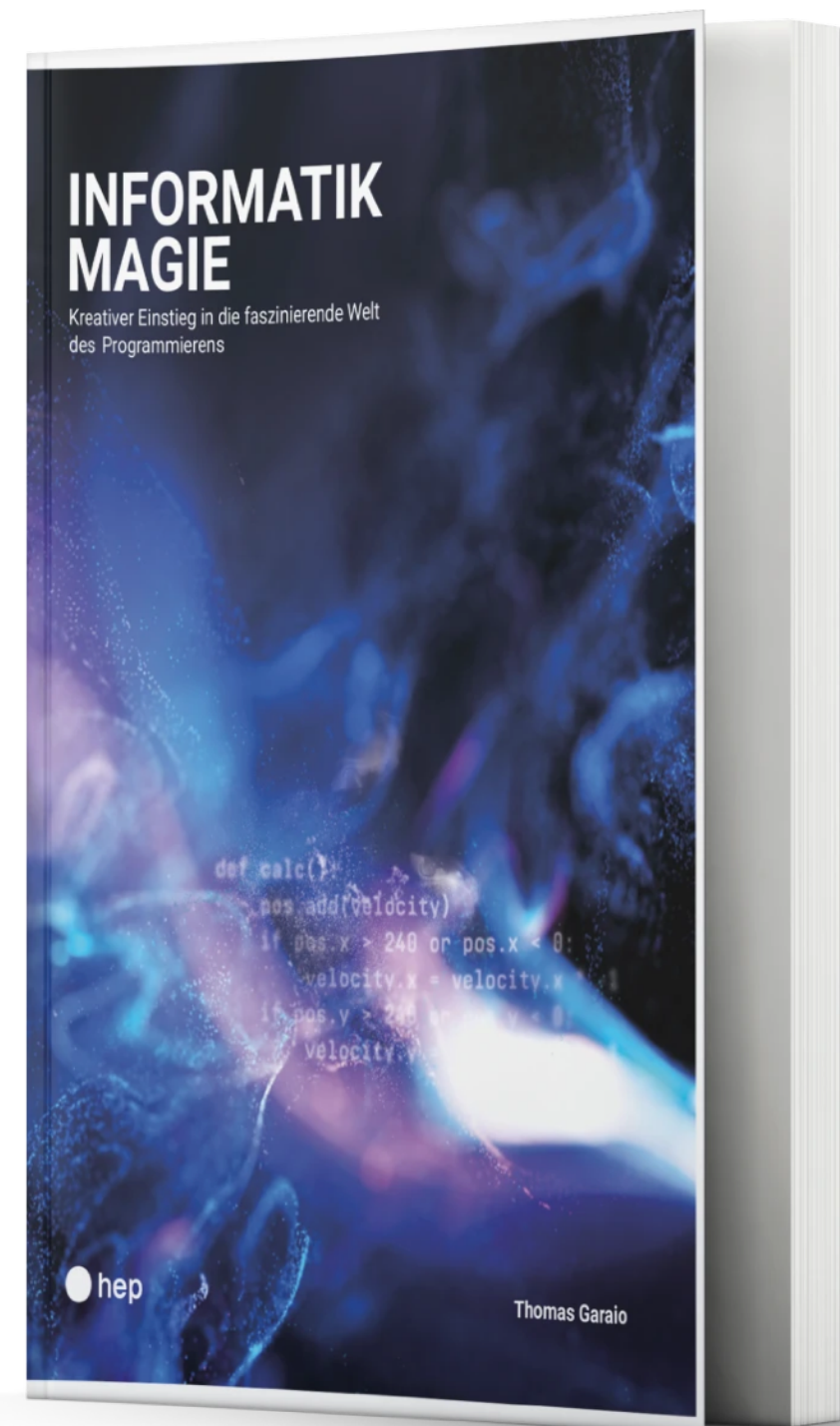
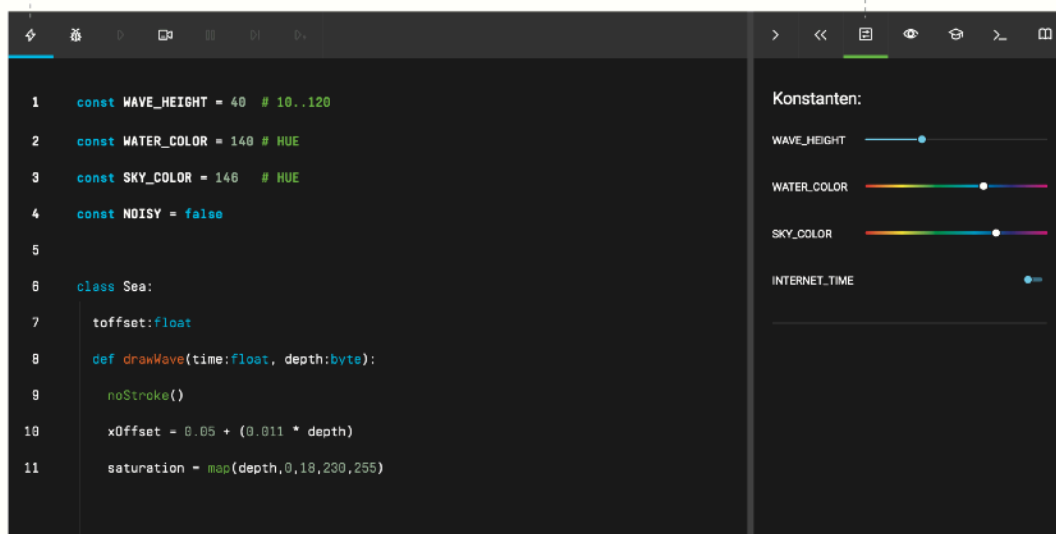
WATER_COLOR
Farbe des Wassers

SKY_COLOR
Farbe des Himmels im Hintergrund

NOISY
Bestimmt, ob das Meer unruhig ist.

1. Starte das Programm

2. Experimentiere mit den Konstanten



Aufbau des Buchs

| | |
|---------------------------|---|
| Einführung | Was sind die Oxocards, wie sind sie aufgebaut und wie funktionieren sie? |
| Einrichten | Wie bringt man die Karte in das W-Lan, wie verknüpft man sie mit dem Browser? |
| Erster Start | Wir zeigen, wie der Editor aufgebaut ist und starten ein erstes Programm, zeigen wie Debugging funktioniert und wie man eigene Programme schreibt. Seite 35: Leseanleitung für Anfänger:innen, Fortgeschrittene und Experten. |
| Lernen am Beispiel | Wir experimentieren mit dreizehn fertig ausprogrammierten Beispielen verschiedene Aspekte der Programmierung (Use-Modify-Create-Konzept) |
| Sprachbeschreibung | Programmierkurs der Programmiersprache NanoPy (vorher: Oxoscript). |
| Referenzen | Auszug auf den wichtigsten Klassen und Funktionen. |

Aufbau der Beispielkapitel



Einführung

Wir erklären das Experiment.



Ausprobieren

Das fertige Programme wird geladen und ausprobiert und angepasst. Hierzu muss nichts programmiert werden. Die SuS lernen den Umgang mit Konstanten/Variablen und "sehen" den Codeaufbau mit dem Debugger.



Wie funktioniert es?

Detaillierte Erläuterung des vollständiges Codes. Es wird wenig Wissen vorausgesetzt, die Texte sind jedoch anspruchsvoller. Es wird hier auch mathematische und naturwissenschaftliches Wissen eingestreut.

Experimentieren

Kleine Programmieraufgaben dienen zur Vertiefung und Festigung der neuen Konzepte.

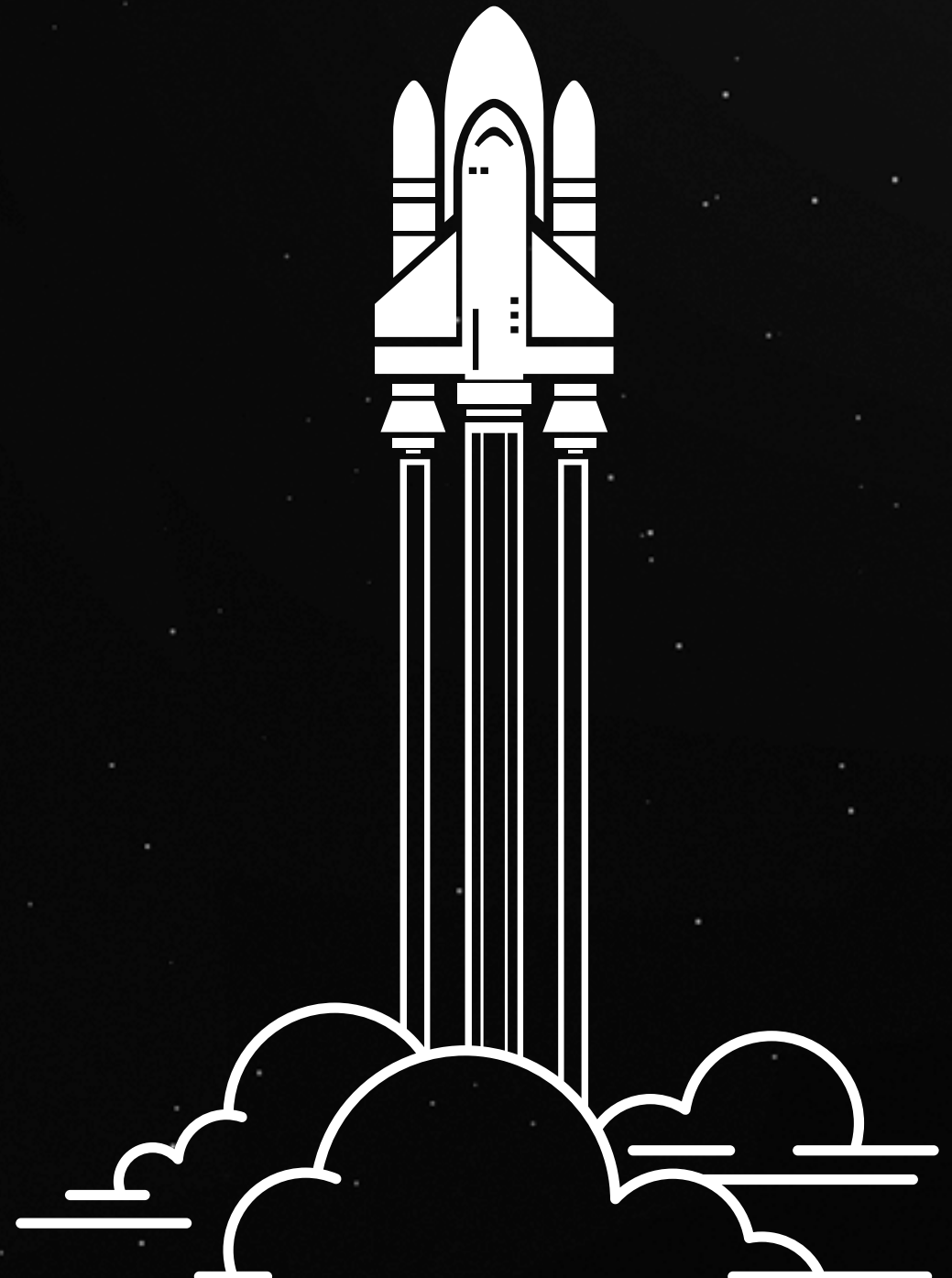
Weiterführende Informationen

Bezug zur Sprachreferenz und zu den verwendeten Funktionen. Je nach Vorwissen können hier Grundlagen erarbeitet oder vertieft werden.

Wichtig zu wissen

Zusammenfassung der wichtigsten Themen.

AUSPROBIEREN



Innovationen

| | |
|--------------------------|---|
| Use-Modify-Create | Das neue Lernkonzept "Use-Modify-Create" beginnt mit fertigen Programmen, die beobachtet, analysiert und angepasst werden können. |
| Fertige Apps | Fertige Beispielprogramme, die von Profis entwickelt wurden, können mit den eingebauten Tutorials genutzt, beobachtet, angepasst oder als Grundlage für eigene Projekte verwendet werden. |
| Constant-Editor | Mit dem Konstanten-Editor lässt sich der Code mithilfe von Schiebereglern ändern, was den Einstieg in die textbasierte Programmierung erleichtert. |
| Debugger | Mit dem integrierten Debugger können Programme während der Ausführung beobachtet und untersucht werden |
| NanoPy | Die Programmiersprache NanoPy lehnt sich stark an die weit verbreitete Sprache Python an, optimiert sie aber für kleine Geräte und Bildungsanwendungen. |
| Funktionen | Mit den optimierten Zeichen-, Sensor- und Mathematikfunktionen können beeindruckende Ergebnisse erzielt werden, die nur wenige Zeilen Code erfordern. |



FRAGEN?

