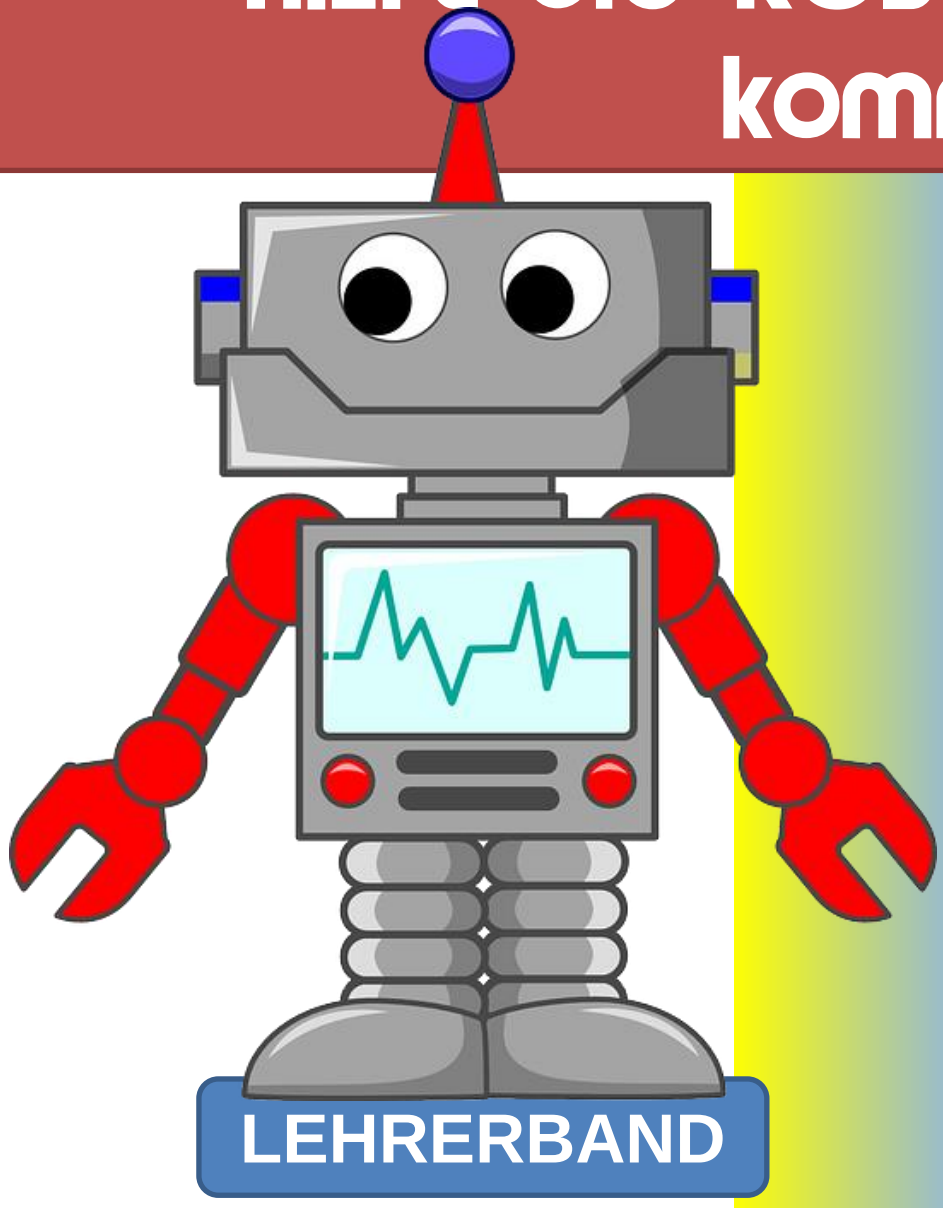
HILFE die ROBOTER kommen!



Hermann Milchram

1.2.2022

Bildnachweis:
sofern nicht anders angegeben befinden sich die Grafiken im Eigentum von NÖ Media oder sind eigene Aufnahmen und Zeichnungen oder Screenshots von Programmen.

Roboter auf der Titelseite robot-312566_640 Quelle Pixabay CC0
LEDs led-306562_640 Quelle Pixabay CC0
zielscheibe-ziel-bogenschießen-2304567 Quelle Pixabay CC0
smiley-1914523_640 Quelle Pixabay CC0
robot-1470108_640 Quelle Pixabay CC0

Herzlichen Dank für das **Korrekturlesen** an **Ilse Doppler** und **Angela Kampichler**!

INHALTSVERZEICHNIS

Programmierbare Roboter

vom Kindergarten bis zum Ende der Schulpflicht

Computertechnik be-greifen!
IT-Unterricht ab Kindergarten und Vorschulstufe?

Unsere Kinder sind heute von mehr Technik umgeben als irgendeine Generation davor. Videospiele, PCs, Tablets und Smartphones sind im Leben der meisten SchülerInnen, aber auch vieler Kindergartenkinder bereits allgegenwärtig. Das alles sind Formen von Kommunikation, die ihr Leben Tag für Tag beeinflussen. Das Verständnis für die Technik dahinter aber ist kaum vorhanden. Was im anglo-amerikanischen Raum und einigen ost- und nordeuropäischen Ländern bereits selbstverständlich ist, hält nur in sehr zögerlichem Ausmaß, meist durch Einzelinitiativen engagierter Pädagoginnen und Pädagogen, Einzug in den Alltag unserer Sprösslinge.

"*Wer aktuelle Technologien verstehen, produktiv nutzen und vielleicht sogar mitgestalten will, benötigt ein grundsätzliches Verständnis für die Art und Weise, wie die Welt der Computer funktioniert. Nur wer versteht, wie Computer „ticken", kann auch morgen noch kompetent handeln."* (c't 2014, Heft 14)

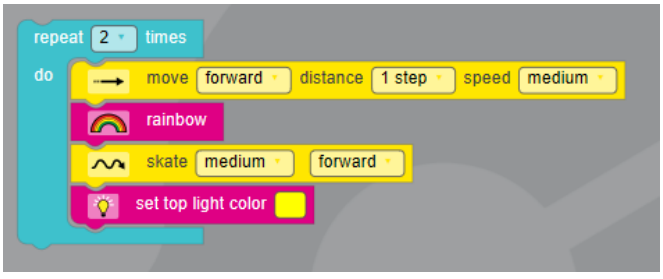
„Programm or be programmed"
(Medientheoretiker Douglas Rushkoff)

Informatikkenntnisse

- helfen beim Lösen von Problemen in allen Lebensbereichen
- vermitteln eine Reihe von Denkwerkzeugen, die sich auch auf komplexe Probleme im Alltag anwenden lassen.

Die amerikanische Computerwissenschaftlerin *Jeannette M. Wing* prägte dafür den Begriff „**Computational Thinking**", welchen sie als vierte Grundfertigkeit dem Lesen, Schreiben und Rechnen hinzufügen möchte.

Programmieren bedeutet Information in eine Form zu bringen, die von einem Computer verstanden werden kann. Programmierung spielt in vielen Bereichen unseres Alltags eine wesentliche Rolle, ohne dass wir uns dessen bewusst sind: die Programmierung der Mikrowelle, der Waschmaschine, des Backofens oder eines SAT-Empfängers sind nur einige wenige Beispiele dafür. Das heutige Programmieren hat vielfach nur mehr wenig mit dem Schreiben von Computerroutinen in Programmiersprachen wie BASIC, PASCAL oder gar ASSEMBLER zu tun. Moderne, für den Unterricht entwickelte Programmiersprachen, wie **Scratch** https://scratch.mit.edu/ oder **Blockly** https://blockly-games.appspot.com/ sind Generatoren mit einer grafischen Oberfläche, in der Programmabläufe per **DRAG&DROP** zusammengefügt werden. Es ist fesselnd, führt rasch zu Ergebnissen und macht Spaß. Trotzdem werden dabei die grundlegenden Konzepte der Programmierung vermittelt und die Fähigkeit Probleme zu lösen gestärkt. Eine zeitgemäße, auf das Berufsleben vorbereitende Ausbildung kommt am Thema Informatik nicht vorbei. Aber ab wann kann und soll informatische Bildung beginnen? Viele Gründe sprechen dafür, Informatik möglichst früh auf den Stundenplan zu



setzten. Eines der wesentlichsten Argumente ist, dass jenseits des Unterrichts Smartphones, Tablets, Computer und auch Roboter längst zum Alltag unserer Kinder gehören.

Erstmals gibt es ab dem SJ 2018/19 für die NMS und AHS Unterstufe ein verbindliches Fach „**Digitale Grundbildung**“, in dem unter anderem auch der Begriff des „Computational Thinking“ verankert ist.

Computational Thinking			
Mit Algorithmen arbeiten	– nennen und beschreiben Abläufe aus dem Alltag, verwenden, erstellen und reflektieren Codierungen (z.B. Geheimschrift, QR-Code), – vollziehen eindeutige Handlungsanleitungen (Algorithmen) nach und führen diese aus, – formulieren eindeutige Handlungsanleitungen (Algorithmen) verbal und schriftlich.	– entdecken Gemeinsamkeiten und Regeln (Muster) in Handlungsanleitungen, – erkennen die Bedeutung von Algorithmen in automatisierten digitalen Prozessen (z.B. automatisiertes Vorschlagen von potenziell interessanten Informationen).	– können intuitiv nutzbare Benutzeroberflächen und dahinterstehende technische Abläufe einschätzen.
Kreative Nutzung von Programmiersprachen	– erstellen einfache Programme oder Webanwendungen mit geeigneten Tools, um ein bestimmtes Problem zu lösen oder eine bestimmte Aufgabe zu erfüllen, – kennen unterschiedliche Programmiersprachen und Produktionsabläufe.	– beherrschen grundlegende Programmierstrukturen (Verzweigung, Schleifen, Prozeduren).	– reflektieren die Grenzen und Möglichkeiten von Simulationen.

Das Vermitteln von grundlegenden Konzepten der Informatik in Kindergarten, Grundschule und Unterstufe stellt Pädagogen jedoch vor große Herausforderungen. Da das abstrakte Denkvermögen erst langsam entwickelt wird, muss der didaktische Ansatz dahingehen, die abstrakten Konzepte der Computertechnologie im wahrsten Sinne des Wortes begreifbar zu machen. Hier können die auf den folgenden Seiten vorgestellten programmierbaren Roboter eine wichtige Unterstützung bieten.

Was bringt die Beschäftigung mit programmierbaren Robotern?

- ☑ fördert die Entwicklung von Problemlösungsstrategien
- ☑ stärkt die Fähigkeit Fehler zu finden und zu korrigieren
- ☑ fördert das analytische und logische Denken
- ☑ fördert das kollaborative Arbeiten
- ☑ stärkt die Kommunikations- und Diskussionsfähigkeit
- ☑ hilft bei der Entwicklung des räumlichen Denkens

Noch vor einigen Jahrzehnten waren Roboter bloß Science-Fiction, heute sind wir auf Schritt und Tritt von diesen modernen Helferleins umgeben. Sie bauen Autos, unterstützen Ärzte bei Operationen, entschärfen Bomben, tauchen in die Tiefen der Ozeane und erkunden für uns den Weltraum, mähen für uns den Rasen (**Mähroboter)** und übernehmen Reinigungsarbeiten im Haus (**Staubsaugroboter**). Auch die Raumfahrt ist bei ihren Missionen auf die Unterstützung von Robotern angewiesen. Auf unseren Straßen sind bereits die ersten autonom fahrenden Fahrzeuge unterwegs.

NÖ Media Streaming→**Roboter: Helfer des Menschen**
https://bit.ly/2QOGXZ0

http://www.zukunftstechnologien.info/technik-und-wirtschaft/robotik/

Roboterfreund für einsame Kinder https://bit.ly/2vQUxCg

Planet Wissen „Roboter“
https://bit.ly/2vNcTEn

AB1_Ein Blick in den Garten
Lösung

Beschreibe, was du auf dem Bild siehst.
Wozu sind die verschiedenen Teile da?

1. Welche Arbeit führt der Roboter auf dem Bild aus?
2. Wer hat dies früher gemacht?
3. Welches Werkzeug wurde dazu benötigt?
4. Wer steuert den Roboter auf dem Bild?
5. Woher weiß er, was er tun soll?
6. Kennst du noch andere Roboter?
7. Welche Arbeiten erledigen diese?
8. Erkläre mit eigenen Worten, was alle Roboter gemeinsam haben.

Was sind Roboter?

Ein Roboter ist eine Maschine, die imstande ist selbstständig Aufgaben zu erfüllen. Dazu verfügt der Roboter über definierte Schnittstellen, um seine Umwelt zu erfassen, mit ihr zu kommunizieren und sie gegebenenfalls auch zu verändern.

NÖ Media Streaming→**Was ist was: Computer und Roboter**
https://bit.ly/2DmtiG6

Filmausschnitt:
WALL-E’s “Day At Work”
https://bit.ly/1AuQl91

AB2_Was sind Roboter
Lösung

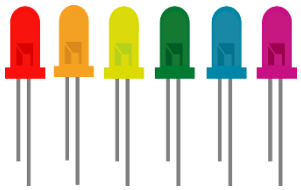
Beschreibung eines Roboters:

1. Er arbeitet nach programmierten Anweisungen.
2. Er nimmt die Umgebung mit Sensoren wahr.
3. Er kann verschiedene Tätigkeiten selbständig ausführen.
4. Er wird durch einen Prozessor (Teil vom Computer) gesteuert.

Bauteile eines Roboters

Ein Roboter braucht:

- ✓ **Sensoren**→ Lichtsensor, Temperaturfühler, Drucksensor, Lagesensor, Ultraschallsensor, Infrarotsensor etc. nehmen die Umwelt wahr und wandeln diese Informationen in elektrische Impulse verschiedener Stärke und Dauer um.
- ✓ **Aktoren** → elektrische Impulse werden in mechanische Bewegungen umgewandelt, um Motoren, Greifarme usw. zu bewegen
- ✓ **LEDs (Light Emitting Diodes)** →können sowohl als **Aktoren** (Ausgabe von Lichtsignalen in unterschiedlichen Farben) als auch als **Sensoren** zur Erkennung des Umgebungslichts verwendet werden.



- ✓ **Schnittstellen für die Kommunikation**

USB-Interface: dient der Verbindung zum Computer und wird meist auch zum Aufladen verwendet

WLAN: Erlaubt ähnlich wie Bluetooth den drahtlosen Datenaustausch zwischen verschiedenen Endgeräten und deinem Roboter.

Vorteil: Ist schneller und geht auch über größere Distanzen.

Bluetooth: Über Bluetooth können Signale drahtlos mit PCs, Tablets und Smartphones ausgetauscht werden. Bevor du Bluetooth zum Datenaustausch verwenden kannst, musst du deinen Roboter mit dem gewünschten Endgerät koppeln!

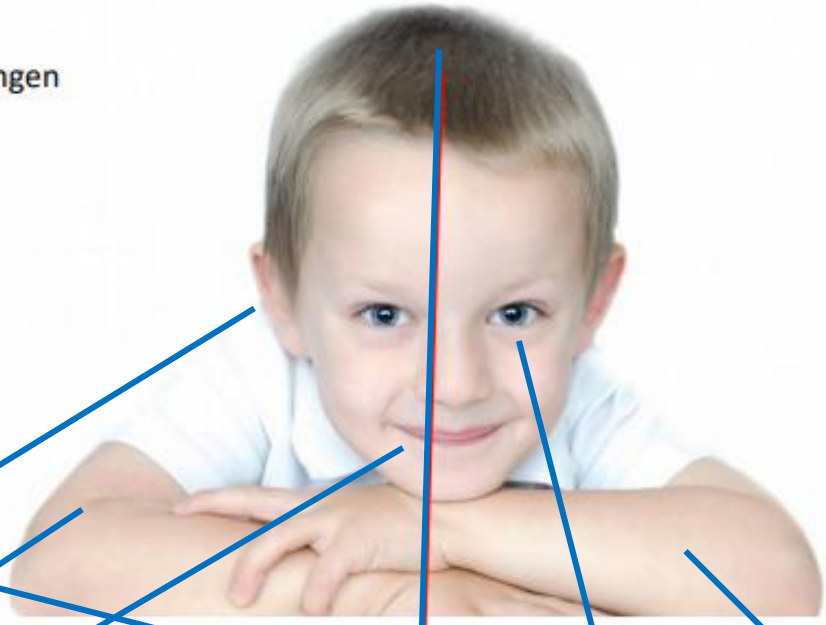
- ✓ **Mikroprozessor** → steuert die Aktoren und Sensoren
- ✓ **künstliche Intelligenz** (Software = Programme).

Roboter arbeiten nach dem **EVA**-Prinzip

AB3_Vergleich Mensch – Roboter
AB4_Von Sensoren und Aktoren

AB3_Vergleich Mensch – Roboter Lösung

Aufgabe: Verbinde die entsprechenden Bezeichnungen und Bildteile miteinander.



Ohr | Haut | Mund, Hand-zeichen | Nahrung | Gehirn | Auge | Muskeln

Daten verarbeiten	hören	sehen	spüren	sprechen, Zeichen geben	bewegen	Energie nutzen
Prozessor	Mikrofon	Licht-sensoren, Kamera	Tempera-tursensor, Druck-sensoren	Laut-sprecher, Licht-zeichen	Motoren	Akku / Batterie

Von spannenden Problemen zu kreativen Lösungen PH Luzern

AB4_Von Sensoren und Aktoren Lösung

Ein Sensor ist eine passive Komponente, er erfasst Dinge.

➔ Sensoren sind in etwa vergleichbar mit den fünf Sinnen beim Menschen.

Ein **Aktor** ist eine aktive Komponente, er **bewegt** Dinge oder **führt etwas aus**.

➔ Aktoren sind zum Beispiel vergleichbar mit dem Gehen, dem Sprechen oder der Mimik beim Menschen.

Aufgabe: Kreuze an, welche der folgenden Bauteile **Sensoren** und welche **Aktoren** sind.

Tipp: Es gibt auch Bauteile, die nichts von beidem sind.

☐ Sensor
☒ Aktor

☒ Sensor
☐ Aktor

☒ Sensor
☐ Aktor

☐ Sensor
☒ Aktor

☒ Sensor
☐ Aktor

☐ Sensor
☒ Aktor

☒ Sensor
☐ Aktor

☐ Sensor
☒ Aktor

☐ Sensor
☒ Aktor

☐ Sensor
☐ Aktor

Ein Roboter besteht aus verschiedenen Sensoren und Aktoren.

Von spannenden Problemen zu kreativen Lösungen PH Luzern

LEDs

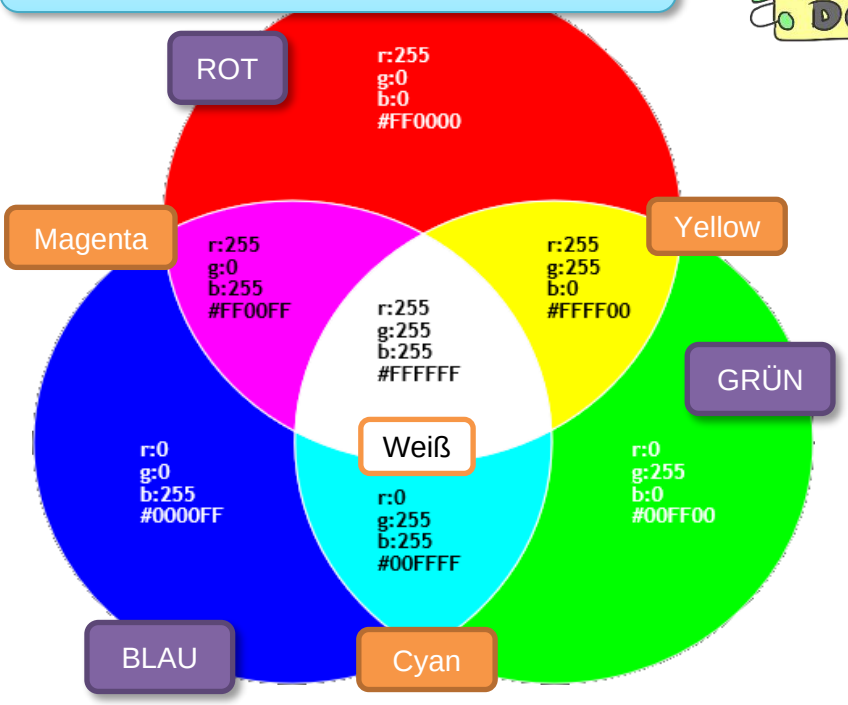
Am PC oder bei RGB-LEDs kann eine Farbe durch ihre Anteile an den drei Primärfarben **R**ot, **G**rün und **B**lau definiert werden.

Eigenschaften der Grundfarben (Primärvalenzen)

- ✓ Grundfarben können nicht aus anderen Farben gemischt werden
- ✓ Durch Mischen der drei Grundfarben können alle anderen Lichtfarben erzeugt werden

Additiver Farbmischer:
http://www.spectrumcolors.de/cor_rgb_demo.php

AB5_LEDs
Farbmischung Lösung



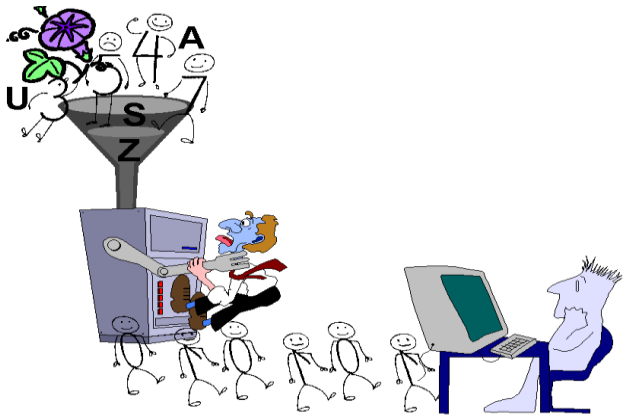
Dieses als **RGB-System** bezeichnete Verfahren baut auf der **additiven Farbmischung** auf. Jede Farbe wird durch ihren Anteil an den drei Primärfarben (**Rotwert** 0-255, **Grünwert** 0-255, **Blauviolettwert** 0-255) definiert.

Die drei Farbwerte werden jeweils durch ein **Byte** = 8 Bits dargestellt und können somit die Werte von 0 bis 255 annehmen. Wird eine Grundfarbe für die Darstellung des Farbtons nicht benötigt, beträgt der entsprechende Zahlenwert 0. Da jede der drei Grundfarben in 256 Stufen (0 - 255) dargestellt werden kann, sind insgesamt 256^3 = 256 x 256 x 256 = 16 777 216 unterschiedliche Farben darstellbar.

Die meisten Farbdrucker verwenden die Farben **CYAN**, **MAGENTA**, **YELLOW**
→**Subtraktive Farbmischung** http://www.spectrumcolors.de/cor_cmyk_demo.php

Wie arbeiten Computer

Die Vorstellung, dass **Computer** oder **Roboter** Informationen (über Tastatur eingegeben oder über Sensoren wahrgenommen) verstehen, ist eigentlich falsch. Die Verarbeitung der Daten beruht auf Zahlzeichen, **Zahlensystemen**, Rechengesetzen, die mit rasender Geschwindigkeit im **Mikroprozessor** verarbeitet werden. Die in unserem Alltag verwendeten Schrift- und Zahlensysteme sind aber für die Arbeit mit dem Computer nicht geeignet. Letztendlich müssen alle Informationen für den Computer oder einen Roboter in eine Folge von nur zwei verschiedenen Zahlen (0 und 1) übersetzt werden. Schon **Gottfried Wilhelm Leibnitz** fand im Jahr 1679 erstmals die besondere Eignung des **Zweiersystems** (Binärsystem, Dualsystem) für die maschinelle Verarbeitung von Zahlen. George Boole (1815 - 1864) baute, ausgehend von den logischen Möglichkeiten „**wahr** (true)-**falsch** (false)“, die Grundlagen für die heutige Informatik auf. (**Boolesche Algebra**)

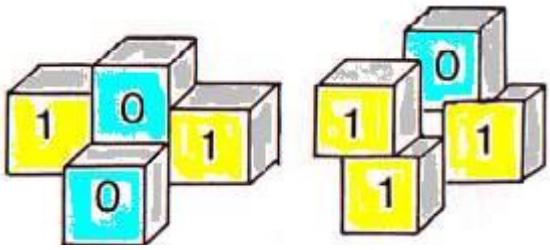


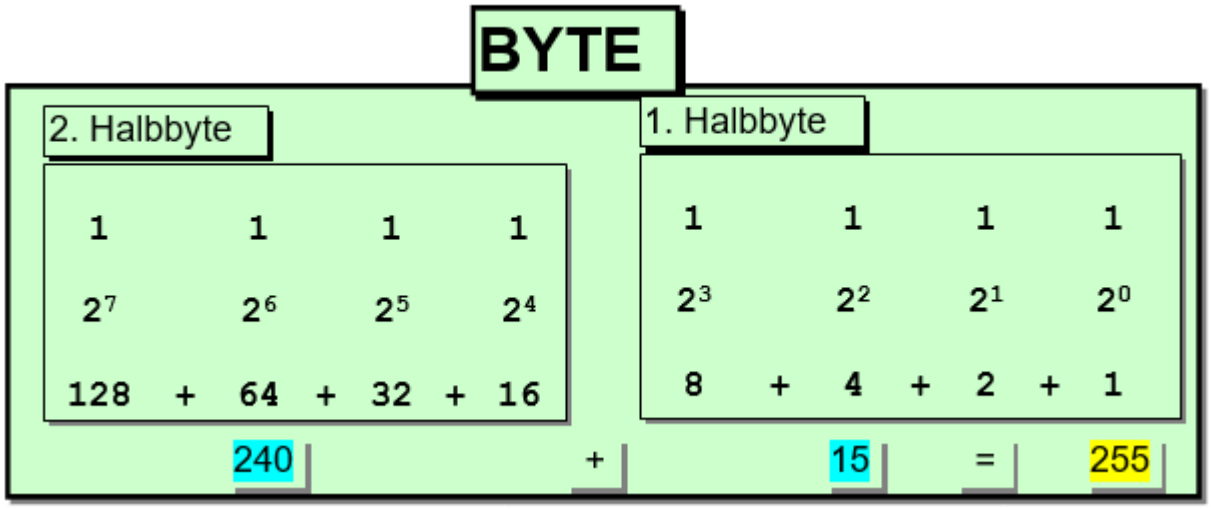
Dualsystem (Binärsystem)

Bit = **binary digit**, kleinste Informationseinheit (bestehend aus 0 oder 1)

Im Computer entspricht ein Bit einem Schaltvorgang: **Strom fließt** oder **Stromkreis unterbrochen**.

Ketten von Bits werden zur Darstellung von Buchstaben und größeren Zahlen benutzt. Mit **8 Bits** = 1 **Byte** können dadurch 2^8 = 256 Zahlen (0 - 255) dargestellt werden.



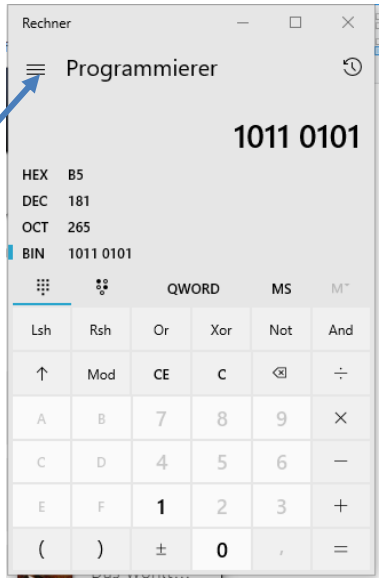


Bei der Umrechnung von Binärzahlen in Dezimalzahlen werden die **Stellenwerte** aller mit **1** belegten Stellen addiert!

z.B. **1011 0101** = $2^7+2^5+2^4+2^2+2^0$ = 128 + 32+ 16 +4 + 1 = **181**

Tipp: Schneller geht die Umrechnung zwischen den verschiedenen Zahlensystemen mit dem **Windows-Rechner**!

Hier kannst du zwischen den verschiedenen **Modi des Taschenrechners wechseln**! Für die Umwandlung zwischen den verschiedenen Zahlensystemen muss der Modus „**Programmierer**“ aktiviert werden!



AB6_Eine binäre Geschichte

AB_5 Eine binäre Geschichte Lösung

✓ I was born in year **1770** in Bonn.

✓ I became a professional musician at the age of **11**.

✓ I lived in Vienna since **1794**.

✓ I wrote my first symphony in **1800**.

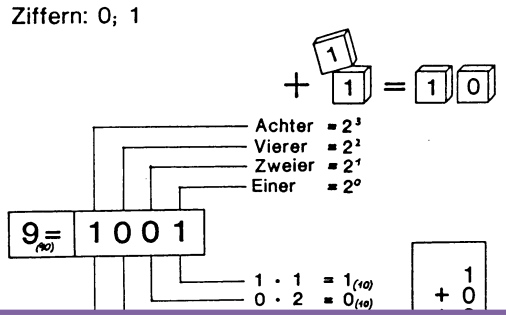
✓ The third symphony „EROICA“ was written in **1804**.

✓ When I died in **1827** I was years **56** old.

My name is **Ludwig van Beethoven**.

Umrechnung von Dezimalzahlen in Binärzahlen

Zur Umwandlung einer Dezimalzahl in eine **Binärzahl** wird die Dezimalzahl durch die Basis (beim Dualsystem 2) dividiert. Das Ergebnis der Division wird ohne Berücksichtigung des Restes erneut dividiert. Dieser Vorgang wird so lange wiederholt, bis das Ergebnis Null ist. Die **Divisionsreste** (Überlauf) bilden **von rechts nach links angeschrieben** die gesuchte Binärzahl.



Binäre Zahlen werden wie dezimale Zahlen erzeugt. Anstelle von 10 Ziffern (0,1,2 ...9) gibt es aber nur 2 Ziffern (0,1) Bei jeder Erweiterung um eine Stelle verdoppelt sich der Stellenwert.

Dieser Vorgang, bei dem die Reste der Division ermittelt werden, wird als **Restwertdivision (Modulo = mod)** bezeichnet!

9 mod 2 = 1;

12 mod 7 = 5

16 mod 8 = 0

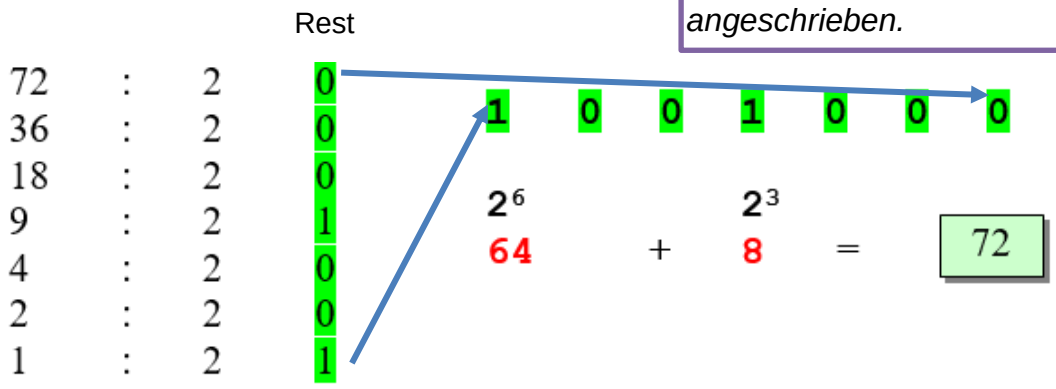
72 mod 2 = 0

AB7_Dezimalzahl in Binärzahl umwandeln

In den meisten **Programmiersprachen** wird für die **Modulo-Funktion** das Zeichen „**%**“ verwendet.

Umwandlung der Dezimalzahl 72 in eine Binärzahl

Die einzeln ermittelten Restwerte werden anschließend, beginnend mit dem zuletzt ermittelten Restwert, in einer Reihe angeschrieben.



AB_6 Dezimalzahl in Binärzahl umrechnen Lösung

134 % 2 = 0 **134** = BIN **10000110** = $2^1 + 2^2 + 2^7$ = 2 + 4 + 128 = 134

67 % 2 = 1
33 % 2 = 1
16 % 2 = 0
8 % 2 = 0
4 % 2 = 0
2 % 2 = 0
1 % 2 = 1

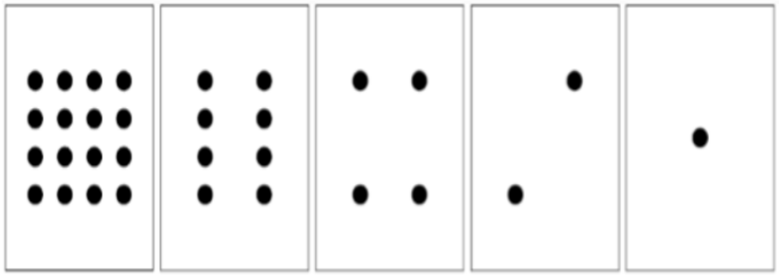
229 % 2 = 1 BIN **11100101** = $2^0 + 2^2 + 2^5 + 2^6 + 2^7$ =
1 + 4 + 32 +64 +128 = **229**

114 % 2 = 0
57 % 2 = 1
28 % 2 = 0
14 % 2 = 0
7 % 2 = 1
3 % 2 = 1
1 % 2 = 1

AB8_Binäres zählen

Bevor das Arbeitsblatt „AB7_Binäres zählen“ verteilt wird, ist es sinnvoll sein das Prinzip den SchülerInnen zunächst einmal zu demonstrieren. Dazu werden fünf Karten, welche auf der einen Seite mit Punkten (2^0=**1**, 2^1=**2**, 2^2=**4**, 2^3=**8**, 2^4=**16**) bedruckt sind, während sie auf der anderen Seite leer sind. Fünf Kinder werden nach vorne gebeten. Die Karten werden an die Kinder in der folgenden Reihenfolge verteilt.

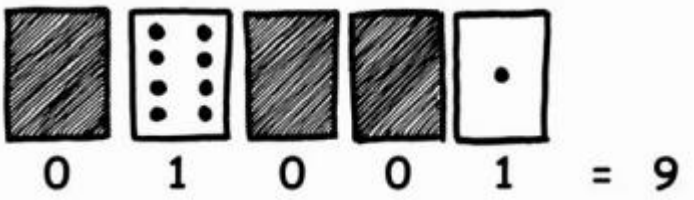
Links → rechts



Fragen an die SchülerInnen:

- ✓ Was fällt euch an der Anzahl der Punkte auf den Karten auf?
  Jede Karte enthält jeweils doppelt so viele Punkte wie die vorherige Karte
- ✓ Wie viele Punkte müssten also auf den nächsten Karte auf der linken Seite sein, wenn wir noch weitere Karten hinzufügen würden? 2^5=**32**, 2^6=**64**, 2^7=**128**, →8 Karten entsprechen einem 8Bit-System, damit können die Zahlen 0 -255 (256 Zahlen = 2^8) dargestellt werden.

Wir können diese Karten verwenden um Zahlen darzustellen, indem wir gewisse Karten drehen, sodass deren Punkte sichtbar sind (dies schreiben wir als 1), oder wir lassen sie, sodass keine Punkte sichtbar sind (wir schreiben eine 0). Wir zählen die Anzahl der sichtbaren Punkte. Fragen Sie die Kinder, wie man die Zahl 6 darstellen kann (4 Punkte und 2 Punkte), danach 15 (8, 4, 2 und 1 Punkt), danach 21 (16, 4 und 1).



- ✓ Versuchen Sie nun die Klasse von null hochzählen zu lassen. Der Rest der Klasse soll sorgfältig zuschauen und versuchen das Muster zu erkennen.
  Jede Karte dreht halb so oft wie die Karte rechts davon.

AB_7 Binäres zählen Lösung

- ✓ Wie stellt man **3** = **00011** **12** = **01100** und **19** = **10011** dar?
- ✓ Gibt es mehrere Möglichkeiten eine Zahl zu bilden? **Nein**
- ✓ Gib die größte **31** bzw. kleinste Zahl **0** an, die du auf diese Weise mit den 5 Karten bilden kannst?
- ✓ Gibt es Zahlen zwischen der größten und der kleinsten Zahl, die man nicht erzeugen kann? **Nein**

Quelle: https://classic.csunplugged.org/wp-content/uploads/2015/11/binary_CSunplugged-german-staub.pdf

Andere Zahlensysteme in der EDV

Beim Binärsystem enthält jedes **Halb-Byte** 16 Kombinationsmöglichkeiten. Die meisten modernen Datenverarbeitungsanlagen verwenden daher heute das **Hexadezimalsystem** (Sedezimalsystem). Dieses System arbeitet mit der Zahl 16 als Basis. Mit diesem System ist es daher möglich, ein Halb-Byte durch ein einziges Zeichen darzustellen.

Dezimal Basis 10	Binär Basis 2	Hexadezimal Basis 16
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10		A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

239

2. Halbbyte	1. Halbbyte
1 1 1 0	1 1 1 0
14	**15**
E	**F**

Keine Buchstaben sondern **SEDEZIMAL - Zeichen**!

Die dezimale Zahl **239** heißt im Dualsystem **1110 1111**. Im Hexadezimalsystem reichen für deren Darstellung die beiden Sedezimalzeichen **EF** aus.

Welche dezimale Zahl wird durch **9D** dargestellt?

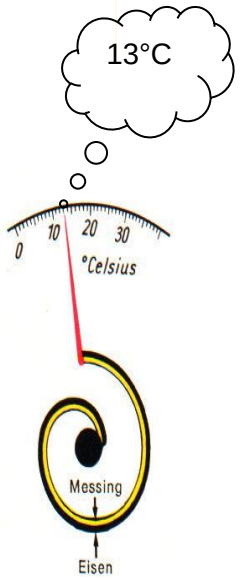
Hexdadezimal	Binär	Dezimal
9D	**1001 1101**	$2^7+0+0+2^4+2^3+2^2+0+2^0=128+16+8+4+1=$**157**

AB_9 Hexadezimalsystem Lösung

Hexdadezimal	Binär	Dezimal
89	**1001 1101**	$2^7+0+0+2^4+2^3+2^2+0+2^0=128+16+8+4+1=$**157**
2A	**0010 1010**	$0 + 0 +2^5+ 0 +2^3 + 0 + 2^1+ 0 = 32+8+2 =$ **42**
CD	**1100 1101**	$2^7 + 2^6+ 0 + 0 +2^3+ 2^2 +0 + 2^0 = 128 + 64 +8 + 4 +1 =$ **205**

ANALOG – DIGITAL

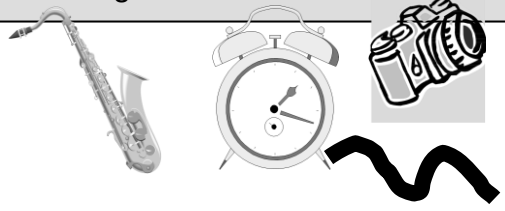
Der Mensch hat seit jeher versucht alles genau zu messen und zu qualifizieren. Nicht alles aber ist so leicht in Zahlen festzuhalten wie z.B. eine Herde Schafe auf einer Weide. Viele Vorgänge in der Natur lassen sich nur unvollkommen in zählbaren Vorgängen ausdrücken. Denken wir z.B. nur an ein Gefäß mit Wasser, das erhitzt wird. Der Temperaturanstieg ist ein kontinuierlich verlaufender Vorgang der durch Zahlen (Tabelle, Thermometer mit Digitalanzeige) nur ausschnittsweise wiedergegeben werden kann. Verwendet man zur Darstellung ein Flüssigkeitsthermometer oder ein **Bimetall Thermometer**, so kann man am Anstieg der Flüssigkeitssäule bzw. an der Krümmung des Bimetall Streifens den kontinuierlichen Anstieg, analog zum tatsächlichen Anstieg der Temperatur, genau mitverfolgen. Versucht man jedoch den Wert auf der Skala abzulesen (13°C), so erhält man nur einen, für einen Moment gültigen digitalisierten Näherungswert. Die Genauigkeit des abgelesenen Wertes hängt von der Genauigkeit der Skala ab. Betrachtet man die Skala mit einer Lupe, könnte man feststellen, dass der Wert etwa bei 13,2° C liegt. Je feiner unterteilt die Skala ist, desto genauer wäre der abzulesende Wert. Trotzdem stellt die bestimmte Größe immer nur einen Näherungswert dar.



Mit zunehmender Erwärmung biegt sich der **Bimetallstreifen** immer weiter. Der Vorgang verläuft kontinuierlich, **analog** zum tatsächlichen Temperaturanstieg.

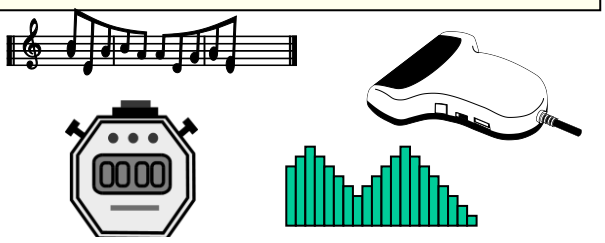
Analog

Unendliche Menge von nicht immer eindeutig unterscheidbaren Zeichen, die eine **wirklichkeitsgetreue Darstellung** der Umwelt ermöglichen.



Digital

Endliche Menge eindeutig unterscheidbarer Zeichen, die uns eine **Momentaufnahme** eines Zustandes wiedergeben.



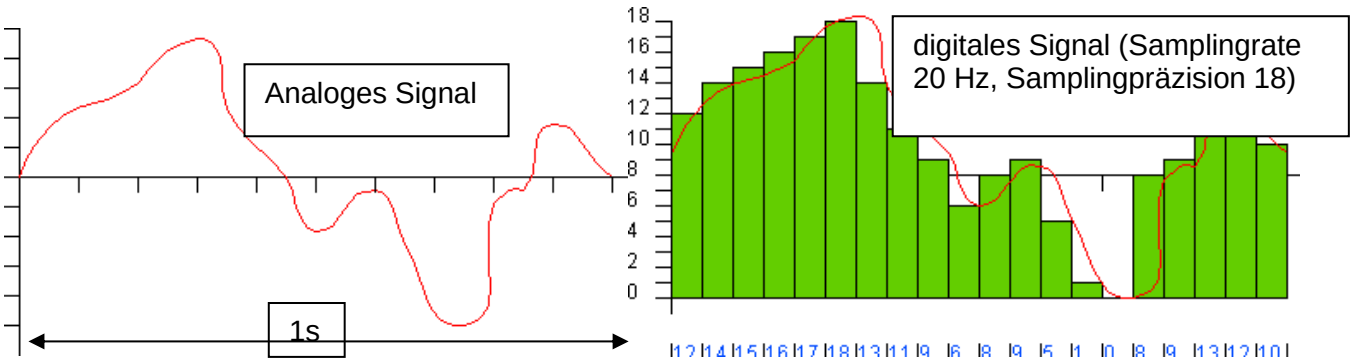
Wir Menschen treffen oft sehr unklare Aussagen: „ein bisschen, etwas, ein wenig ...“ Herkömmliche elektronische Schaltungen können mit solchen analogen Entscheidungen nichts anfangen.

Die meisten Wahrnehmungen, die der Mensch mit seinen Sinnen (Auge, Ohr, Nase ...) aufnimmt, sind solche analogen Informationen. Digitale Informationen dagegen bedienen sich eines endlichen, abzählbaren Zeichenvorrates (Ziffern, Buchstaben, ASCII-Tabelle ...)

Zur Umwandlung von digitalen Signalen in analoge Signale und umgekehrt werden **Digital-Analog Umsetzer** (DA-Wandler) verwendet.

Umwandlung analoger Signale

Analoge Signale müssen zur Weiterverarbeitung im Computer digitalisiert werden! Der **Analog-Digital-Wandler** (auch AD-Wandler) tastet nun nach und nach die Welle ab und merkt sich für jeden Moment eine Zahl, die angibt, wie hoch der Wellenberg ist. Die verschiedenen AD-Wandler unterscheiden sich dabei durch ihre sogenannte **Samplingrate**, also wie oft pro Sekunde sie abtasten, und die **Samplingpräzision**, also wie genau eine Zahl ermittelt wird.



Beispiel für DA- und AD-Wandlung

- ✓ Ein **Scanner** verwandelt die analogen Bildinformationen einer Fotografie in ein digitales Muster für den PC.
- ✓ Ein **Modem** verwandelt die Binärdaten eines PCs in analoge Signale zur Übertragung im herkömmlichen Telefonnetz
- ✓ Mit einer **Soundkarte** können analoge Aufnahmen von Kassetten und Schallplatten digitalisiert und auf CDs gebrannt werden.
- ✓ Die **Sensoren** in einem Roboter wandeln die empfangenen Signale in elektrische Impulse unterschiedlicher Stärke um.

Fuzzy Logic

Es gibt heute aber bereits schon Computersysteme, die mit Fuzzy Logic arbeiten. Durch die Kombination der Fuzzy Logic und neuronaler Netze sind beispielsweise lernfähige Systeme möglich, die mit „unpräzisen“ Informationen arbeiten könnten. Die Möglichkeit, dass eine Variable einen bestimmten Wert annimmt, wird mit einem Wahrheitsgrad (Zugehörigkeitswert) zwischen 1 (wahr) und 0 (falsch) angegeben. In der **Fuzzy Logic** wird das Ergebnis einer Operation mit einem gewissen Wahrscheinlichkeitswert errechnet. Neben den herkömmlichen Werten „wahr“ oder „falsch“ kann ein Ergebnis beispielsweise noch die Werte „wahrscheinlich wahr“ „möglicherweise wahr“, „möglicherweise falsch“ und „wahrscheinlich falsch“ annehmen.

Fuzzy bedeutet so viel wie „undeutliche Logik“, eine Form der Logik, die in einigen Expertensystemen und anderen Anwendungen der Künstlichen Intelligenz (KI) eingesetzt wird.

Programmierung

Unter Programmieren versteht man die Festlegung einer Arbeitsvorschrift (**Programm**), die eine automatische Verarbeitung durch den Computer ermöglicht.

Ein **Programm** ist eine **zeitliche und logische Abfolge von Anweisungen**. Wir begegnen solchen Abläufen fast überall in unserem Leben: zB Fernsehprogramm, Waschprogramm, Bedienungsanleitungen, Bastelanleitungen …

AB10_Was ist ein Programm

In der Computerwissenschaft ist eine **Programmiersprache** eine künstliche Sprache zur Aufstellung von Befehlsfolgen, die schließlich von einem Mikroprozessor abgearbeitet werden können.

Verschiedene Arten von Programmiersprachen

Eine Programmiersprache kann entweder der Arbeitsweise eines Prozessors angepasst sein
→ **maschinenorientierte Expertensprache** (zB ASSEMBLER)

Ein **Assembler** ist ein Programm, das in **Assemblersprache** geschriebene Anweisungen in **Maschinensprache** übersetzt!

Code in Assemblersprache:

342B:0l00	B84000	MOV	AX,0040
342B:0103	8ED8	MOV	DS,AX
342B:0105	B83412	MOV	AX,1234
342B:0108	A37200	MOV	[00721],AX
342B:OlOB	EAOOOOFFFF	JMP	FFFF:0000

Dasselbe Programm in Maschinensprache (Binärschreibweise):

```
10111000 01000000 00000000 10001110 11011000 10111000 00110100 00010010
10100011 01110010 00000000 11101010 00000000 00000000 11111111 11111111
```

oder sich an dem zu lösenden Problem orientieren
→ **anwenderorientierte Problemsprache** (zB Python, C#, C, C++, Visual Basic, JavaSkript. PHP, ..)

```
// ConsoleOutput.cs

class ConsoleOutput
{
   static void Main()
   {
     System.Console.WriteLine();
     System.Console.WriteLine("Hello world!");
   }
}
```

Zusätzlich gibt es noch eine Gruppe von hochentwickelten Spezialsprachen, die einem besonderen Anwendungsbereich z.B. Datenbankabfragen (**SQL**) dienen.

```
SELECT COUNT(CustomerID), Country FROM Customers GROUP BY Country;
```

Grundelemente einer modernen Programmiersprache

Die wesentlichen Elemente jeder modernen Programmiersprache sind:

- ✓ **Funktionen**/Methoden
- ✓ **Variablen**/Objekte
- ✓ **Befehle**/Anweisungen
- ✓ **Operatoren** (+, - , /, *, <, > …)
- ✓ **Entscheidungen** und **Schleifen** (ITERATIONEN)
- ✓ **Kommentare** (dienen lediglich der Beschreibung, werden nicht vom Prozessor ausgeführt!)

Arten der Programmierung

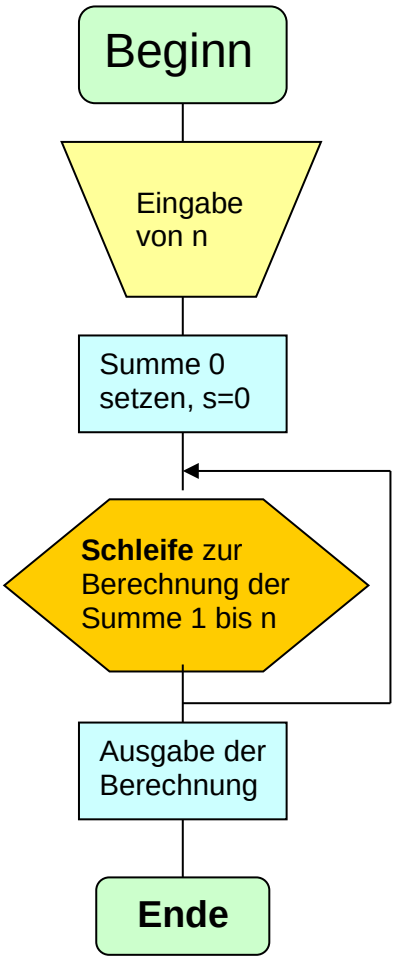
- ✓ **Strukturierte Programmierung**
  Zerlegungen von Programmen in Unterprogramme (Teilprogramme, Funktionen, Methoden). Die einzelnen Anweisungen im Hauptprogramm und in den einzelnen Teilprogrammen werden der Reihe nach (sequentiell) abgearbeitet. Durch Entscheidungen (IF … THEN … ELSE) und Schleifen (ITERATIONEN), kann die Programmausführung gestoppt oder gewisse Abläufe wiederholt werden.

- ✓ **Objektorientierte Programmierung (OOP)**
  Moderne Betriebssysteme bzw. graphische Benutzeroberflächen wie Windows sind wahre Programmmonster und bestehen meist aus vielen Millionen Zeilen kompliziertester Quellcodes. Windows unterstützt zum Beispiel die unterschiedlichsten Hardwaresysteme, Drucker, Sprachen und Anwendungsgebiete. Normalerweise müsste ein Entwickler all diese verschiedenen Möglichkeiten bei der Programmierung berücksichtigen. Der Aufwand wäre so groß, dass niemand Programme entwickeln würde. Das Zauberwort, um die Programmierung unter Windows zu vereinfachen heißt **OOP** (Objektorientierte Programmierung). Kernpunkt dieses Konzepts ist das **Black-Box-Prinzip**. Das bedeutet, dass der Programmierer Funktionen von Windows nutzen kann, ohne zu wissen, wie sie eigentlich funktionieren.

  Die Aufgaben eines Programmierers unter Windows beschränken sich dabei auf drei Grundgebiete:

Eigenschaften	vergeben und ablesen
Ereignisse	abfangen und darauf reagieren
Methoden	auf Objekte anwenden

Programmablaufplan (PAP)



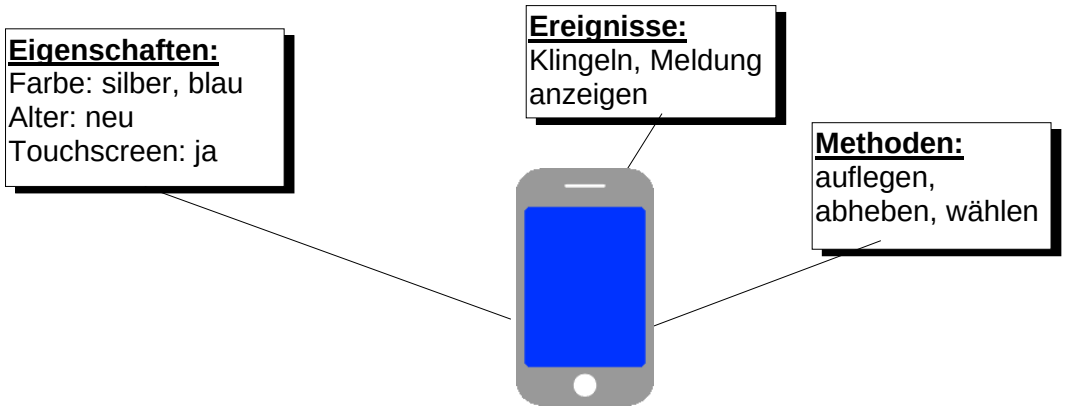
Objekte:
Objekte sind sichtbare oder unsichtbare Elemente eines Programmes (Programmfenster, Schaltflächen, Textfelder, Bildfelder ...), die bestimmte Eigenschaften aufweisen.

Eigenschaften
Eigenschaften sind einstellbare Werte, die das Erscheinungsbild oder die Funktionsweise von Objekten festlegen. Je nach Art und Aufgabe haben verschiedene Objekte auch unterschiedliche Eigenschaften (Schriftart, Schriftgröße, Hintergrundfarbe, Text, Ausrichtung...). Die Eigenschaften der Objekte werden vom Programmierer entweder beim Programmentwurf oder während der Laufzeit des Programms, als Folge eines Ereignisses, verändert.

Ereignisse
Ein Ereignis ist eine Aktion, die von einem Objekt erkannt wird (zB Klicken auf eine Schaltfläche, Eingabe eines Textes in ein Textfeld,...). Aufgabe des Programmierers ist es nun festzulegen, wie auf ein bestimmtes Ereignis reagiert werden soll. Bei der prozessorientierten Programmierung (POP) gibt es keinen durchgehenden Programmcode mit definierten Start- und Endpunkten.

Methoden:
Methoden sind Aktionen, die sich auf ein Objekt beziehen (Löschen eines Textes oder einer Grafik, Drucken, Aktualisieren eines Bildschirminhaltes, Verschieben eines Objektes).

Ein Handy oder Smartphone stellt für die meisten Menschen eine solche Blackbox dar. Jeder von uns benutzt es, die wenigsten aber wissen, wie es wirklich funktioniert. Eine solche Blackbox, deren Innenleben man nicht kennt, nennen wir **Objekt.** Jedes Objekt besitzt bestimmte **Eigenschaften** (Properties) und löst Ereignisse aus, auf die man durch geeignete Methoden reagieren kann.



- ✓ **Prozessorientierte Programmierung** (Ereignisorientierte Programmierung) **POP**
  Bei der ereignisorientierte Programme wird nicht mehr eine Liste von Befehlen nach einem festen Algorithmus abgearbeitet. Stattdessen wird eine Sammlung Prozeduren und Funktionen verwendet. Der Aufruf einzelner Funktionen wird durch bestimmte Ereignisse (Tastatureingaben, Mausklicks, Berührungen am Touchscreen, Rückmeldung von Sensoren) ausgelöst! Solche Programme verfügen also nicht über ein geplantes, festes Ablaufschema, sondern der Ablauf wird mit bei Benutzung durch die Anwender oder durch Sensoren die auf die Umgebung reagieren, neu bestimmt.

Typische Programmierfehler

- ✓ **Syntaxfehler**: Eingegebene Befehle werden vom Computer nicht erkannt.
- ✓ **Bedienungsfehler**: zB, Drucker nicht eingeschaltet, Sensoren falsch angeschlossen, ..
- ✓ **Logische Fehler**: Unlogischer Programmaufbau führt oft zu einem völlig unsinnigen Programmablauf.

Phasen der Programmentwicklung

- ✓ Genaue Beschreibung der **Aufgabenstellung**
- ✓ **Problemanalyse**
- ✓ **Programmstrukturierung** (Welche Daten sind erforderlich, erforderliche Variablen und Konstante, Algorythmen aufstellen, …)
- ✓ **Codierung** (Quelltext erstellen)
- ✓ **Programmtest** (Prüfung auf Syntaxfehler, logische Fehler, Testläufe um Fehler zu erkennen)
- ✓ **Programmdokumentation** (Programmbeschreibung, Kommentare zu Variablen und Konstanten …

Visuelle Programmiersprachen (VPL)

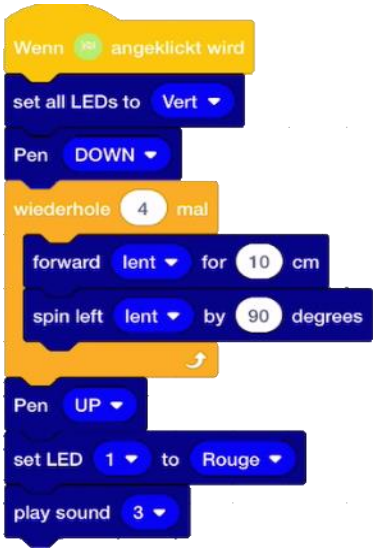
Da die textorientierte Programmierung professioneller Programmiersprachen für die meisten SchülerInnen abschreckend wirkt verwenden wir für die Programmierung der Roboter visuelle Programmiersprachen (**VPL**), die uns das Erlernen von der grundlegen Funktionen moderner Programmiersprachen erleichtern.

Visuelle Programmiersprachen benutzen vorgefertigte Programmelemente, meist in Form von Blöcken, die per Drag&Drop wie Puzzleteile zu einem Programm zusammengefügt werden können.

Die Blöcke sind so gestaltet, dass sich nur auch vom logischen Ablauf passende teile aneinanderfügen lassen. Meist erleichtert eine farbliche Kennzeichnung der einzelnen Blöcke die Orientierung bei der Auswahl der richtigen „Puzzleteile“

Zwei der am häufigsten verwendeten blockorientierten, visuellen Lernsprachen sind **Scratch** und **Blockly**. Vom Prinzip her sind VPLs Programmgeneratoren die aus den ausgewählten und zusammengefügten Programmblöcken einen Quellcode in einer professionellen Programmiersprache (Java Skript, Python, …) generieren.

Diese per **Drag&Drop** in **Scratch** zusammengesetzten Programmblöcke generieren im Hintergrund den nebenstehend Quellcode in **Java-Skript**.




```
//When play button pressed
setAllLedsColor("green");
pen("DOWN");
var cmf = 4;
for(var mvz = 0; mvz < cmf; mvz++)
{
  forwardForDistance("02", "10");
  spinLeftByDegrees("02", "90");
}
pen("UP");
setLedColor("00", "red");
playSound("3");
```

AB11_Visuelle Programmierung (VPL)

Scratch ist eine visuelle Programmiersprache, deren Ziel es ist, bereits Kinder mit den Prinzipien der Programmierung vertraut zu machen.

(Bildquelle: Flickr CC BY 2.0, Wesley Fryer)

*„**Blockly** ist eine Bibliothek, die Webanwendungen, Android- und iOS-Apps einen visuellen Code-Editor hinzufügt. Der Blockly-Editor verwendet verzahnte, grafische Blöcke zur Darstellung von Programmierkonzepten wie Variablen, logischen Ausdrücken, Schleifen und mehr. Es ermöglicht den Benutzern, Programmierprinzipien anzuwenden, ohne sich Gedanken über die Syntax machen zu müssen. Die abgebildeten Algorithmen können u. a. als **JavaScript**- oder **Python-Code** exportiert werden“.*

(https://de.wikipedia.org/wiki/Blockly)

AB12_Prüfe dein Wissen
AB13_Das Magisches Robotik Quadrat

AB_11 Prüfe dein Wissen Lösung

(1) Welche Bauteile eines Roboters nehmen Informationen aus der Umwelt war?
- ☐ Bluetooth-Schnittstelle
- ☐ Motoren
- ☒ Sensoren
- ☐ LEDs

(2) Welcher Bauteil eines Roboters steuert die Motoren und Sensoren?
- ☐ USB-Interface
- ☒ Mikroprozessor
- ☐ Aktoren
- ☐ Bluetooth Schnittstelle

(3) Welche Bauteile in einem Roboter wandeln empfangene Signale in elektrische Impulse um?
- ☐ Aktoren
- ☒ Sensoren
- ☐ Motoren
- ☐ Rezeptoren

(4) Welches der folgenden Zahlensysteme wird nicht für die Verarbeitung der Daten im Computer verwendet?
- ☐ Dualsystem
- ☐ Hexadezimalsystem
- ☒ Dezimalsystem
- ☐ Binärsystem

(5) Das Byte:
- ☒ 1 Byte = 8 Bit.
- ☐ 1 Byte = die kleinste Informationseinheit.
- ☐ 1 Byte entspricht der Größe eines Gebissabdrucks in einem frischen Hamburger.
- ☐ Es gibt 256 von ihnen.

(6) Welche der folgenden Aussagen stimmen? (mehrere Antworten sind richtig)
- ☐ Ein Bit sind 8 Byte
- ☒ Ein Byte ist eine Einheit zur Angabe der Größe eines Datenträgers
- ☒ Ein Bit ist die kleinste Informationseinheit
- ☒ Ein Bit ist der Speicherplatz für eines der beiden Zeichen: 0 oder 1

(7) Welcher der folgenden Begriffe ist keine Programmiersprache?
- ☐ Blockly
- ☐ Scratch
- ☒ Android
- ☐ Python

(8) Gottfried Wilhelm Leibnitz erkannte bereits 1679 die besondere Eignung des Dualsystems für die maschinelle Verarbeitung! ☒ true ☐ false

(9) Die Boolsche Algebra bildet die Grundlage der heutigen Informatik! ☒ true ☐ false

(10) Bluetooth und WLAN sind Schnittstellen zur drahtlosen Datenübermittlung! ☒ true ☐ false

(11) Ein Bit besteht aus zwei Halbbytes! ☐ true ☒ false

(12) Mit einem Byte können 256 Zahlen dargestellt werden! ☒ true ☐ false

(13) Roboter arbeiten nach dem EVA-Prinzip! ☒ true ☐ false

(14) Wie heißt in der Programmierung eine Kontrollstruktur, die das mehrfache Notieren von Befehlen erspart! **Schleife (ITERATION)**

AB_12 Das magische Robotik Quadrat

Trage jeweils die Zahl der richtigen Antwort in das Kästchen bei der Frage ein!
(1) Fuzzy Logic, **(2)** EVA-Prinzip **(3)** LEDs **(4)** Sedezimalzeichen **(5)** Hexadezimalsystem **(6)** George Boole **(7)** Mikroprozessor **(8)** Künstliche Intelligenz **(9)** Blue tooth, **(10)** Sensoren **(11)** AD-Wandler **(12)** Restwertdivision (modulo) **(13)** binary digit **(14)** Binärsystem **(15)** Byte **(16)** USB-Interface

16 Schnittstelle für die Kommunikation mit einem Roboter?	**3** Welcher Bauteil eines Roboters kann sowohl **Sensor** und **Aktor** sein?	**2** Nach welchem Prinzip arbeiten Roboter und Computer?	**13** Was bedeutet die Abkürzung „**bit**“?
5 Welches **Zahlensystem** findet in modernen Rechenanlagen vor allem Verwendung?	**10** Welche Bauteile eines Roboters wandeln empfangene Signale in **elektrische Impulse** um?	**11** Wie heißt ein elektronischer Bauteil, der **analoge** Signale in **digitale** Informationen umwandelt?	**8** Wofür steht die Abkürzung „**KI**“
9 Schnittstelle zur **drahtlosen Datenüberttragung**.	**6** Wer baute ausgehende von den beiden logischen Möglichkeiten „**true**“ und „**false**“ die Grundlagen der heutigen Informatik auf?	**7** Bauteil eines Roboters zur Steuerung der **Aktoren** und **Sensoren**.	**12** Wofür wird in den meisten Programmiersprachen das Zeichen „**%**“ verwendet?
4 Wie heißen die Buchstaben A-F, die im **Hexadezimal-system** zum Darstellen von Zahlen verwendet werden.	**15** Wie heißt eine Kette aus **8 Bits**, die zur Darstellung von Zahlen und Buch-staben verwendet wird?	**14** Andere Bezeichnung für **Dualsystem**.	**1** Wie heißt die **Logik**, die in der künstlichen Intelligenz und bei Experten-systemen verwendet wird?.

Antworten richtig eingetragen, so ergeben die Summen der Spalten, Zeilen und der Diagonalen jeweils die Zahl 34!

1. Ozobot bit und Ozobot evo

https://ozobot.com Alter: **7+**

Ozobot evo ca. 130€ https://bit.ly/3eiWrBi

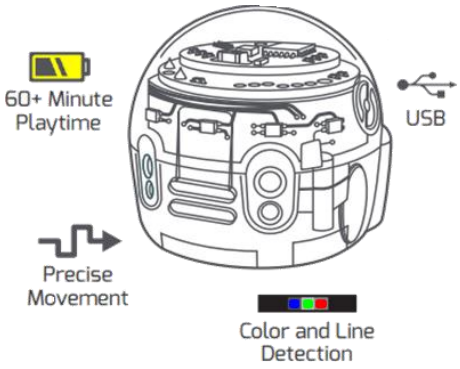
OzoBot: *H. Milchram 2017*

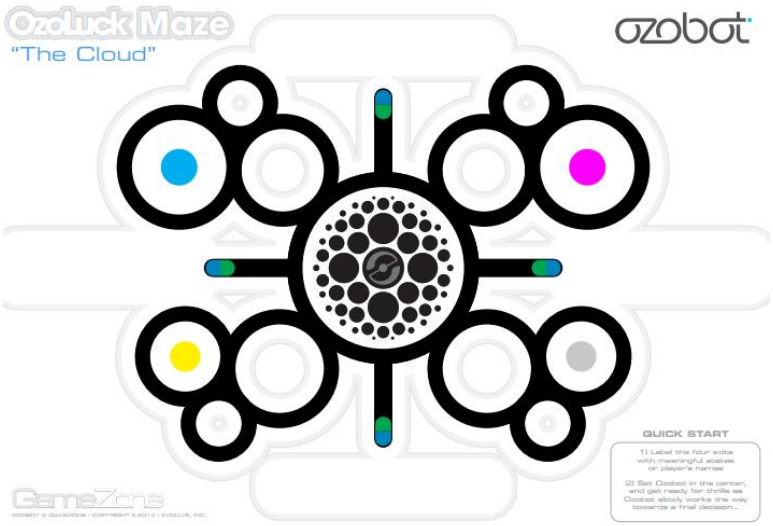
Ozobot bit und **Ozobot evo** sind zwei Mini-Roboter (∅ < 3cm), die auf Formen und Farben reagieren. Das können gemalte Linien auf Papier, gedruckte auf einem Spielplan oder virtuelle auf einem Tablet- oder Smartphone-Bildschirm sein. Mit Hilfe von Farben und Formen werden verschiedene Kommandos (fahren, anhalten, abbiegen oder sich um die eigene Achse drehen) codiert.
http://ilearnit.ch/de/ozobot.html

Ozobot Projektideen

Hochschule Schwyz:
http://bit.ly/2pdekcL

PH-Baden
http://eis.ph-noe.ac.at/ozobot0/



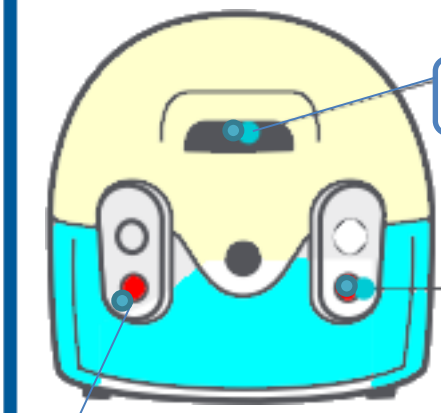


Der **Ozobot evo** besitzt zusätzlich eine **Blutooth** Schnittstelle zur Datenübertragung und **Infrarotsensoren** um Hindernisse zu erkennen. Außerdem hat er sieben **LEDs**, die man alle getrennt in verschiedenen Farben leuchten lassen kann. Ein **Lautsprecher** zur Ausgabe von Geräuschen und Tönen ist ebenso vorhanden.

Unterschiede zwischen Ozobot bit und Ozobot evo: https://bit.ly/2C8J86C

Getting started with Ozobot evo: **https://bit.ly/2MGxeFH**

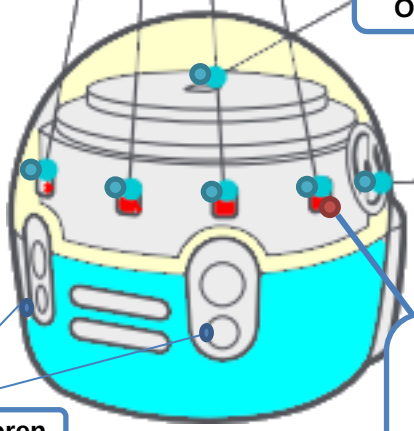
Ozobot kennenlernen



Micro USB Anschluss

Infrarot Sensor

Infrarot Sensor



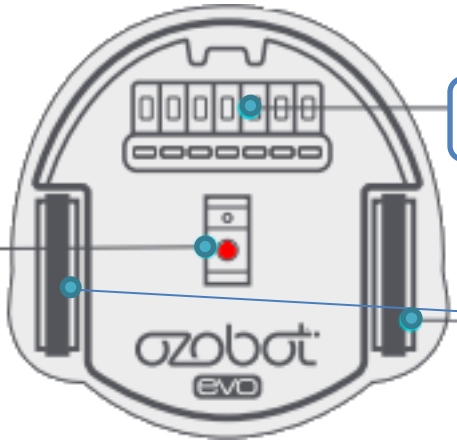
5 Front-LEDs

1 LED an der Oberseite

Knopf zum Ein- und Ausschalten

LED beim Startknopf, leuchtet rot, wenn der Ozobot aufgeladen werden muss!

Infrarot-Sensoren



Sensoren zur Linienverfolgung

Farbsensor

Antriebsräder

Arbeiten mit dem Ozobot

Dein **Ozobot evo** kann auf verschiedene Arten gesteuert werden:

- ✓ gezeichnete Linien und Farbcodes
- ✓ Drei fix einprogrammierte Modi (Tricks) können durch Abdecken der 4 Infrarot-Sensoren ausgewählt werden (davonlaufen, nachlaufen und Musik machen) https://www.youtube.com/watch?v=MMuy6xlzBl8
- ✓ Programmierung mit **Ozoblockly** im Browser
- ✓ Programmierung und Spiele mit der **Ozobot evo-APP** für Android und iOS

Evo by Ozobot

Available on the App Store

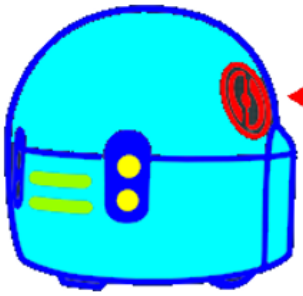
Google play

Keinesfalls die sich drehenden Räder blockieren, da sonst die Kalibrierung der Motoren gestört und eventuell sogar die Mechanik durch das Brechen von Zahnrädern kaputt gehen könnte!

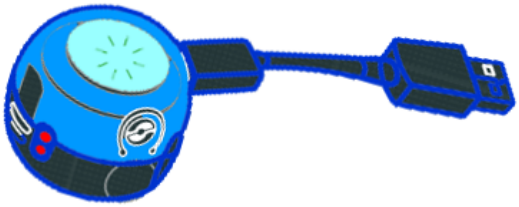
Ozobot AB1 Lösung

Ozobot einschalten

Der Ozobot hat nur einen einzigen **Knopf auf der linken Seite**, der zum Ein- und Abschalten dient.



Einschalten: kurz drücken, LEDs beginnen zu leuchten
Ausschalten: etwas länger drücken



USB-Ladegerät, PC oder **Notebook**

Dein Ozobot besitzt zur **Stromversorgung** einen **AKKU**. Sollte er leer sein (**rot** blinkendes Licht), muss er zunächst über ein **USB-Kabel** aufgeladen werden. Der **Micro USB-Anschluss** befindet sich auf der **Rückseite**! Während des Ladevorgangs blinken die LEDs grün. Sobald dein Ozobot vollständig aufgeladen ist, leuchten die LEDs konstant grün.

Die **Akkulaufzeit** beträgt etwa 1 – 1,5 Stunden!

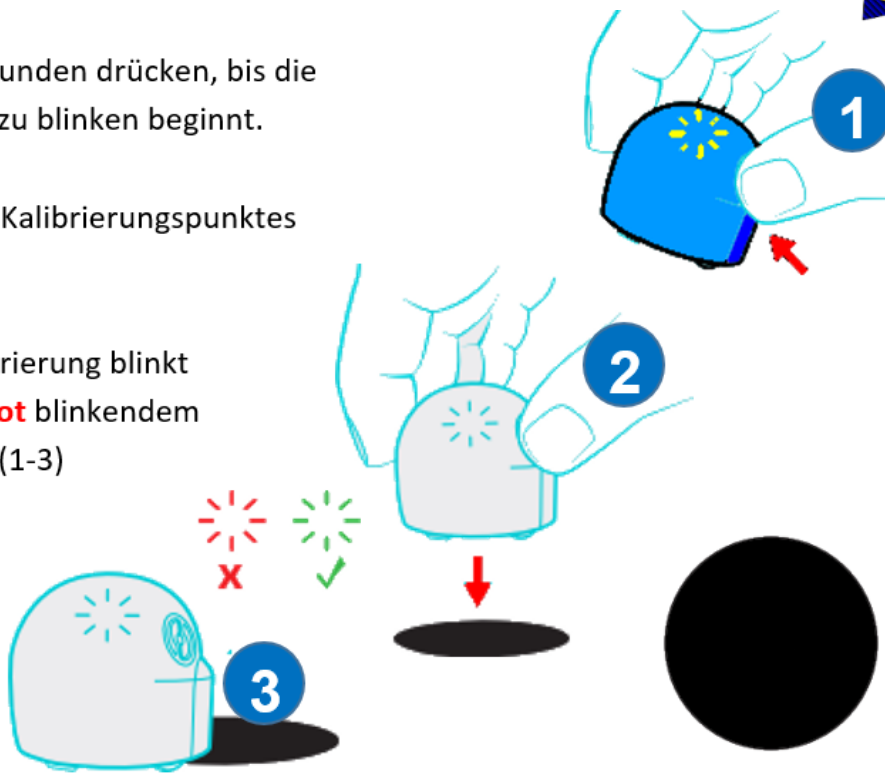
Die Sensoren deines Ozobots müssen nach dem Einschalten mit ihrer Umgebung vertraut gemacht werden. Dieser Vorgang wird als **Kalibrierung** bezeichnet.

Ozobot kalibrieren

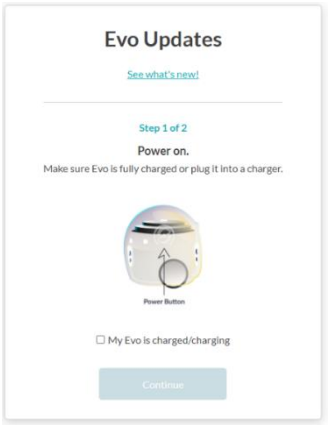
1. Einschaltknopf ca. 2 Sekunden drücken, bis die obere LED-Lampe **weiß** zu blinken beginnt.

2. Ozobot in die Mitte des Kalibrierungspunktes setzen und loslassen.

3. Nach erfolgreicher Kalibrierung blinkt dein Ozobot **grün**. Bei **Rot** blinkendem Licht muss der Vorgang (1-3) wiederholt werden!



Ozobot Firmware updaten

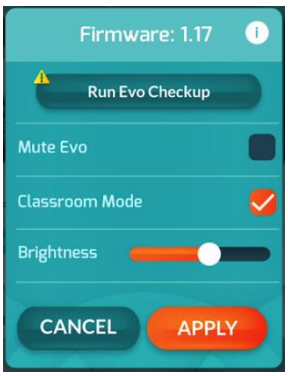


2 Möglichkeiten →Bluetooth muss eingeschaltet sein!

(1) Öffne die **Website** https://ozobot.com/support/evo-update und folge den Anweisungen →Bluetooth muss eingeschaltet sein!

(2) Verwendung der **Ozobot evo APP**

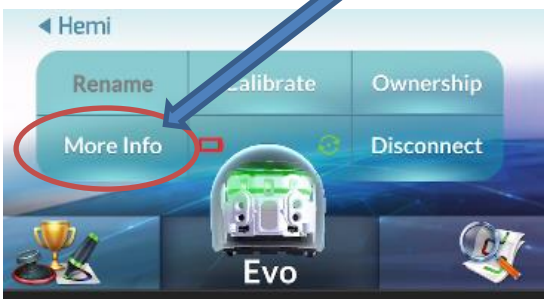
Settings→More Info →Run Evo Checkup



Ozobot Klassenraum-Modus (Education Mode)

Im Klassenraum-Modus werden einige Fähigkeiten des Ozobot evo (soziale Aspekte, Plauderei, Animationen→Tricks) deaktiviert.

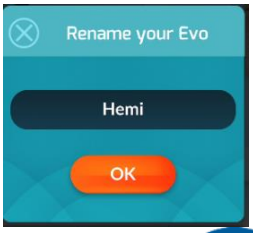
- ✓ Zum Aktivieren/Deaktivieren des Klassenraum-Modus (**Education Mode**) ist die Ozobot evo APP erforderlich.
- ✓ Öffne die APP (Bluetooth muss aktiviert sein!) und folge den Anweisungen am Bildschirm.
- ✓ Sobald dein Ozobot verbunden ist, kannst du mit Hilfe des Zahnrades  in die **Einstellungen** (Settings) wechseln und hier den **Klassenraum-Modus aktivieren**!
  Alternativ kann der Klassenraum-Modus (**Education Mode**) auch über die Schaltfläche **Settings→„More Info“** aktiviert werden!



Ozobot benennen

Bei der gleichzeitigen Verwendung mehrerer Ozobots ist es sinnvoll, jeden mit einem eigenen Namen zu versehen. Den Namen des Ozobot siehst du, sobald du die Ozobot App gestartet hast. Um den Namen ändern zu können brauchst du einen **Account**. Die Erstellung eines Accounts erfolgt direkt in der „Ozobot evo“-APP.

- ✓ Ozobot evo APP öffnen und anmelden (Bluetooth muss aktiviert sein!)
- ✓ Ozobot, der verbunden werden soll, auswählen
- ✓ Schaltfläche **Settings→„Rename“** auswählen
- ✓ Gewünschten Namen eingeben und mit **OK** bestätigen

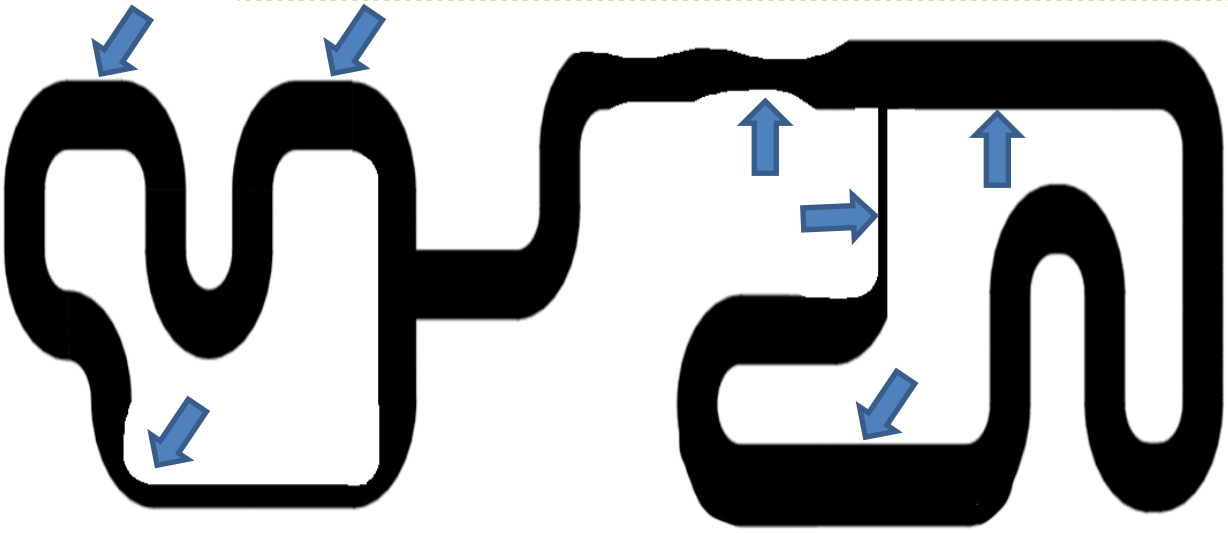


Ozobots programmieren – Linienverfolgung und Farbcodes

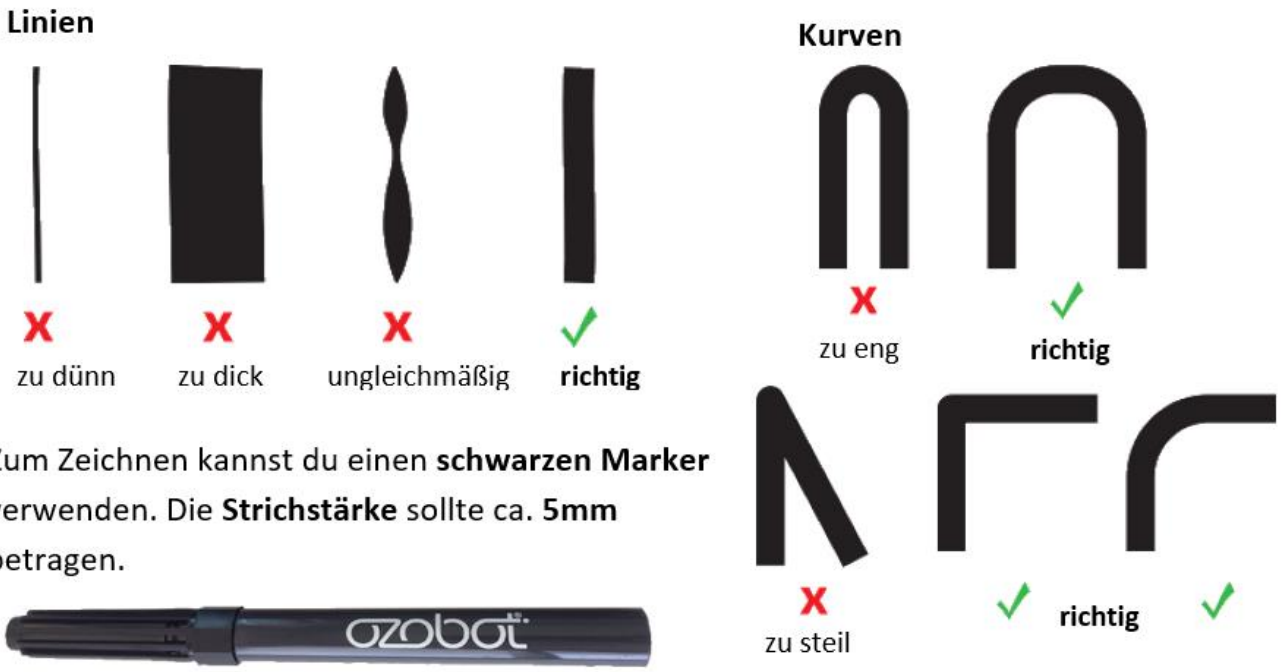
Dein Ozobot ist so gebaut, dass er nach dem Einschalten und Absetzen auf einer weißen Fläche, sofort mit der Suche nach **schwarzen Linien** beginnt. Sobald er eine gefunden hat, fährt er dieser nach. An Kreuzungspunkten entscheidet er zufällig, in welche Richtung er weiterfährt.

Ozobot AB2a Lösung

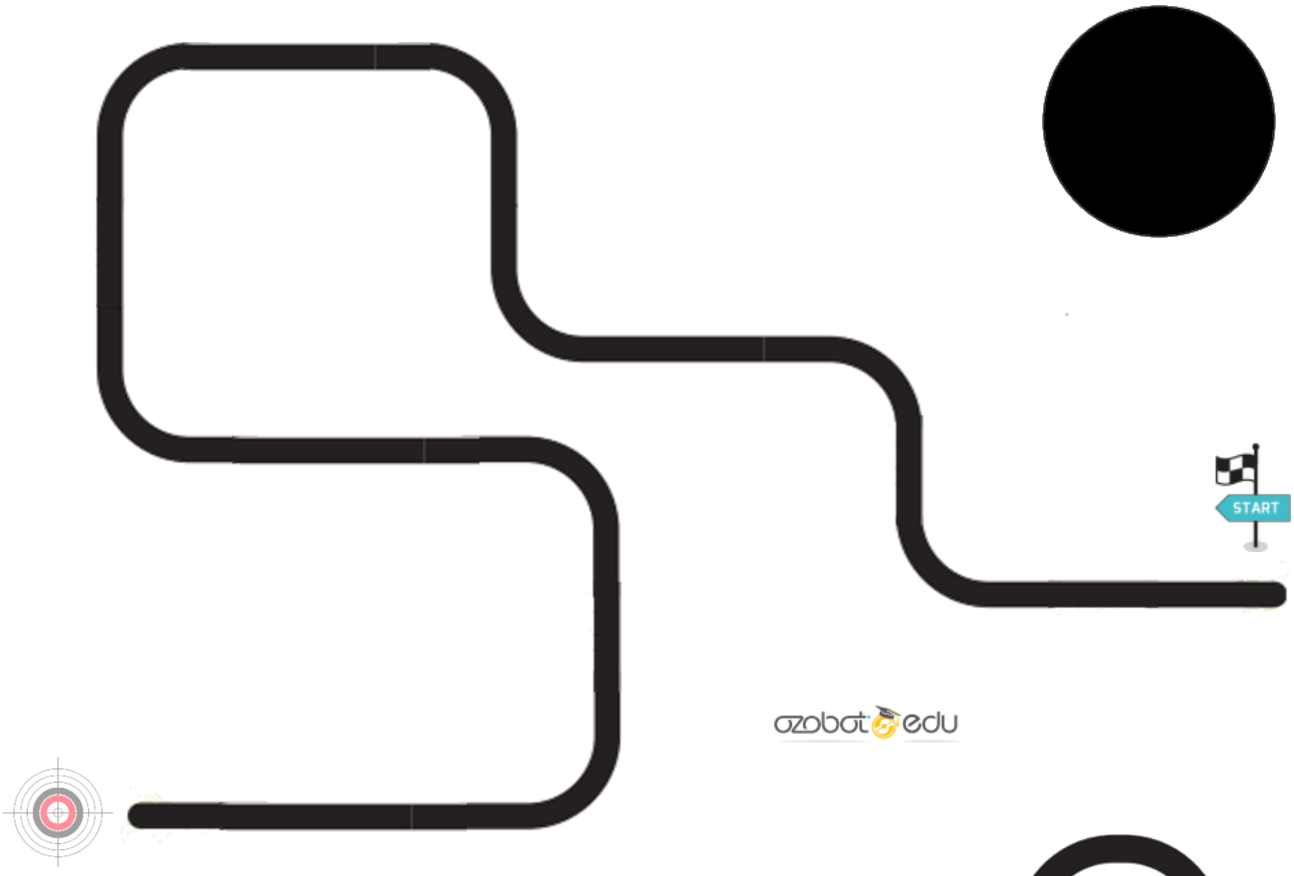
Bei zu **dick** oder zu **dünn** gezeichneten Linien verliert dein Ozobot die Orientierung. Auch zu **enge** oder zu **spitze Kurven** verursachen Probleme.

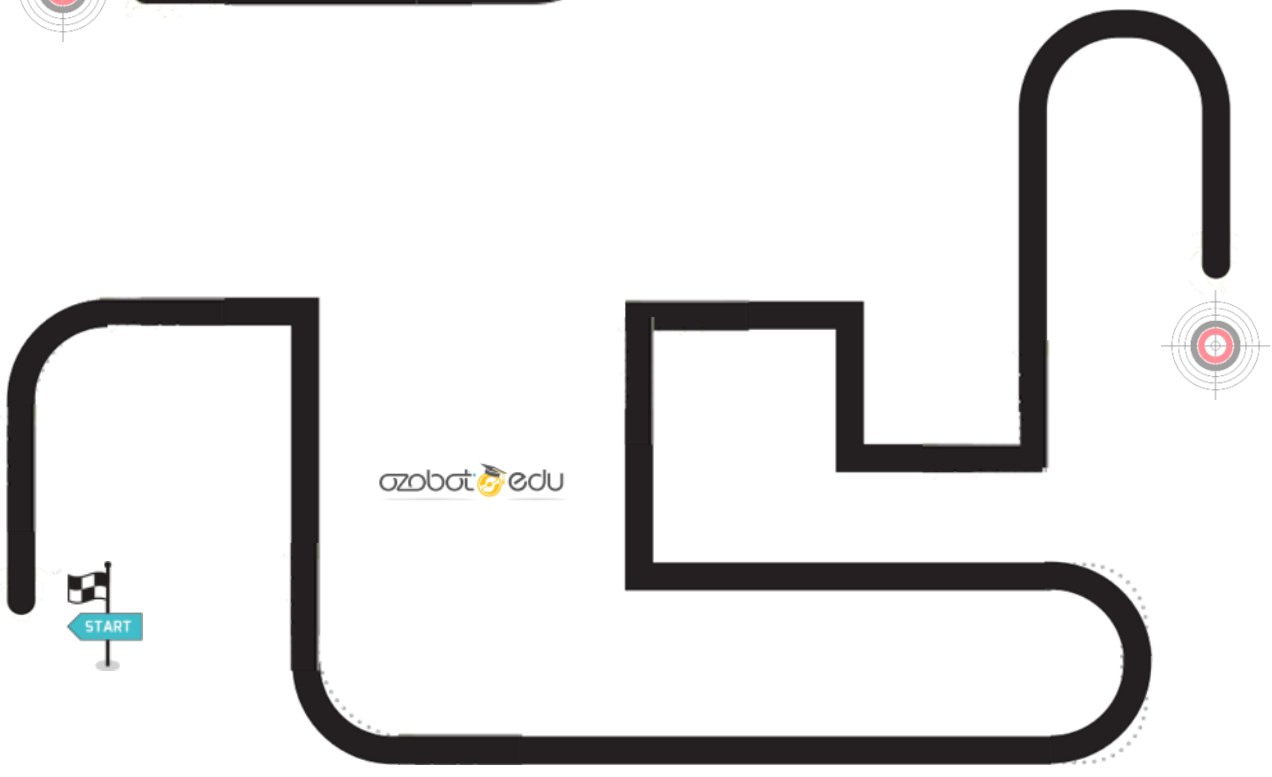


Ozobot AB2b Lösung



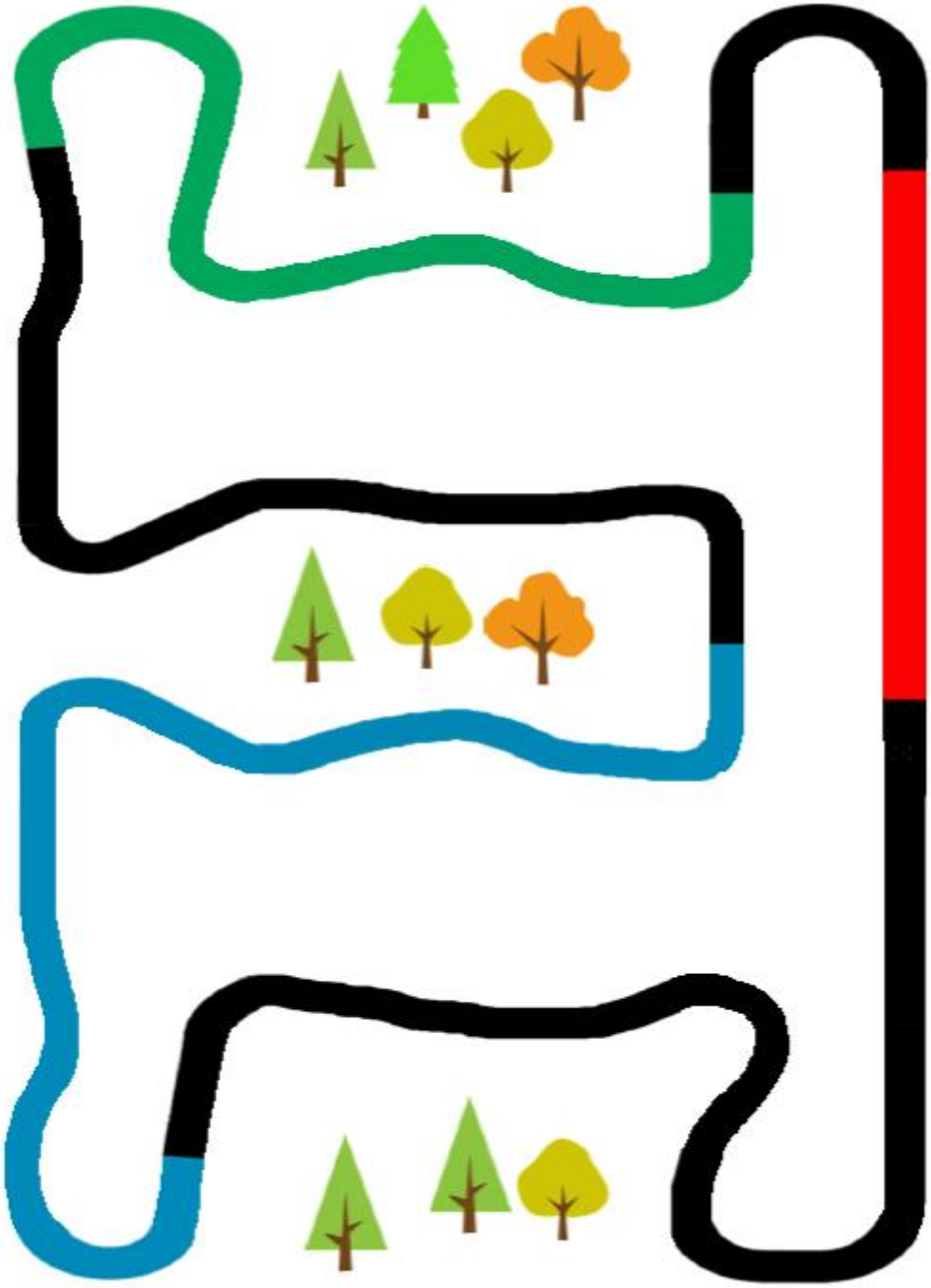
Zum Zeichnen kannst du einen **schwarzen Marker** verwenden. Die **Strichstärke** sollte ca. **5mm** betragen.





Ozobot AB4 Lösung

Sobald dein Ozobot auf eine andersfarbige Linie trifft, beginnen die LEDs in dieser Farbe zu leuchten!



Dein Ozobot erkennt nicht nur Farben, sondern kann auch spezielle **Farbcodes** interpretieren. Damit hast du die Möglichkeit z.B. Geschwindigkeitsänderungen oder Richtungswechsel durchzuführen.

Ozobot Farbcodes: Regeln für die Verwendung

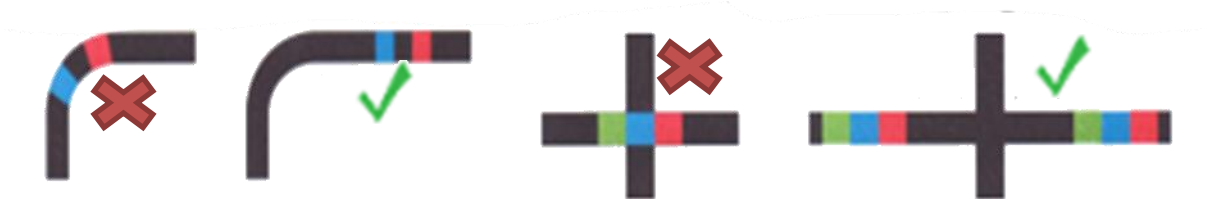
Dein **Ozobot** kann mit **Ozocodes**, einer besonderen Programmiersprache programmiert werden, die aus kurzen **Farbsequenzen** (**rot**, **blau**, **grün**) auf **schwarzen Linien** besteht.  Erkennt dein Ozobot eine Farbsequenz, führt er bestimmte, vorprogrammierte **Aktionen** aus oder ändert sein **Aussehen** (LEDs leuchten in unterschiedlichen Farben).

Ozobot AB5 Lösung

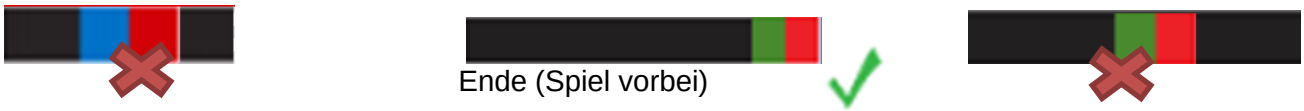
(1) Farbcodes müssen auf schwarzen Linien eingefügt werden!

(2) Keine Überschneidungen, Lücken oder verschiedene Längen der Codes!

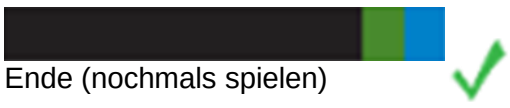
(3) Die Farbcodes dürfen nicht zu nahe an Kurven und Verzweigungen liegen!



(4) Zweifarbige Codes können nur am Ende einer schwarzen Linie stehen!



Ende (Spiel vorbei)



Ende (nochmals spielen)

Umdrehen

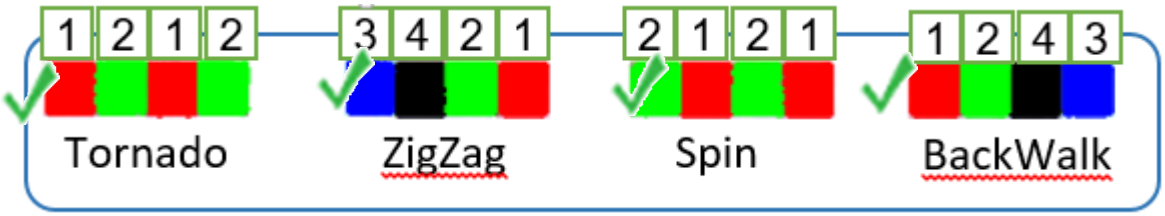
FARBCODES

Geschwindigkeit:

Richtung ändern an einer Kreuzung

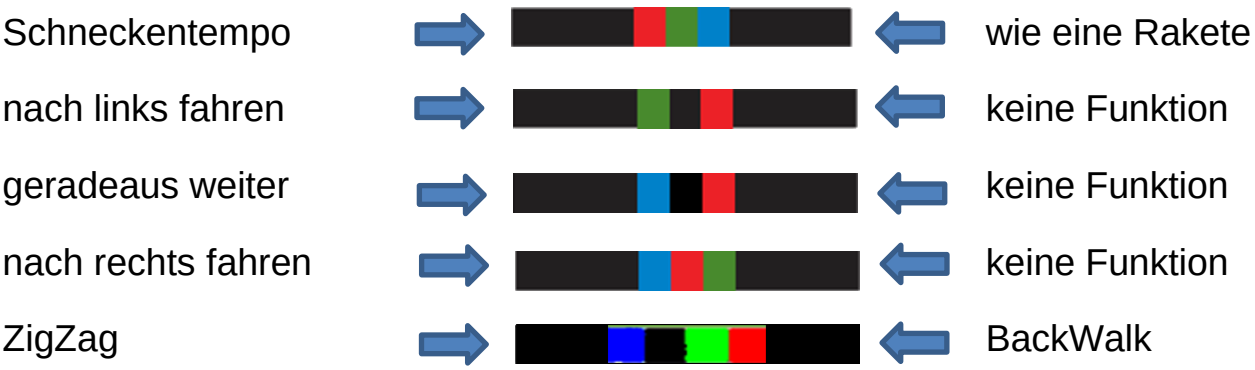
Linie suchen

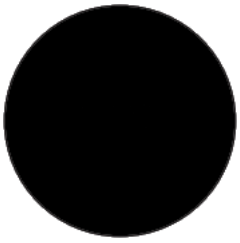
Umdrehen (180° Drehung)

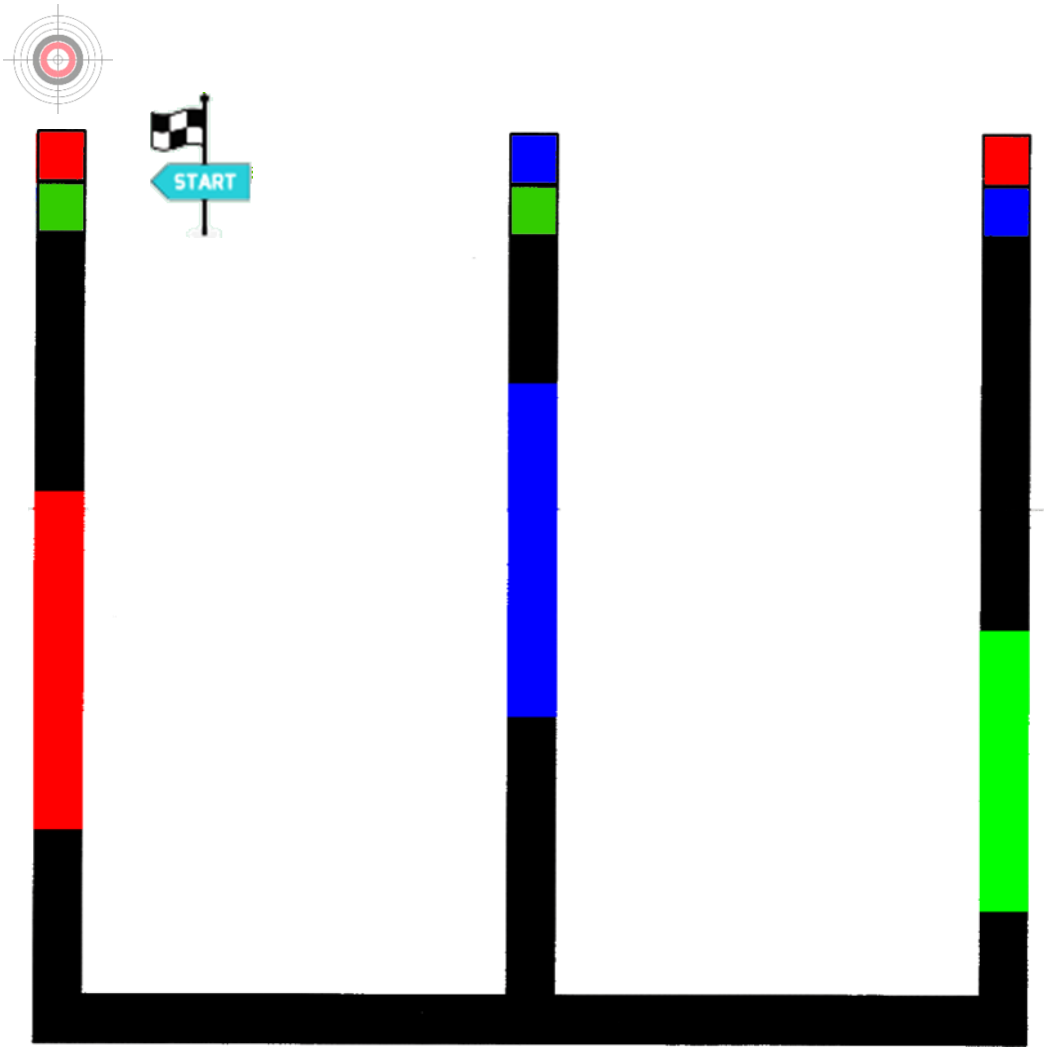


Manche Farbcodes haben je nach Richtung, von der aus sie gelesen werden, eine andere Bedeutung. zB **Tornado** →  ← **Spin**

Ozobot AB6 Lösung

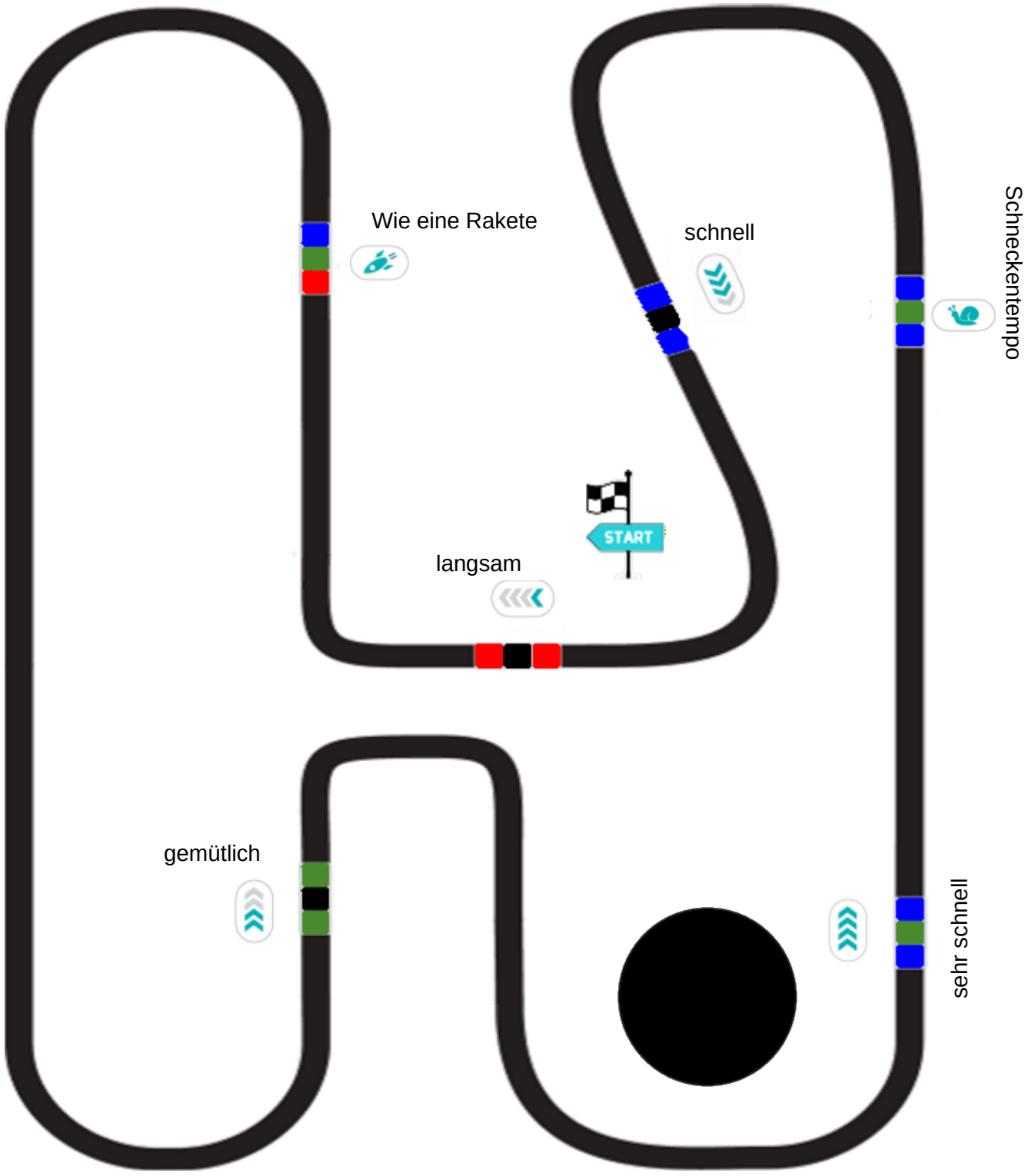






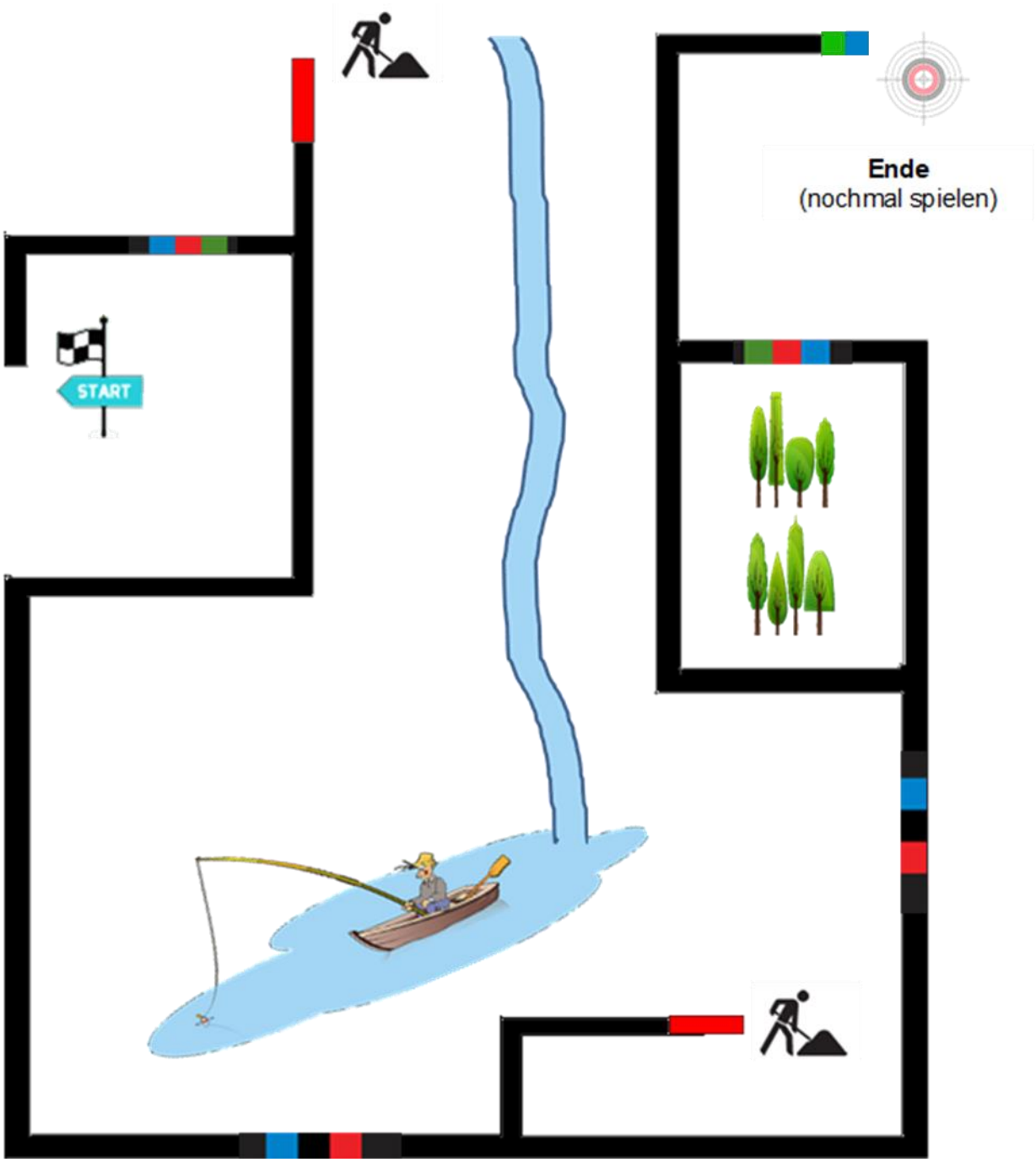
.

Ozobot AB8 Lösung



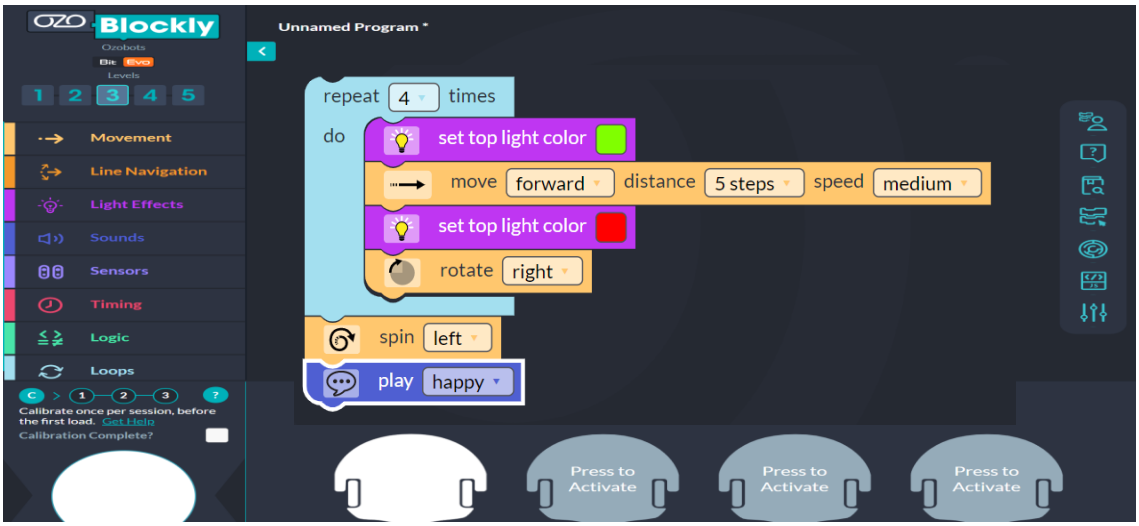
geradeaus weiter

nach rechts fahren



Ozobot programmieren – OzoBlockly (evo und bit)

Der Ozobot lässt sich ab Version 2.0 außerdem mit **OzoBlockly** (**https://ozoblockly.com**), einer **visuellen Programmiersprache,** auch ganz ohne Linien und Farben verwenden und so vollkommen frei steuern. OzoBlockly kennt fünf Schwierigkeitsstufen von **Pre-Reader** (Programmierung mit ganz einfachen Symbolen, Eingabe von Texten) bis hin zu **MASTER** (Schleifen, komplexe Funktionen, Variablen) und die genaue Steuerung der einzelnen Motoren des Ozobots.



Eine mögliche Lösung!

Ozobot AB11a Lösung

Dein Ozobot fährt auf der Eisbahn und zeigt verschiedene Bewegungen und Lichteffekte.

Ausführen der Programme auf dem Ozobot

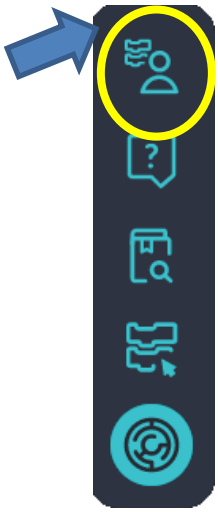
Es gibt **zwei Möglichkeiten**, ein Programm auf deinem Ozobot auszuführen: **Flashing** und **LIVE-Modus.**

1. Mit FLASHING den Bereich für die Kalibrierung und Programmübermittlung einblenden
2. Zum Laden des Programms wird der Ozobot auf die **weiße Fläche** gesetzt und die Schaltfläche Load Evo angeklickt. Das Programm wird dabei durch Lichtsignale übertragen (**flashen**).
3. Während des Ladevorgangs sollte die obere LED grün blinken. Leuchtet die LED rot, musst du den Ozobot neu kalibrieren und den Vorgang wiederholen!
4. Das Programm wird anschließend durch **2-maliges Drücken des Einschaltknopfes** gestartet!

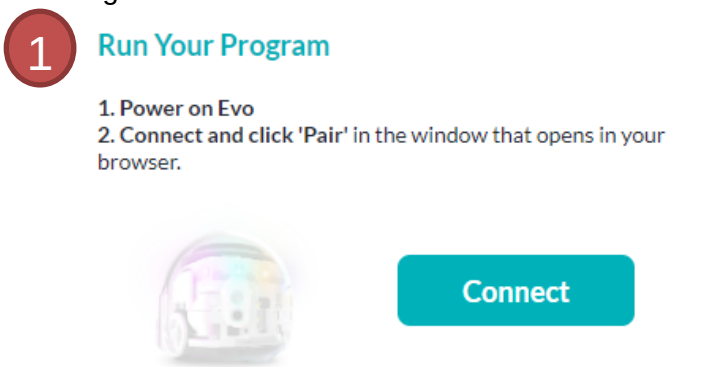
Vor dem ersten Flash-Vorgang musst du deinen **OzoBot kalibrieren**. Falls der weiße Kalibrierungspunkt nicht erscheint, klicke mit der Maus das C an!

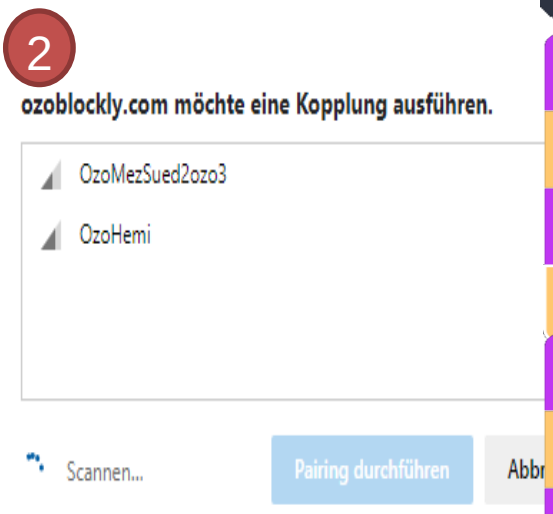
LIVE-Modus (dein Ozobot wird über Bluetooth mit dem Computer verbunden!)

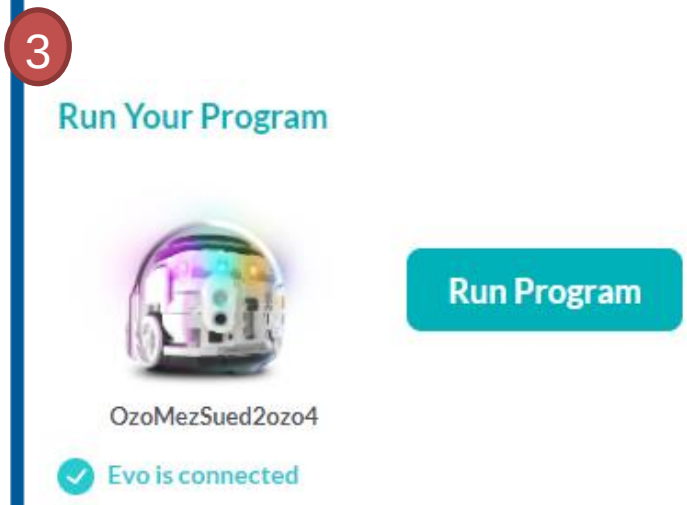
- ✓ wähle aus der Werkzeugleiste auf der rechten Seite „**Programs**“ aus!



- ✓ Schalte deinen Ozobot ein und klicke auf **Connect**
- ✓ Wähle den gewünschten OzoBot aus und führe das **Pairing (**Koppelung**)** durch.
- ✓ Über die Schaltfläche **Run Program** wird das Programm sofort ausgeführt. (Es wird aber nicht so ie beim „Flashen“ im Speicher des Ozobot abgelegt →Sobald die Verbindung getrennt wird, kann das Programm daher nicht mehr gestartet werden!







Ozobot AB11b Lösung

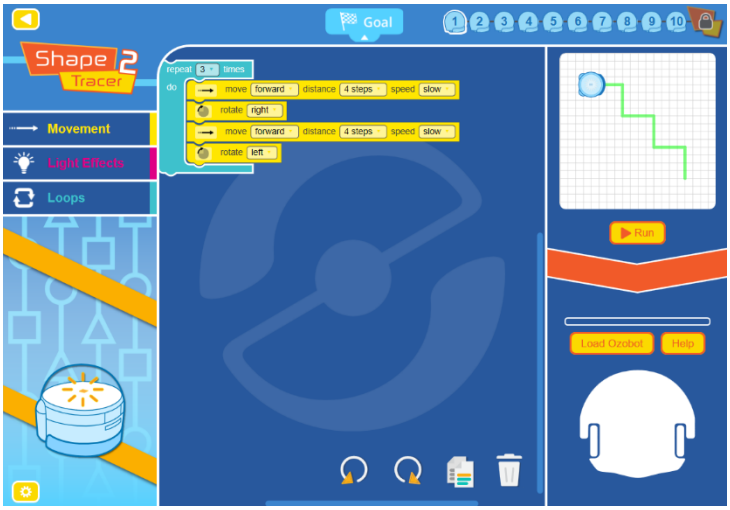
Dein Ozobot soll einen Spaziergang machen. Die zurückgelegte Wegstrecke soll dabei die Form eines Quadrats mit einer Seitenlänge von 5 cm betragen. Bei jeder Vorwärtsbewegung soll die obere LED grün leuchten, bei jeder Drehung rot. Am Ende des Spaziergangs soll dein Ozobot sich einmal linksherum im Kreis drehen und dann eine fröhliche Melodie abspielen! (1 Schritt = 1 step = 1 cm;)

Führe das Programm sowohl im **Live-Modus** als auch im **Autonomen-Modus** (Flashing) aus!

H. Milchram Oktober 2021

Virtual Ozobot (OzoBlockly Challenges)

Auf der Website https://ozobot.com/create/challenges
können mit dem Programmen „**ShapeTracer**“, Ozobot Simmulator und OzoTown, die grundlegenden Schritte der Programmiersprache **Blockly** selbständig erarbeitet werden.

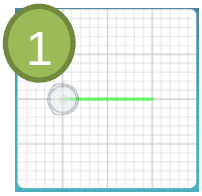


Für die Ausführung der Aufgaben in

Shape Tracer | Shape Tracer 2 | ozobot SIMULATOR | OZO TOWN

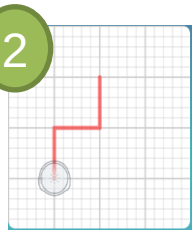
ist ein Ozobot nicht zwingend erforderlich!

Lösungen ShapeTracer 1



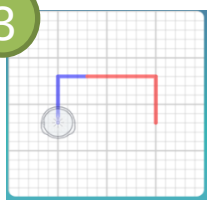
1

set light color
move forward distance 10 steps speed slow



2

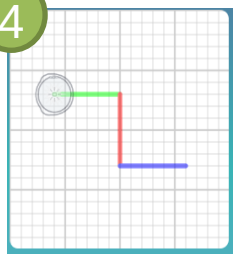
set light color
move forward distance 5 steps speed slow
rotate right
move forward distance 5 steps speed slow
rotate left
move forward distance 5 steps speed slow



3

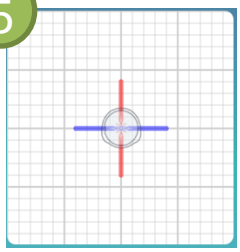
set light color
move forward distance 5 steps speed slow
rotate right
move forward distance 3 steps speed slow
set light color
move forward distance 7 steps speed slow
rotate right
move forward distance 5 steps speed slow

4



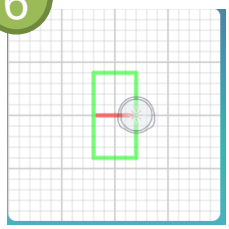
```
set light color
move forward distance 6 steps speed slow
set light color
rotate right
move forward distance 6 steps speed slow
set light color
rotate left
move forward distance 6 steps speed slow
```

5



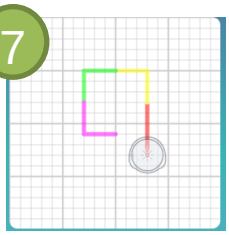
```
set light color
move forward distance 4 steps speed slow
move backward distance 8 steps speed slow
move forward distance 4 steps speed slow
set light color
rotate right
move forward distance 4 steps speed slow
move backward distance 8 steps speed slow
```

6



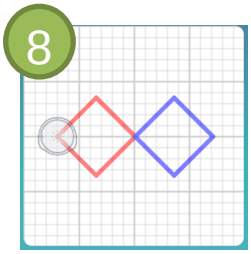
```
set light color
move forward distance 4 steps speed slow
rotate left
move forward distance 4 steps speed slow
rotate left
move forward distance 8 steps speed slow
rotate left
move forward distance 4 steps speed slow
rotate left
move forward distance 4 steps speed slow
set light color
rotate left
move forward distance 4 steps speed slow
```

7



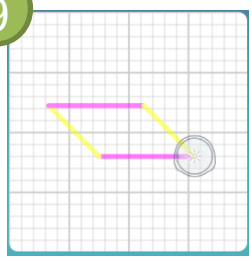
```
set light color
move forward distance 5 steps speed slow
set light color
move forward distance 3 steps speed slow
rotate left
move forward distance 3 steps speed slow
set light color
move forward distance 3 steps speed slow
rotate left
move forward distance 3 steps speed slow
set light color
move forward distance 3 steps speed slow
rotate left
move forward distance 3 steps speed slow
```

8



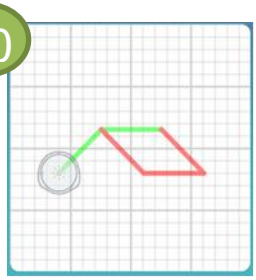
```
set light color
rotate slight right
move forward distance 5 steps speed slow
rotate right
move forward distance 5 steps speed slow
set light color
move forward distance 5 steps speed slow
rotate left
move forward distance 5 steps speed slow
rotate left
move forward distance 5 steps speed slow
rotate left
move forward distance 5 steps speed slow
set light color
move forward distance 5 steps speed slow
rotate right
move forward distance 5 steps speed slow
```

9



```
set light color
rotate left
move forward distance 8 steps speed slow
rotate slight right
set light color
move forward distance 6 steps speed slow
rotate slight right
set light color
rotate right
move forward distance 8 steps speed slow
rotate slight right
set light color
move forward distance 6 steps speed slow
```

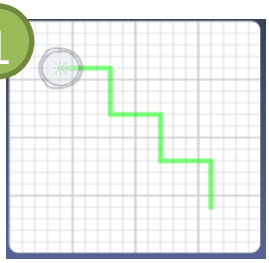
10



```
set light color
rotate slight right
move forward distance 5 steps speed slow
rotate slight right
move forward distance 5 steps speed slow
rotate slight right
set light color
move forward distance 5 steps speed slow
rotate slight right
rotate right
move forward distance 5 steps speed slow
rotate slight right
move forward distance 5 steps speed slow
```

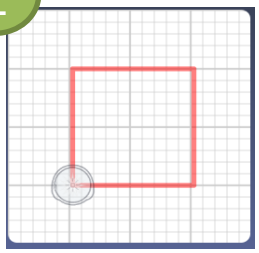
Lösungen ShapeTracer 2

1



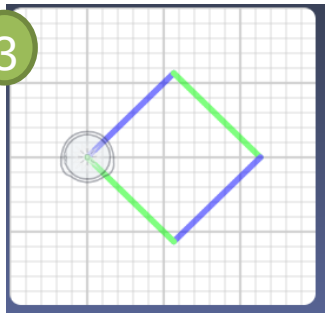
```
repeat 3 times
do
    set light color
    move forward distance 4 steps speed slow
    rotate right
    move forward distance 4 steps speed slow
    rotate left
```

2



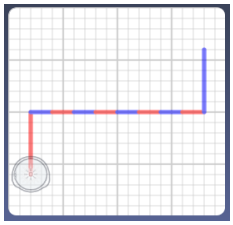
```
set light color
repeat 4 times
do
    move forward distance 10 steps speed slow
    rotate right
```

3



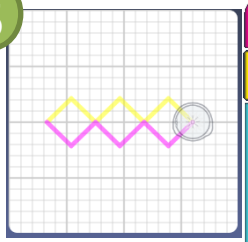
```
rotate slight right
repeat 2 times
do
    set light color
    move forward distance 8 steps speed slow
    set light color
    rotate right
    move forward distance 8 steps speed slow
    rotate right
```

4



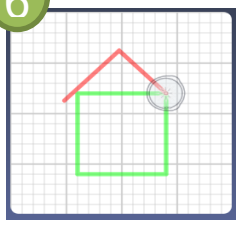
```
set light color
move forward distance 6 steps speed very fast
rotate right
repeat 4 times
do
    set light color
    move forward distance 2 steps speed slow
    set light color
    move forward distance 2 steps speed slow
set light color
rotate left
move forward distance 6 steps speed very fast
```

5



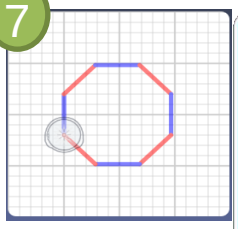
```
set light color
rotate slight left
repeat 3 times
do
    move forward distance 3 steps speed slow
    rotate left
    move forward distance 3 steps speed slow
    rotate right
set light color
rotate u-turn left
repeat 3 times
do
    move forward distance 3 steps speed slow
    rotate left
    move forward distance 3 steps speed slow
    rotate right
```

6



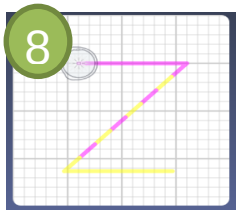
```
rotate left
repeat 4 times
do
    set light color
    move forward distance 8 steps speed slow
    rotate right
set light color
rotate u-turn left
rotate slight left
move forward distance 6 steps speed slow
rotate left
move forward distance 7 steps speed slow
```

7



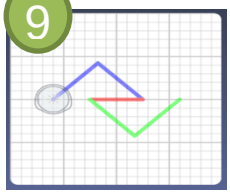
```
repeat 4 times
do
    set light color
    move forward distance 4 steps speed slow
    rotate slight right
    set light color
    move forward distance 4 steps speed slow
    rotate slight right
```

8



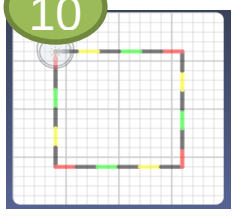
```
set light color
move forward distance 10 steps speed slow
rotate u-turn left
rotate slight left
repeat 4 times
do
    move forward distance 2 steps speed slow
    set light color
    move forward distance 2 steps speed slow
    set light color
set light color
rotate left
rotate slight left
move forward distance 10 steps speed slow
```

9



```
set light color
rotate slight right
repeat 2 times
do
    move forward distance 6 steps speed slow
    rotate right
set light color
rotate slight right
move forward distance 5 steps speed slow
set light color
rotate slight right
repeat 2 times
do
    move backward distance 6 steps speed slow
    rotate left
```

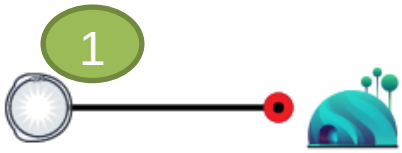
10



```
repeat 4 times
do
    set light color
    move forward distance 2 steps speed slow
    turn top light off
    move forward distance 2 steps speed slow
    set light color
    move forward distance 2 steps speed slow
    turn top light off
    move forward distance 2 steps speed slow
    set light color
    move forward distance 2 steps speed slow
    turn top light off
    move forward distance 2 steps speed slow
    rotate left
```

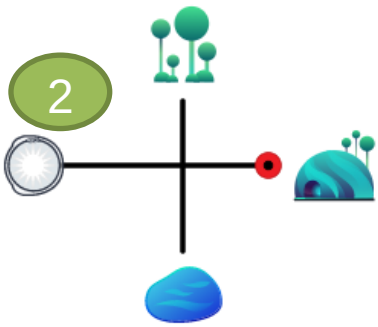
Lösungen OZO Town

1



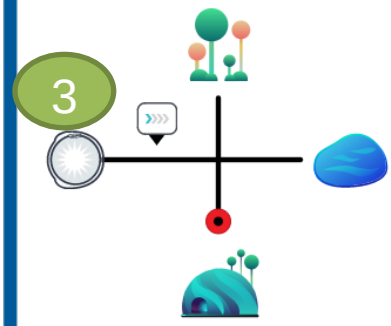
- follow line to next intersection or line end

2



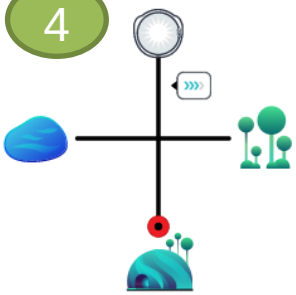
- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end

3



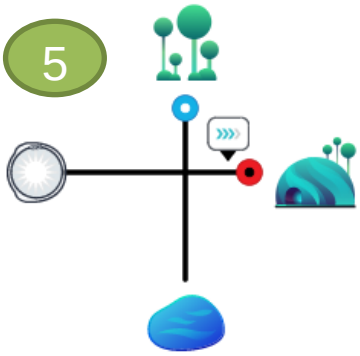
- set line-following speed to slow
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end

4



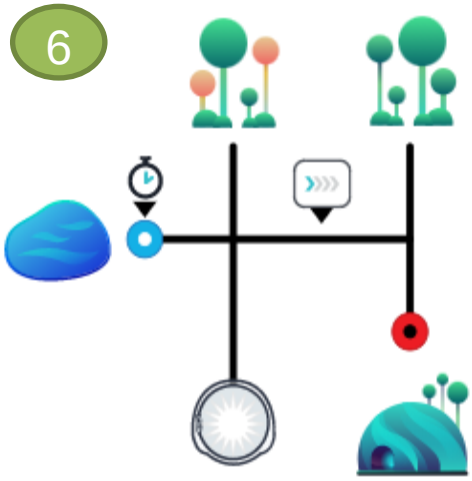
- set line-following speed to fast
- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end

5



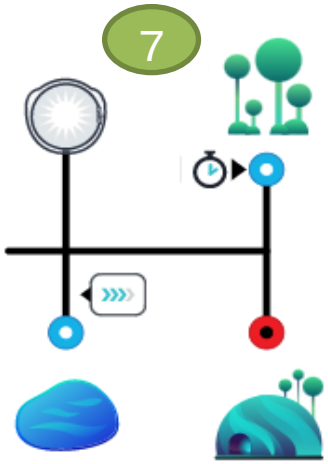
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- pick direction: back
- follow line to next intersection or line end
- pick direction: left
- set line-following speed to fast
- follow line to next intersection or line end

6



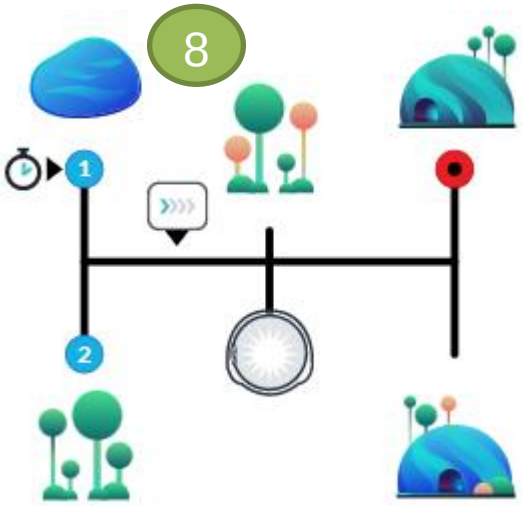
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: straight
- set line-following speed to slow
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end

7



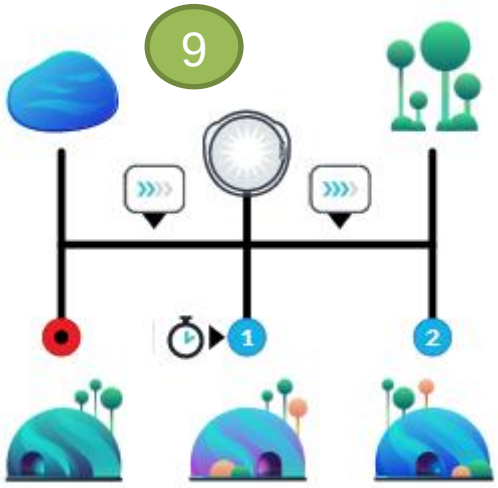
- follow line to next intersection or line end
- pick direction: straight
- set line-following speed to fast
- follow line to next intersection or line end
- pick direction: back
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end

8



- follow line to next intersection or line end
- pick direction: left
- set line-following speed to slow
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end
- pick direction: back
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end

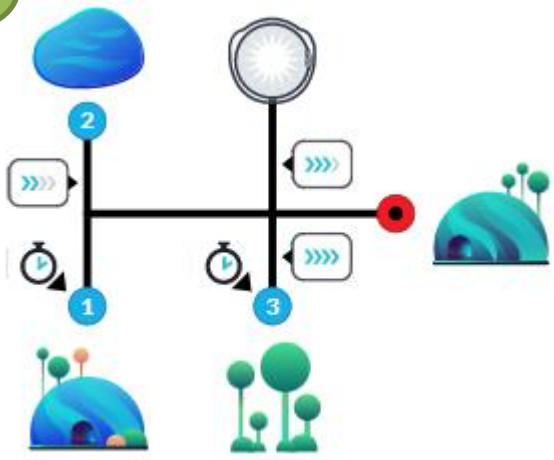
9



- follow line to next intersection or line end
- pick direction: straight
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: right
- set line-following speed to fast
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end
- pick direction: back
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- pick direction: straight
- set line-following speed to medium
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end

- set line-following speed to fast
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: straight
- set line-following speed to medium
- follow line to next intersection or line end
- pick direction: back
- follow line to next intersection or line end
- pick direction: left
- follow line to next intersection or line end
- pick direction: right
- set line-following speed to very fast
- follow line to next intersection or line end
- wait 1 second(s)
- pick direction: back
- follow line to next intersection or line end
- pick direction: right
- follow line to next intersection or line end

10



Nach der erfolgreichen Absolvierung aller 10 Level, kann ein Spiel heruntergeladen werden!

Ozobot bit APP

mit **Ozobot evo** und **Ozobot bit** verwendbar

Die Ozobot bit APP ist sowohl für Android als auch für Apple-Geräte im Store gratis zum Downloaden.
Die APP enthält folgende Anwendungen:

- ✓ **OzoDraw:** eine Zeichen- und Erkundungs-App, um Ozobots Intelligenz zu testen
- ✓ **OzoLuck:** soziales Glücksspiel, bei dem Ozobot in einem Labyrinth reist
- ✓ **OzoGrove:** Genieße Musik auf eine ganz neue Art und Weise beim Spielen und Lernen mit deinem Ozobot bit (funktioniert nicht mit Ozobot evo!)
- ✓ **OzoPath:** strategisches Spiel, du versuchst deinen Gegner auszutricksen, um Ozobot ins Ziel zu bringen.

.

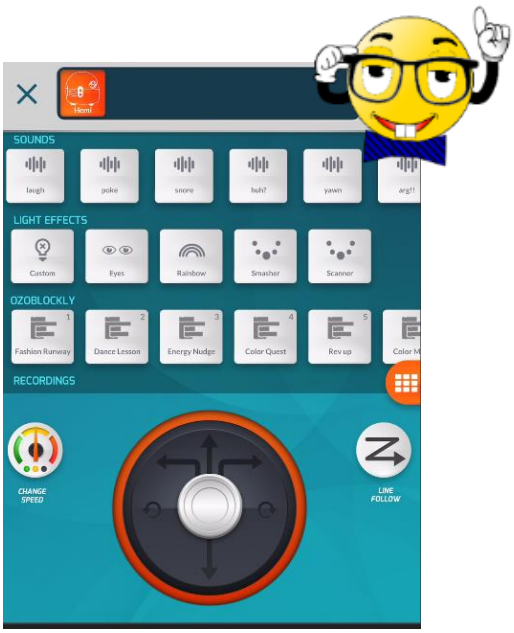
Ozobot evo APP

Für den **Ozobot evo** gibt es zusätzlich eine APP für **Android** und **iOS**, die man kostenlos über die jeweiligen App-Stores downloaden kann. Mit Bluetooth wird anschließend der **Ozobot evo** verbunden. Auf dich warten zahlreiche herausfordernde Aufgaben und Spiele. Über die App lässt sich dein **Ozobot evo** auch steuern und kalibrieren.

Über die Schaltfläche „**Play**“ bekommst du fünf Möglichkeiten, mit dem Ozobot evo zu arbeiten, angeboten!

Drive

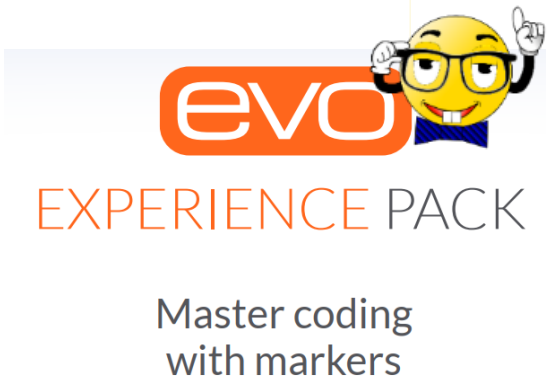
- ✓ Steuere deinen Ozobot wie mit einer **Fernbedienung**
- ✓ Ändere die **Geschwindigkeit**
- ✓ Schalte die **Linienverfolgung** ein
- ✓ schalte die **LEDs** ein und aus
- ✓ ändere die **Farben** der LEDs
- ✓ probiere verschiedene **Lichteffekte** aus
- ✓ probiere verschiedene **Soundeffekte** aus



Experience PACK

https://play.ozobot.com/print/ozobot-evo-experience-pack.pdf

Mit dem Experience Pack lernst du deinen Ozobot evo mit Hilfe von Farbcodes zu steuern.
11 verschiedene Aufgaben führen dich Schritt für Schritt an die Programmierung deines **Ozobot evo** oder **Ozobot bit** heran.

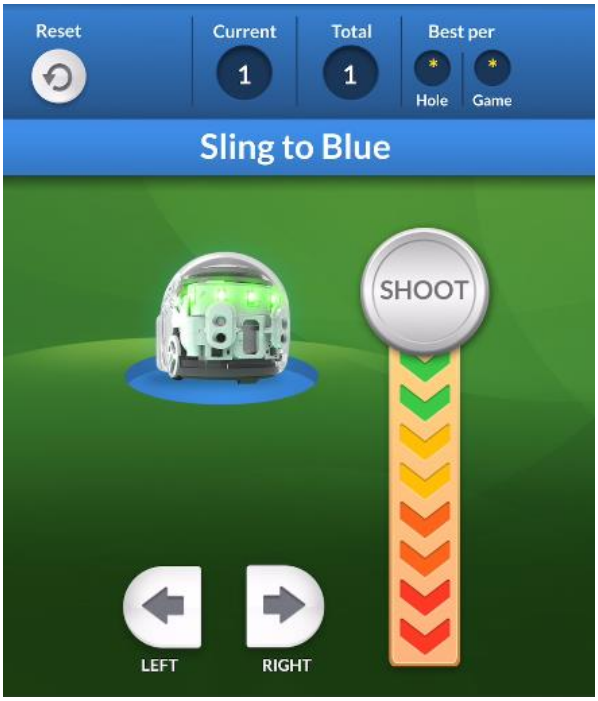


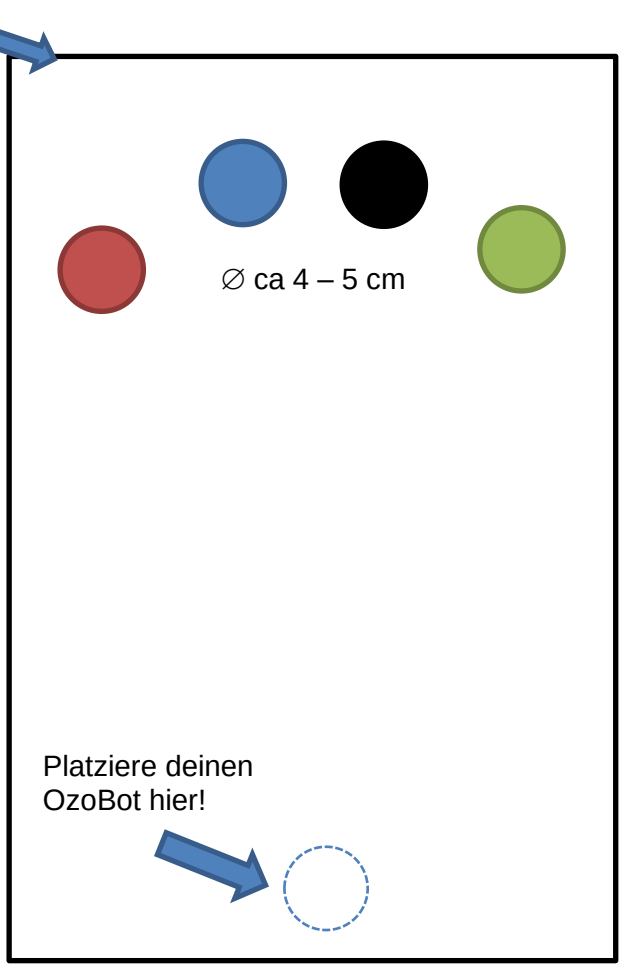
OzoLaunch - Zielschießen mit dem Ozobot evo

Was brauchst du?

- ✓ Ein Blatt weißes Papier (Größe mindestens A3)
- ✓ Marker in den Farben
  ROT, **SCHWARZ**, **BLAU** und **GRÜN**
- ✓ Tablet oder Handy mit der **Ozobot-APP**

1. Gestalte eine **Spielfläche** wie abgebildet
   Die gemalten, farbigen Punkte sollten etwas größer als dein Ozobot sein! ∅ ca 4 – 5 cm
2. Öffne die Ozobot-App
3. Wähle „OzoLaunch“ aus und folge den Anweisungen am Bildschirm

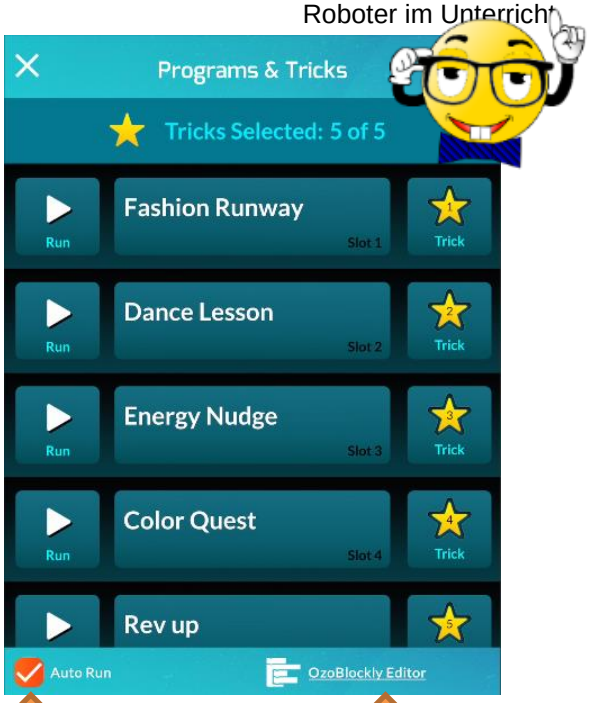




Programs & Tricks

Mit diesen kleinen Beispielprogrammen kannst du einige weitere Fähigkeiten deines Ozobot evo kennenlernen. Bis zu **fünf** dieser Tricks kannst du auf deinem Ozobot evo speichern. Diese Programme kannst du dann jederzeit direkt auf deinem Roboter starten, ohne eine APP oder den Blockly-Editor zu nutzen.
Um einen Trick auszuführen musst du folgende Schritte durchführen:

- ✓ alle **vier Infrarot-Sensoren** abdecken
- ✓ warte bis ein Signal erklingt
- ✓ Durch Drücken des Einschaltknopfes werden die Tricks ausgewählt:
  1x drücken →Trick 1
  2x drücken →Trick 2
  3x drücken →Trick 3
  …
  Die Nummer des ausgewählten Tricks wird jedesmal angesagt!



Ist das Häkchen bei „AutoRun“ gesetzt, werden bis zu fünf ausgewählten Tricks automatisch an deinen Ozobot evo gesendet und stehen dort zum Abrufen bereit!

Mit dem OzoBlockly Editor kannst du deine eigenen Tricks programmieren und in der APP hinzufügen!

OzoBlockly EDITOR

Zum Erstellen deiner eigenen Programme kannst du **OzoBlockly** verwenden.

Programm in der Ozobot evo APP speichern

Programm sofort ausführen!

H. Milchram Oktober 2021

2.InO-Bot

Alter: **8+** (Bezugsquellen: http://www.tts-group.co.uk ca. 115€)

InO-Bot: *H. Milchram 2017*

Der "**Input-Output-Roboter**" (kurz **InO-Bot**) wurde speziell für die Verwendung mit der bekannten Programmiersprache „**Scratch**“ entwickelt.

Der **InO-Bot** kann über Bluetooth mit dem Computer verbunden werden. Die Software kann direkt mit dem InO-Bot kommunizieren, wodurch die Kinder die programmierten Befehle in Echtzeit beobachten können. Die transparente Verkleidung ermöglicht einen Einblick in die technischen Komponenten des InO-Bots. Der InO-Bot wird mit einem wieder aufladbaren Akku betrieben. Das **Aufladen** erfolgt mit einem Standard **USB-A** →**USB-B** Kabel.

Technische Daten:

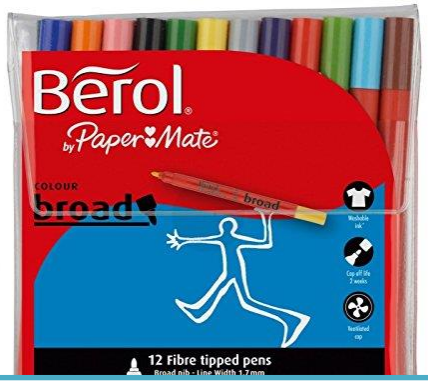
- ✓ Wieder **aufladbare Akkus**
- ✓ **Bluetooth**
- ✓ Transparentes Gehäuse

Sensoren:

- ✓ Ultraschall **Distanzsensor** (Frontseite)
- ✓ 4 optische Näherungssensoren (in den Ecken)
- ✓ **2 optische Sensoren** zur Linienverfolgung
- ✓ **Mikrofon**
- ✓ **Helligkeitssensor**

Aktoren:

- ✓ **8 RGB LEDs**
- ✓ **2 weiße LEDs** Frontlichter
- ✓ **Lautsprecher**
- ✓ **2 Motoren** mit Wegmessung
- ✓ Elektromagnetischer **Stifthalter** mit Vorrichtung zum Heben und Senken des Stiftes



Stifte zum Einsetzen in den InO-Bot
Die Stifte müssen einen **Durchmesser** von **10mm** besitzen

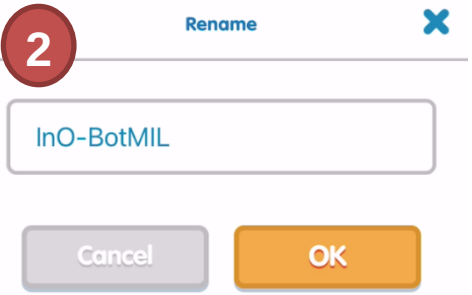
Programmierung

- ✓ mit **Scratch** auf Geräten mit **Windows** - Betriebssystem oder über APP für **Andoid** und **iOS** (Tablet, Handy) programmierbar

Arbeiten mit mehreren InO-Bots im Klassenverband

Werden mehrere **InO-Bots** im Klassenverband verwendet, muss jeder Roboter mit einem eigenen Namen versehen werden. Die Umbenennung des InO-Bots erfolgt über die **InO-Bot APP**.

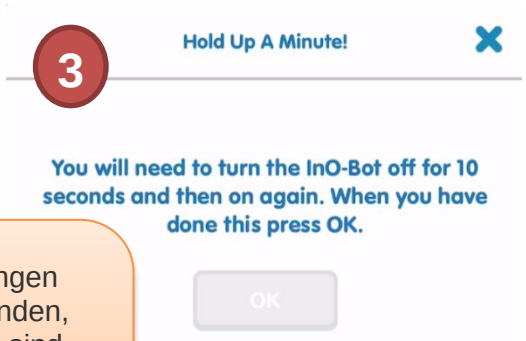
1. **InO-Bot** über das **Bluetooh-Symbol** verbinden 



2. Stift-Symbol antippen 

und den gewünschten Namen eintragen!

3. Den Anweisungen auf dem Bildschirm folgen!



Bitte unbedingt genau die Reihenfolge der Anweisungen befolgen! Die APP unbedingt im **Querformat** verwenden, da sonst einige Elemente der Oberfläche unsichtbar sind.

Achtung: Bei allen Geräten (Notebooks, Handys, Tablets), die nicht für die Programmierung mit einem Ino-Bot verwendet werden, die Bluetooth-Verbindung deaktivieren!

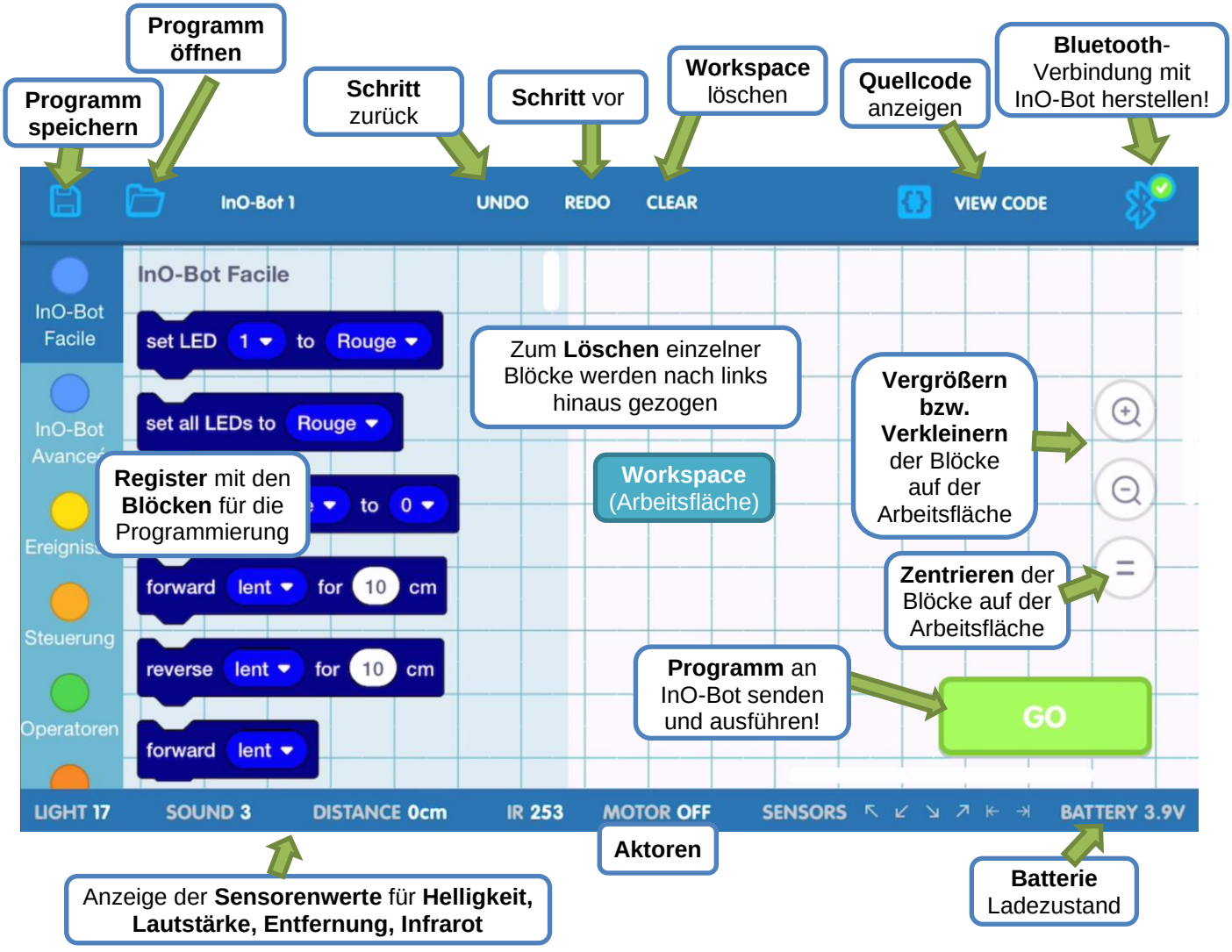
Programmierung unter Verwendung der InO-Bot APP

Die InO-Bot APP kann aus den jeweiligen Stores heruntergeladen werden:

Achtung: *Die Sprache der Beschriftung der einzelnen Befehle ist bunt durcheinander gemischt auf Deutsch, Englisch und Französisch*

- ✓ Handys und Tablets mit **Android** Betriebssystem: Google play-Store
- ✓ iPhone und iPad (ab **iOS10**): Available on the App Store
- ✓ Um die InO-Bot APP unter Windows ausführen zu können, ist zusätzlich ein ANDROID-Emulator wie **BlueStacks** oder **NOX** erforderlich.

APP Programmieroberfläche



Installation und Ausführen der InO-Bot Blockly APP unter Windows

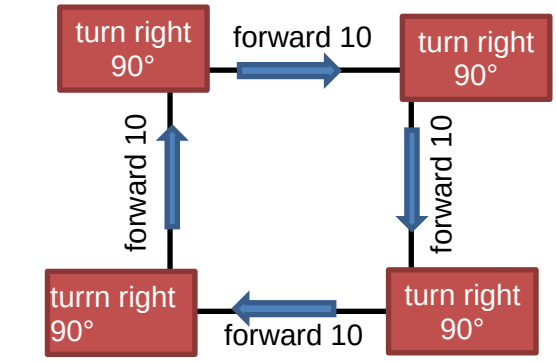
- ✓ Download von einem **Android Software Emulator** für **Windows**
  Dieses Emulationsprogramm ermöglicht das Ausführen von Android-APPs auf einem Windows-PC oder Notebook
  BlueStacks https://bit.ly/2CqBLGW und
  NOX https://noxofficial.com/nox-for-pc/
  sind Android-Emulatoren für Windows PCs zum Ausführen der **InO-Bot APP**

- ✓ Software Emulator am Windows PC installieren
- ✓ Installierten Emulator ausführen und über den Google-Play Store die **InO-Bot APP** suchen und installieren!

H. Milchram Oktober 2021

Erste Programme mit der InO-Bot APP

InO-Bot AB1Lösung



Bringe deinen Ino-Bot dazu, ein **Quadrat mit einer Seitenlänge von 10 cm** zu zeichnen!

Trage hier die Befehle in der richtigen Reihenfolge ein:

1. Wenn angeklickt wird
2. Pen down
3. Forward 10
4. Turn right 90°
5. Forward 10
6. Turn right 90°
7. Forward 10
8. Turn right 90°
9. Forward 10
10. Turn right 90°
11. Pen up

Was fällt dir auf, wenn du den Programmablauf betrachtest?

Der **InO-Bot** besitzt eine integrierte **Stifthalterung**. Ein eingesetzter Stift kann über folgenden Block gesteuert werden:

Stift **heben** (up) und **senken** (down)

TIPP: Denke daran, dass du den **Stift**, vor der ersten Bewegung des Roboters **senken** (pen down) und am Schluss wieder **heben** (pen up) musst!

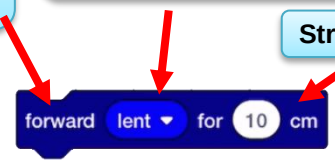
Folgende Befehle aus dem Register „**InO-Bot Facile**“ werden auch noch benötigt:

Moves (Bewegungen) & **Turns** (Drehungen):

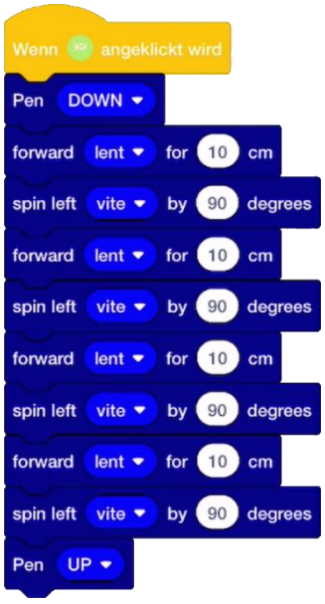
Richtung der Bewegung **forward** (vorwärts), **reverse** (rückwärts)

Geschwindigkeit: **lent** (langsam), **moyen** (mittel), **vite** (schnell)

Strecke in **cm**



Zusätzlich ist ein **Ereignis** erforderlich, durch welches das Programm gestartet wird. Wir verwenden dazu das Ereignis



Richtung der Drehung **left** (links), **right** (rechts)

Winkel in Grad

Geschwindigkeit: **lent** (langsam), **moyen** (mittel), **vite** (schnell)

Was fällt dir auf, wenn du den Programmablauf betrachtest?

Die Befehle „Forward“ und „Turn right“ werden jeweils 4x wiederholt!

Kontrollstrukturen: Schleife (ITERATION):

Eine Schleife (**Iteration**) ist bei der Programmierung eine Kontrollstruktur, bei der eine Anweisung oder ein Anweisungsblock solange wiederholt wird, bis eine Abbruchbedingung erfüllt wird. **Die Schleife** erspart das mehrfache Notieren von Befehlen.

Man unterscheidet:

- **Kopfgesteuerte Schleifen**
  Die Überprüfung der Abbruchbedingung erfolgt vor dem Durchlauf der Schleife, das kann dazu führen, dass die Anweisungen innerhalb der Schleife überhaupt nicht ausgeführt werden!

- **Fußgesteuerte Schleifen**
  Die Abfrage der Abbruchbedingung erfolgt am Ende der Schleife, diese Schleife wird dadurch zumindest einmal durchlaufen!

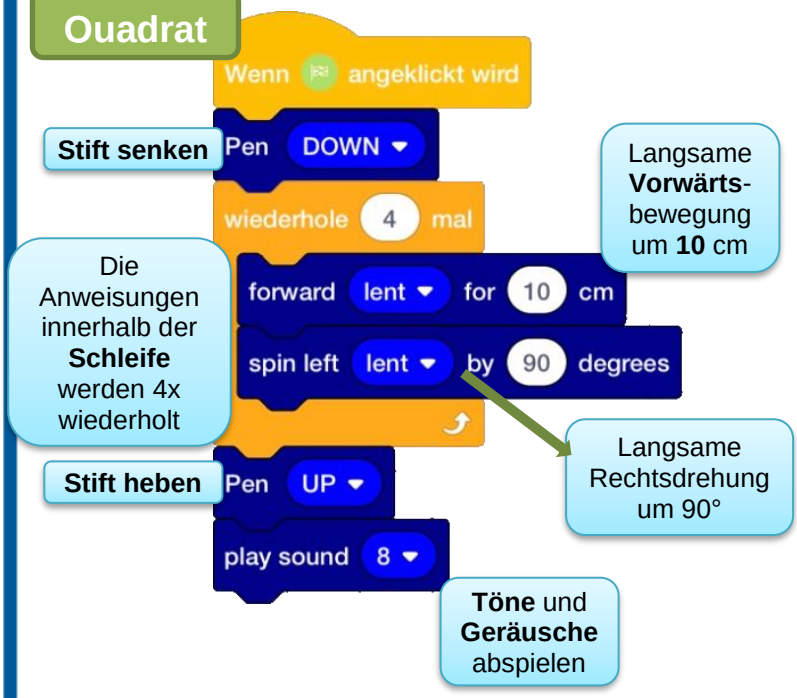
Die Blöcke zum Programmieren einer Schleife findest du im Register „**Steuerung**“ 

Anzahl der

Alle **Blöcke,** die sich innerhalb der Kontrollstruktur befinden, werden so oft wie durch „**Anzahl**“ angegeben **wiederholt.**

InO-Bot AB2a Lösung

Ouadrat



Stift senken

Langsame **Vorwärts**-bewegung um **10** cm

Die Anweisungen innerhalb der **Schleife** werden 4x wiederholt

Stift heben

Langsame Rechtsdrehung um 90°

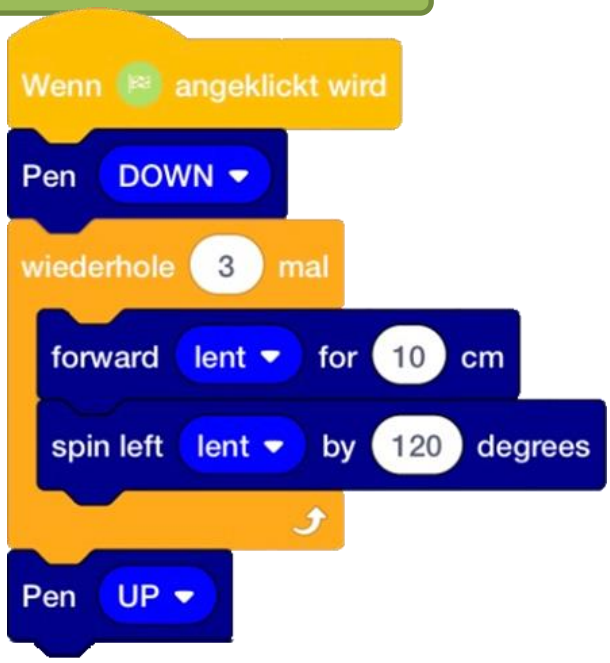
Töne und **Geräusche** abspielen

Quelltext

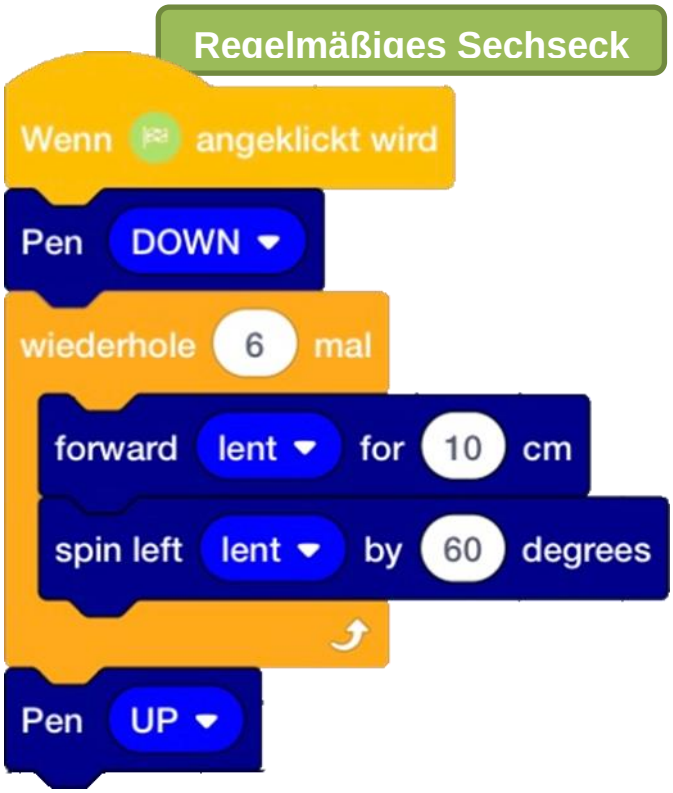
```
//When play button pressed
pen("DOWN");
var xye = 4;
for(var upp = 0; upp < xye; upp++)
{
  forwardForDistance("02", "10");
  spinLeftByDegrees("02", "90");
}
pen("UP");
playSound("8");
```

InO-Bot AB2b Lösung

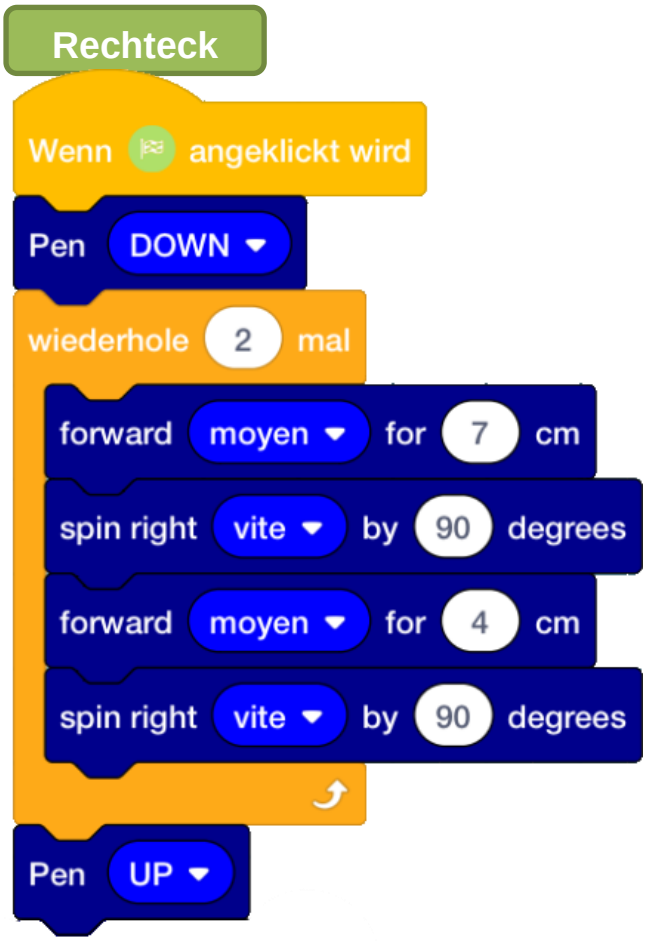
Gleichseitiges Dreieck



Regelmäßiges Sechseck



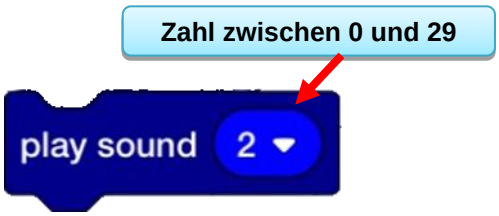
Rechteck



Ausgabe von Tönen und Geräuschen (Sounds):

Der Befehl „play“ für die Ausgabe von Tönen und Geräuschen befindet sich im Register „**InO-Bot Facile**“ 

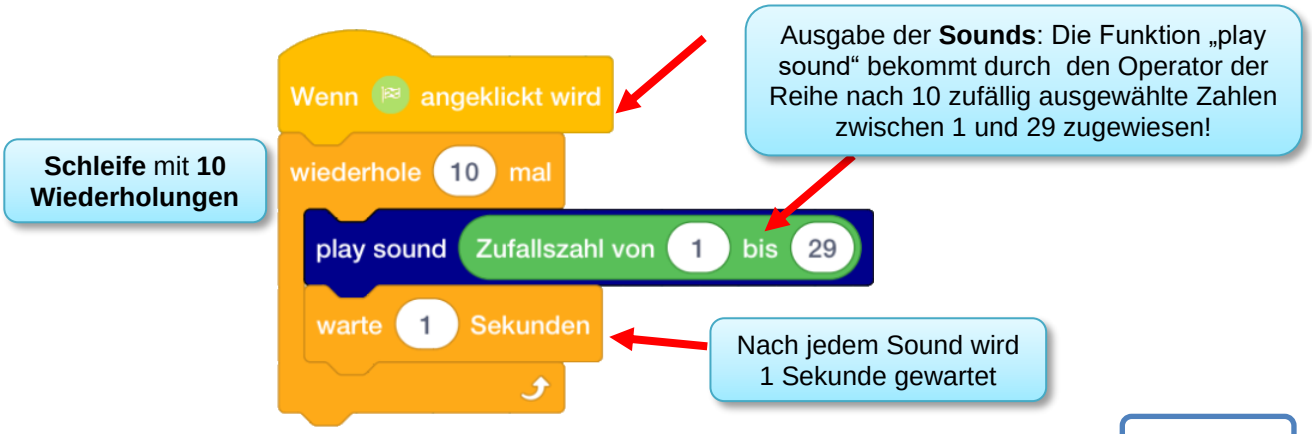
Es gibt insgesamt **29** verschiedene voreingestellte Möglichkeiten, die über den folgenden Block zugewiesen werden!



29 verschiedene Sounds

- ✓ 1 – 9: verschiedene **Klangeffekte**
- ✓ 10 – 16: **Klavier (**Tonleiter**)**
- ✓ 17 – 29: **Xylophon (**Tonleiter**)**

InO-Bot AB3 Lösung



Quellcode

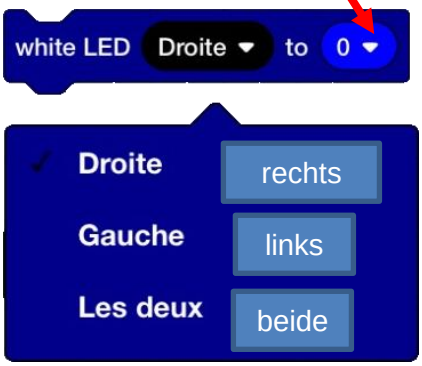
```
var ton, test;

//When play button pressed
var yid = 10;
for(var itz = 0; itz < yid; itz++)
{
  playSound("randomNumber(1, 29)");
}
waitForSecond(1);
```

Steuerung der Front-LEDs

Der InO-Bot besitzt **2 Front-LEDs** (weiß), die unabhängig voneinander gesteuert werden können!

- ✓ **Helligkeitswerte** für die **Front-LEDs** 0-10

Helligkeitswerte 0-10

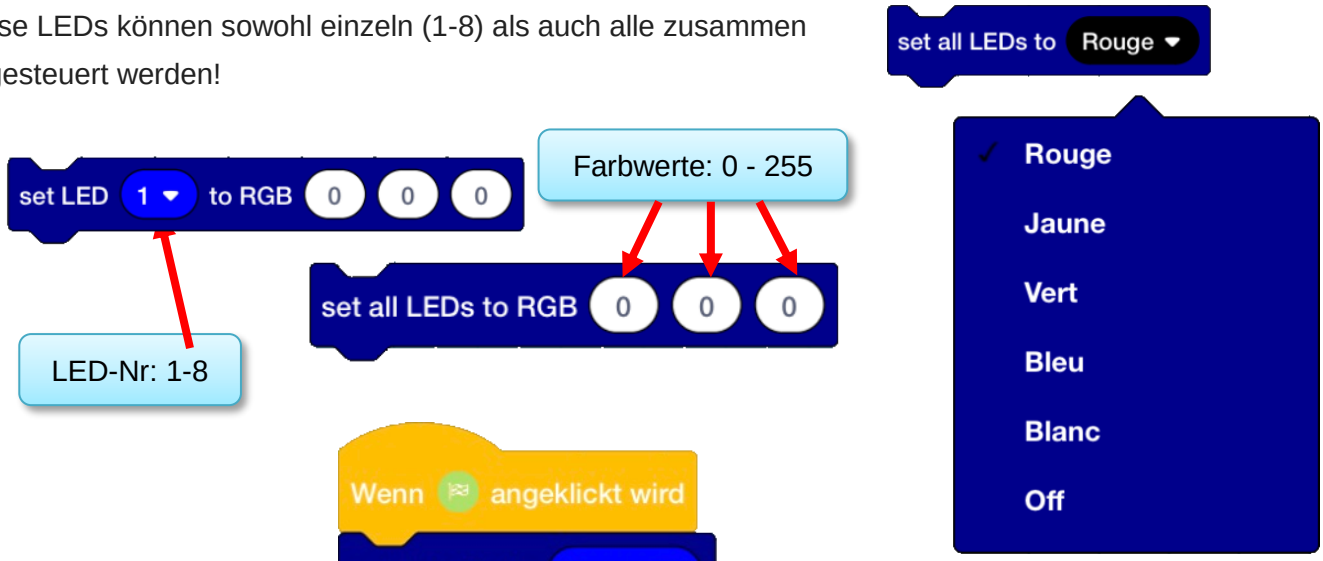


Endlos-Schleife

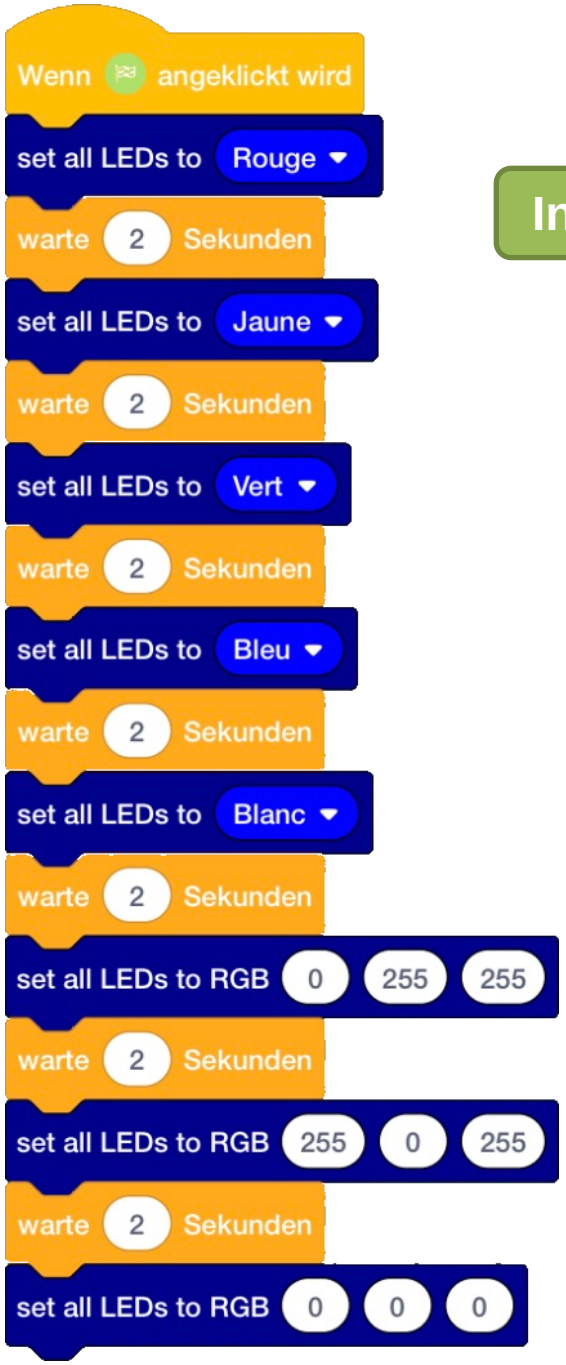
RGB LEDs – Little light show

Zusätzlich zu den beiden weißen Front-LEDs besitzt der InO-Bot insgesamt 8 einzeln steuerbare RGB-LEDs. Die Farben der LEDs können sowohl Farbbezeichnungen (nur einige Grundfarben →Rot, Gelb, Grün, Blau, Weiß) und über **Farbwerte** 0 – 255 gesteuert werden.

Diese LEDs können sowohl einzeln (1-8) als auch alle zusammen angesteuert werden!

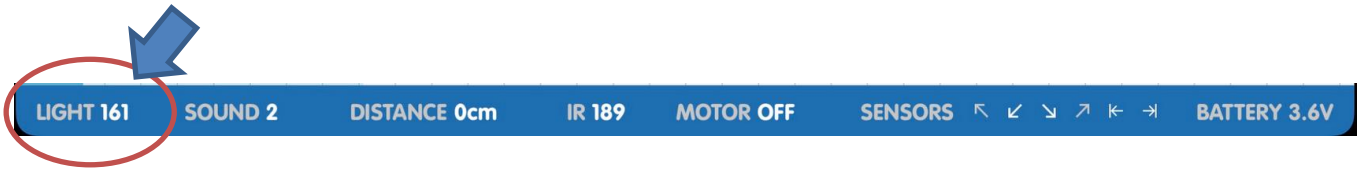


InO-Bot AB4b Lösung

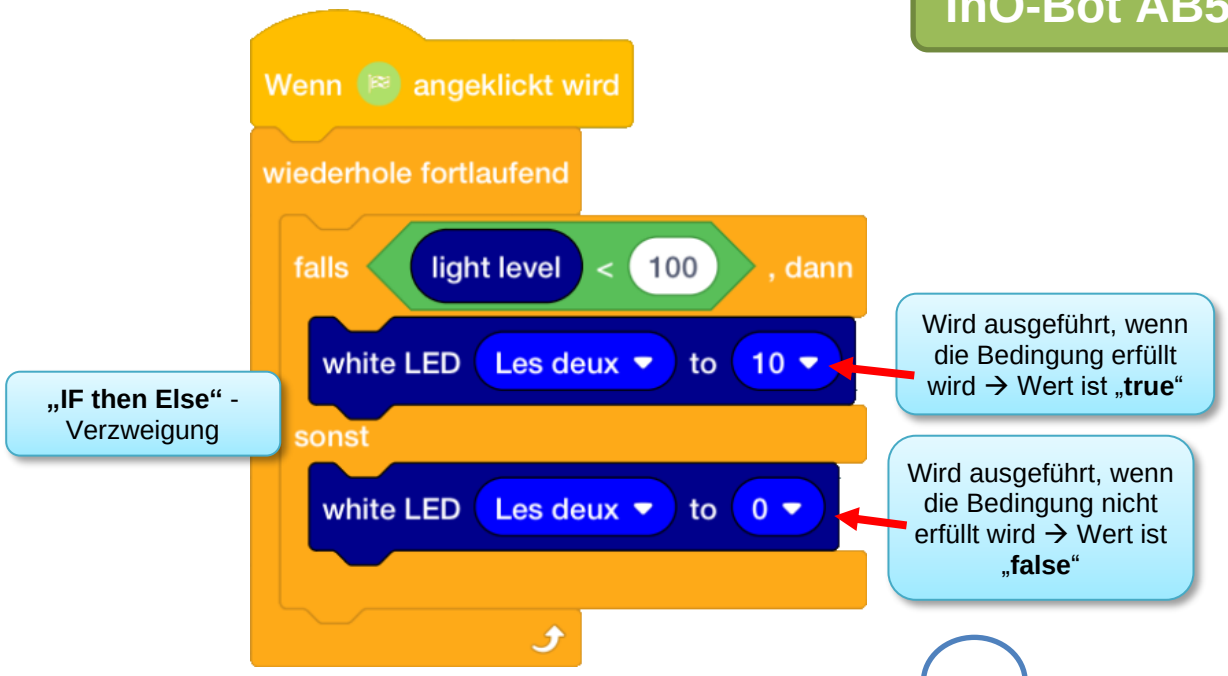


SENSOREN: Lichtsensor

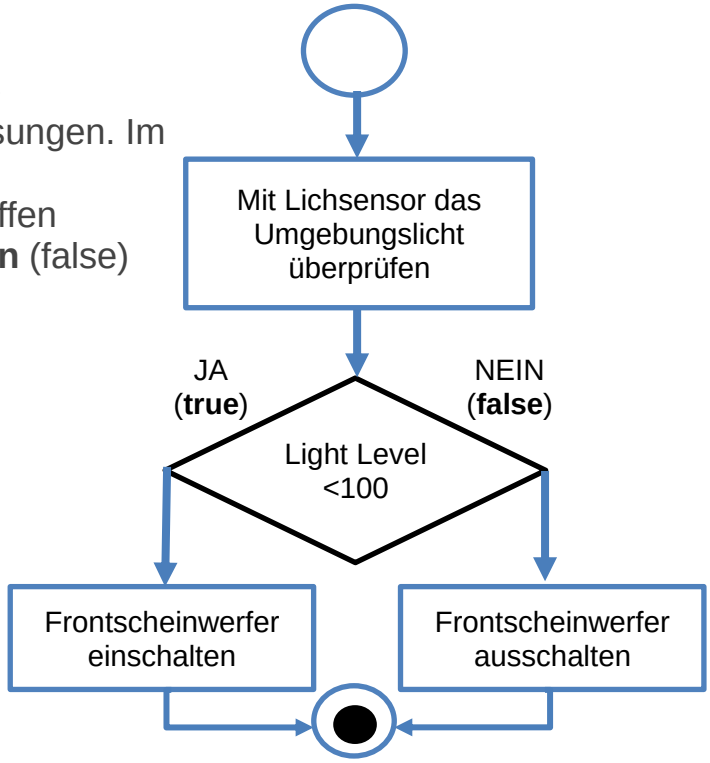
Der InO-Bot besitzt einen **Lichtsensor,** mit dem die LEDs abhängig von der Helligkeit ein- oder ausgeschaltet werden können! In der Statusleiste der Programmieroberfläche kann jederzeit der aktuelle Helligkeitswert abgelesen werden.



InO-Bot AB5 Lösung



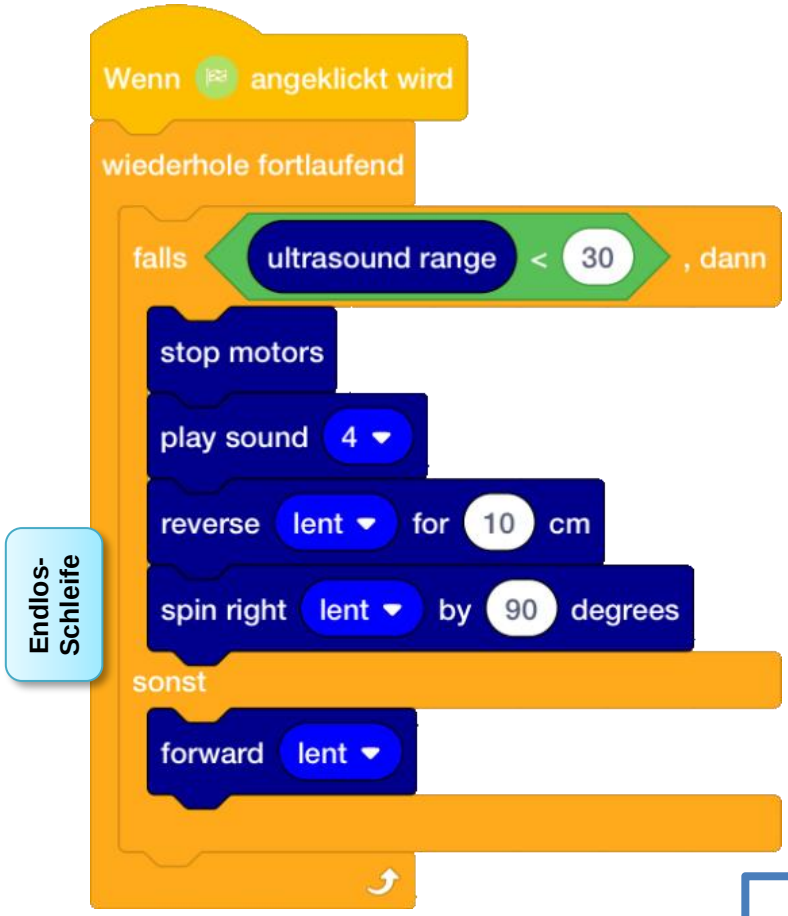
Bislang waren unsere Programme eine Hintereinander-Ausführung von Anweisungen. Im obigen Programm muss aber je nach Lichtintensität eine Entscheidung getroffen werden, die man mit **Ja** (true) oder **Nein** (false) beantwortet werden kann.

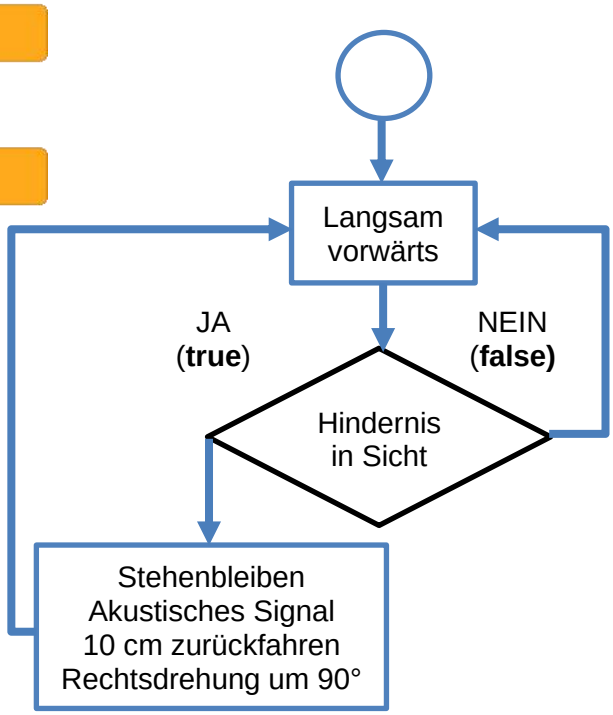


SENSOREN: Distanzsensor

Mit Hilfe seiner beiden Ultraschall **Distanzsensoren** kann dein InO-Bot erkennen, ob sich ein Hindernis vor ihm befindet und damit eine Kollision droht. Die Entfernung wird in cm (0 – 255 cm) angegeben.

InO-Bot AB6 Lösung





In einer **Endlosschleife** ist die zugehörige Bedingung immer erfüllt, es gibt kein Abbruchkriterium. Bekannt sind Endlosschleifen in erster Linie als unerwünschte Programmierfehler. Jedoch gibt es auch Szenarien (siehe Programm), in denen man durchaus ganz bewusst keinen Abbruch möchte.

Weitere interessante Anwendungsbeispiele ergeben sich aus dem Einsatz der zwei Sensoren auf der Unterseite des InO-Bot. Der Roboter kann damit so programmiert werden, dass er einer schwarzen Linie folgt

Quellcode

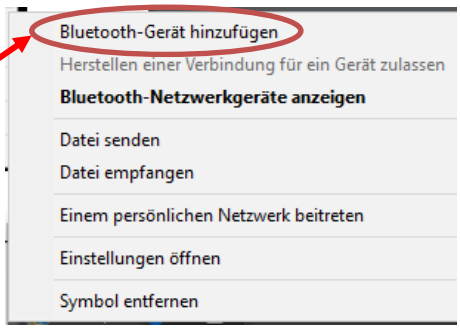
Programmierung mit Scratch

Folgende Dinge werden benötigt:

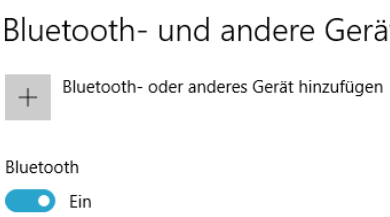
- ✓ PC oder Tablet mit Windows7, 8.1 oder **Windows10**
- ✓ **Bluetoothverbindung**
- ✓ Offline Version von **Scratch** →**https://scratch.mit.edu/download**
  Achtung: Unbedingt **Scratch 2.0** verwenden, Version 3.x und neuer verursacht Probleme!!!
- ✓ **Scratch Launcher** von →https://bit.ly/2P70wuf

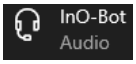
Herstellung der Blutoothverbindung mit Windows10

- ✓ InO-Bot einschalten
- ✓ Bluetooth am PC aktivieren und anschließend im System Tray auf das Bluetooth Icon klicken
- ✓ „**Bluetooth-Gerät hinzufügen**“ auswählen



- ✓

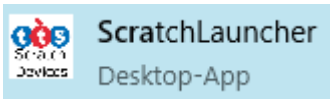


- ✓ Sobald der InO-Bot gefunden wurde, Headphone-Symbol 

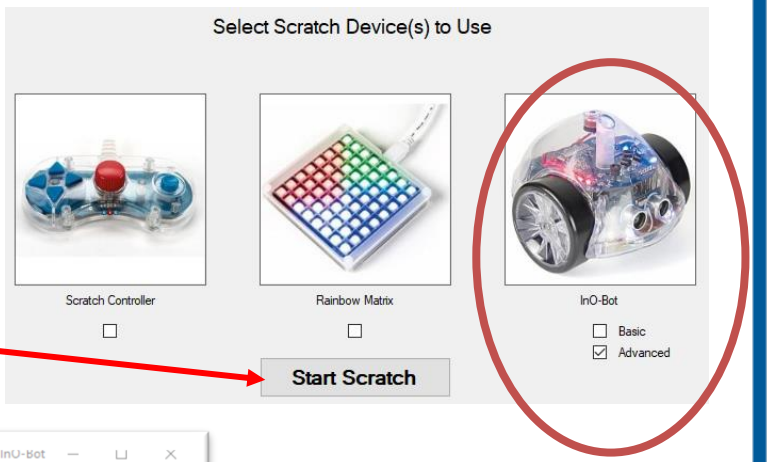
anklicken, um den Roboter mit dem PC zu verbinden

Programmierung starten

- ✓ **InO-Bot** einschalten
- ✓ **Scratch Launcher** starten

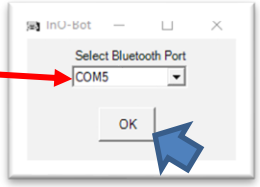


- ✓ **InO-Bot** auswählen
- ✓ Gewünschten **Modus** (**Basic** oder **Advanced**) auswählen
- ✓ **Start Scratch**-Schaltfläche anklicken



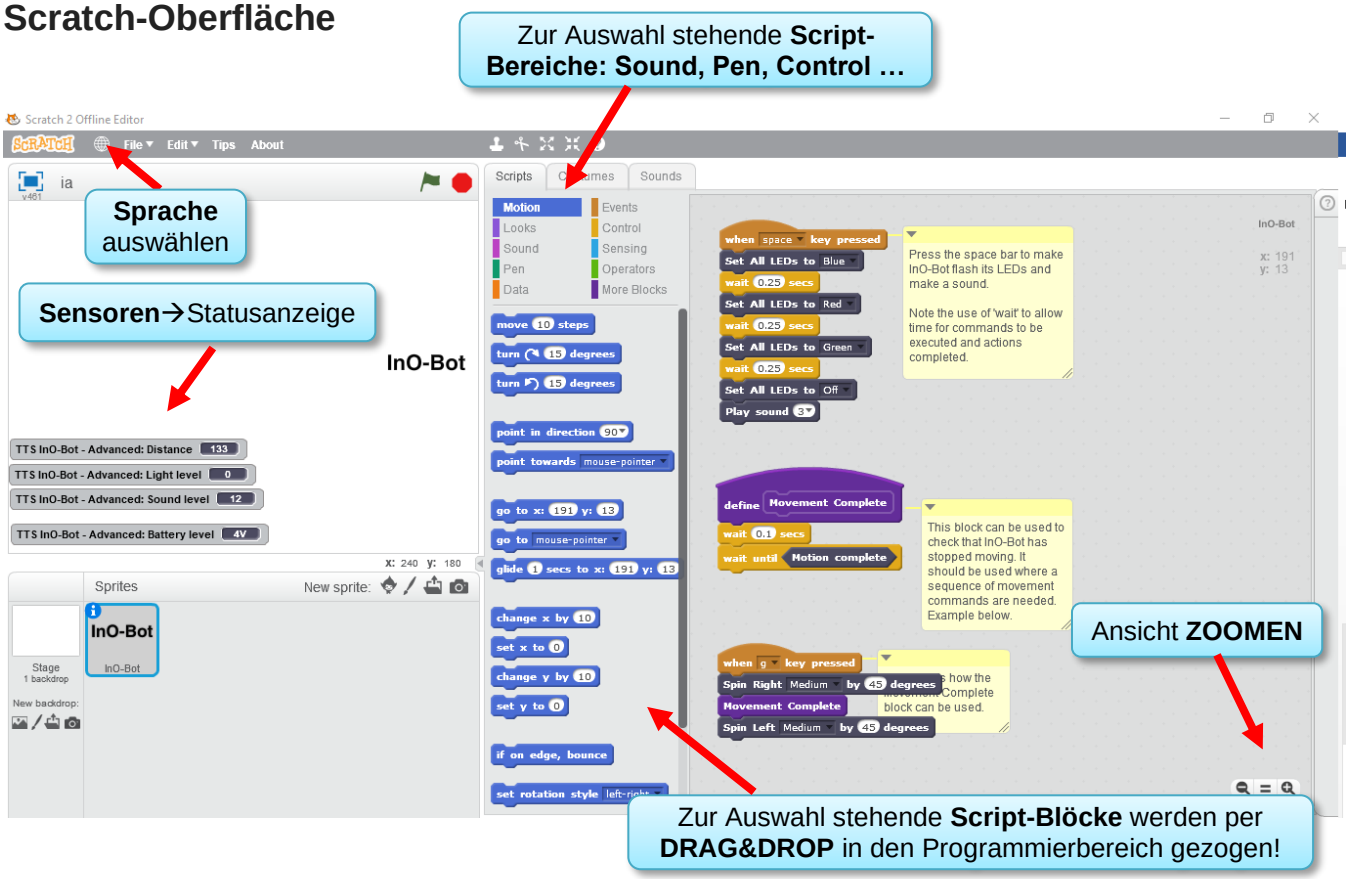
- ✓

- ✓ **Bluetooth-Port** auswählen (Voreinstellung beibehalten!)



- ✓ **Scratch Programmierumgebung** wird aufgerufen

Scratch-Oberfläche



Programmbeispiele

InO-Bot AB7 Lösung

Weitere Aufgaben findest du in den

ACTIONCards for InO-BOT

Einführung in die **Turtle-Grafik** mit **Scratch**

H. Milchram Oktober 2021

3.Thymio

Alter: **8+** (Thymio Challange Pack ca. 230€, Thymio II ab ca 130€)
https://www.thymio.org
https://youtu.be/25CXJF3m1qM
https://www.generationrobots.com/de/)

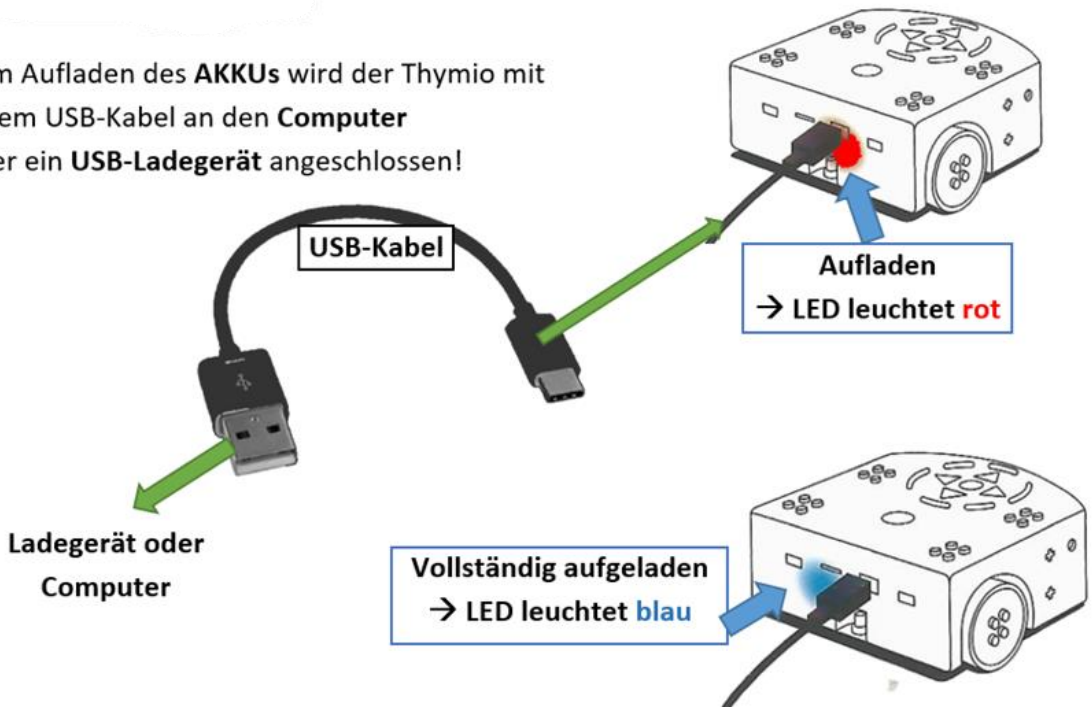
Thymio wurde an den technischen Hochschulen von Lausanne und Zürich entwickelt. Die gesamte Hardware, Software und alle Dokumentationen sind Open Source. Mit der **Aseba Software Suite** steht eine einfache Programmierumgebung zur Verfügung, die es ermöglicht, den Roboter sowohl textbasiert als auch über eine grafische Oberfläche visuell zu programmieren. Auch die Programmierung mit **Scratch** und **Blockly** ist möglich.

Technische Details

- ✓ Maße: 11 cm x 11,2 cm x 5,3 cm; 270 Gramm
- ✓ Li-Po Akku: 3,7 V, 1 ' 500 mAh, wiederaufladbar über einen MicroUSB-Port
- ✓ 2 Räder, max. Geschwindigkeit: 14 cm/s
- ✓ **Sensoren**: Infrarotsensoren, Beschleunigungsmesser, Thermometer, Infrarotempfänger für Fernbedienung
- ✓ **Aktoren:** 2 Motoren, 1 Lautsprecher, 39 LEDs
- ✓ 5 berührungssensitive Knöpfe, 1 Mikrofon
- ✓ Die **Aseba Software Suite** kann unter Linux, MacOS und Windows installiert werden
- ✓ Stifthalter
- ✓ Lego-kompatible mechanische Befestigungen

Thymio aufladen

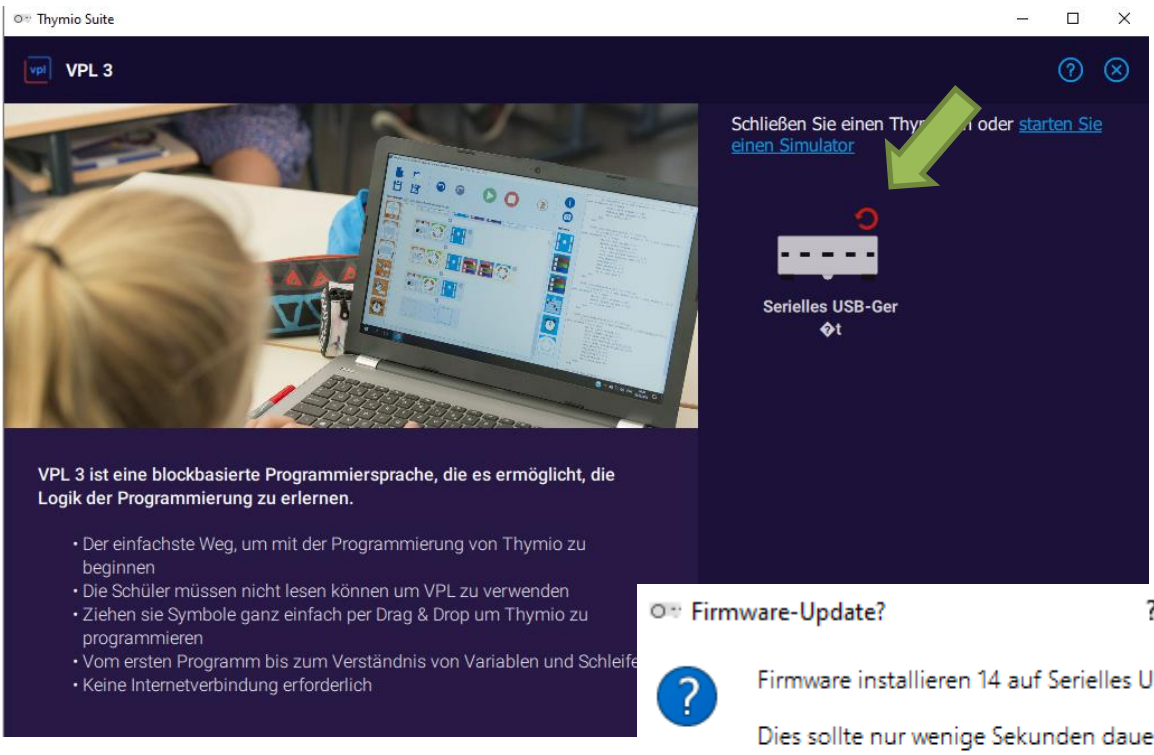
Zum Aufladen des **AKKUs** wird der Thymio mit einem USB-Kabel an den **Computer** oder ein **USB-Ladegerät** angeschlossen!

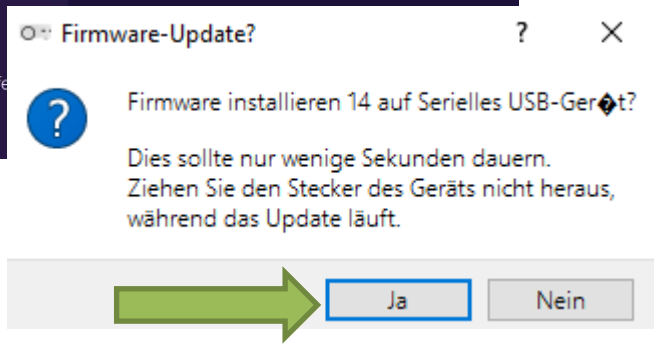


Thymio Firmware Upgrade

- ✓ Thymio Suite downloaden und installieren
- ✓ Thymio einschalten und mit USB-Kabel am Computer anschließen
- ✓ Thymio Suite starten

- ✓ Starte eine beliebige der 5 zur Auswahl stehenden Programmiersprachen. (zB VPL3)
  Steht ein Update zur Verfügung, siehst du rechts oben neben dem Robotersymbol einen roten Pfeil 





- ✓ Update-Symbol  mit der linken Maustaste anklicken und im angezeigten Fenster „JA“ auswählen!

- ✓ Ein geglücktes Firmwareupgrade wird durch eine kurze Tonfolge am Thymio angezeigt!

Firmware-Updates sollten regelmäßig durchgeführt werden!

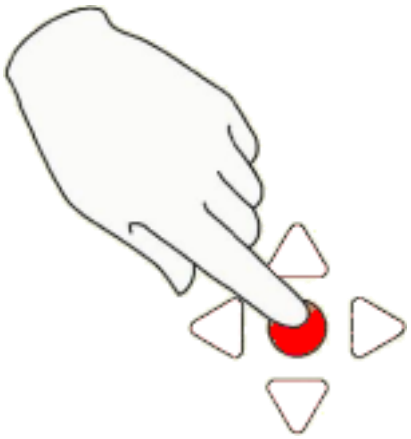
H. Milchram Oktober 2021

Thymio Einstellungen ändern

Das **Einstellungsmenü** erreicht man, wenn man im Menü der vorprogrammierten Verhaltensmuster gleichzeitig die **linke** und **rechte Taste** für **3 Sekunden drückt** und anschließend die **Auswahl mit der mittleren Sensortaste bestätigt**.

Gelb Lautstärk

Grün Kalibrierung der Motoren

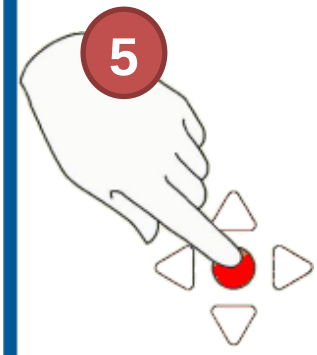


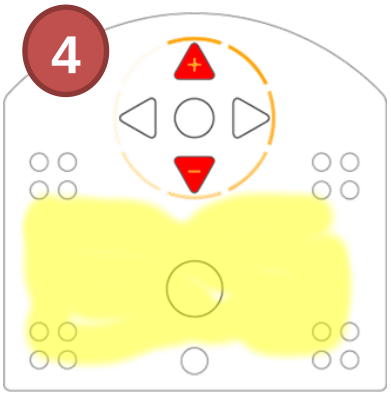
Rosa Wireless Einstellungen

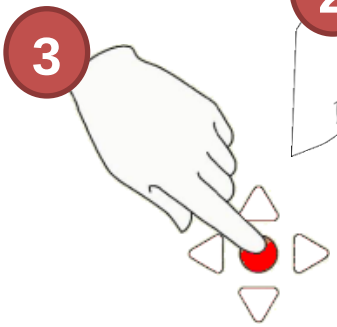
Um die **Einstellungen** zu **speichern**, musst du den **Thymio ausschalten**.

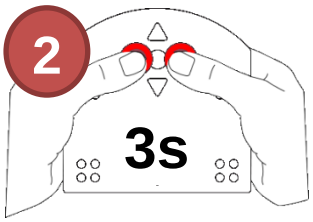
Lautstärke ändern

1. **Verhaltensmuster** „Neugier“ (**gelb**) auswählen
2. Wechsle ins **Einstellungsmenü**
3. Auswahl mit der mittleren Sensortaste bestätigen
4. Lautstärke mit den Sensortasten „**VOR→lauter**“ und „**Zurück→leiser**“ ändern
5. Speichern der Einstellungen →Thymio ausschalten!







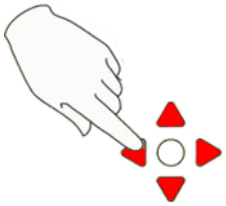


Motoren kalibrieren

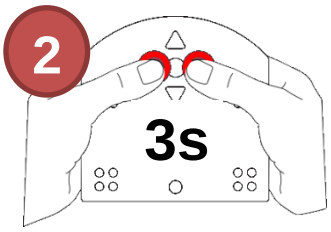
Wenn Thymio nicht mindestens 40 cm geradeaus fährt, müssen die Motoren kalibriert werden!

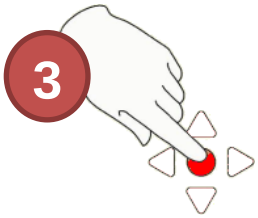
Die **Kopiervorlage** für die **Kalibrierung der Motoren** findest du im **Anhang**!

1. **Verhaltensmuster** „Freundlich“ (**grün**) auswählen.
2. Wechsle ins **Einstellungsmenü.**
3. Auswahl mit der mittleren Sensortaste bestätigen.
4. Mit den **Sensortasten** wird dein Thymio **vor** und **zurück** bewegt. Mehrmaliges Drücken erhöht die Geschwindigkeit.
5. Die Sensortasten für links und rechts ändern die **Kurvenkorrektur.**
6. Sobald der Roboter geradeaus fährt, die **mittlere Sensortaste** berühren, dadurch stoppen die Motoren.
7. Zum Speichern der Einstellungen **Thymio ausschalten**!



Kalibrierung der Motoren
Dein Roboter sollte während der Vor- oder Rückfahrt zwischen den beiden Begrenzungslinien bleiben!





Thymio Wireless-Einstellungen

Der Wireless-Thymio wird zusammen mit einem USB-Adapter (**Dongle**) ausgeliefert. Die Thymios sind so konfiguriert, dass sich alle im selben Netzwerk befinden und damit untereinander kommunizieren können und auch von einem Dongle aus gemeinsam angesprochen werden können.

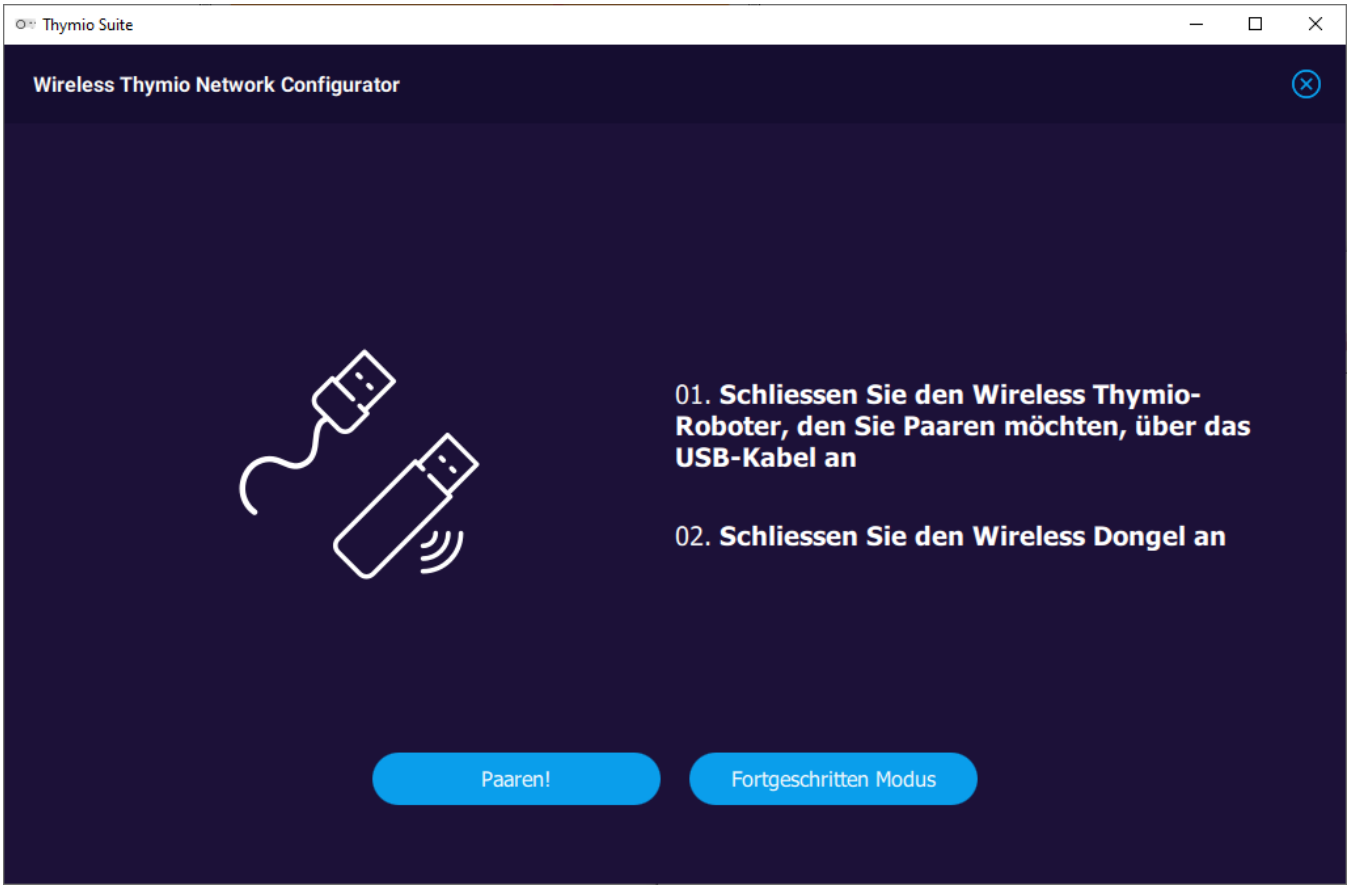
- ✓ alle Thymios befinden sich im selben Funkkanal (es stehen 3 Kanäle 1,2,3 zur Auswahl)
- ✓ alle Thymios haben dieselbe Netzwerk-Kennung (pan-ID)
- ✓ alle Thymios besitzen eine eindeutige Knotenkennung (node-ID)

Bei der Arbeit mit mehreren Thymios im Klassenverband kann es von Vorteil sein, wenn jeder Dongle nur mit genau einem Thymio im selben Netzwerk ist.

Dongle mit Thymio paaren

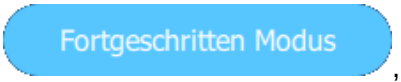
Einstellungen auf dem PC

- ✓ Dongle am Computer anstecken
- ✓ Thymio Suite starten und rechts oben unter der Titelzeile das Symbol  auswählen! Die Thymio-Toolbox wird angezeigt!
- ✓ „*Paaren Sie einen Wireless Thymio mit einem Wireless dongle*“ auswählen!
  Es öffnet sich der Wireless Thymio **Netzwerkkonfigurator.**



- ✓ Mit 

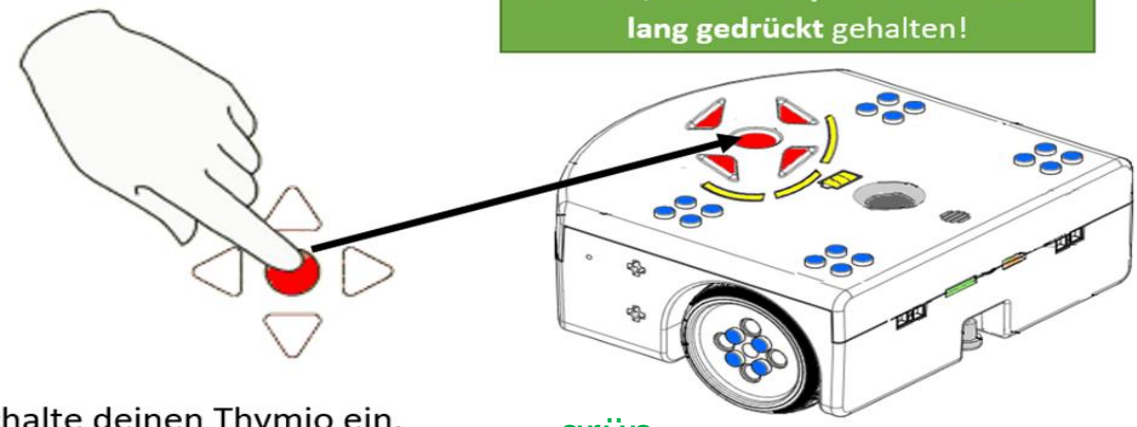
werden die Standardwerte gesetzt (Kanal 2, Netzwerkkennung 404C)

Sollen mehrere Thymios gepaart werden wählt man den 

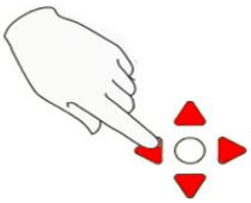
, hier kann jedem Thymio die Netzwerkkennung individuell zugewiesen werden! Noch einfacher geht es mit der Auswahl „*Paaren Sie einen Koffer von Wireless Thymio*“.

Thymio einschalten

Zum **Ein**- und **Ausschalten** wird der mittlere, runde **Knopf** ca. **3 Sekunden lang gedrückt** gehalten!

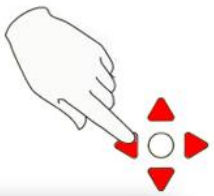


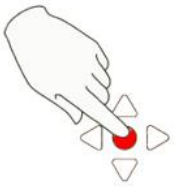
1. Schalte deinen Thymio ein.
   Welche Farbe zeigen die LEDs? grün
2. Mit den Pfeiltasten kannst du die Farbe der LEDs ändern.
   Wie viele Farben gibt es? 6



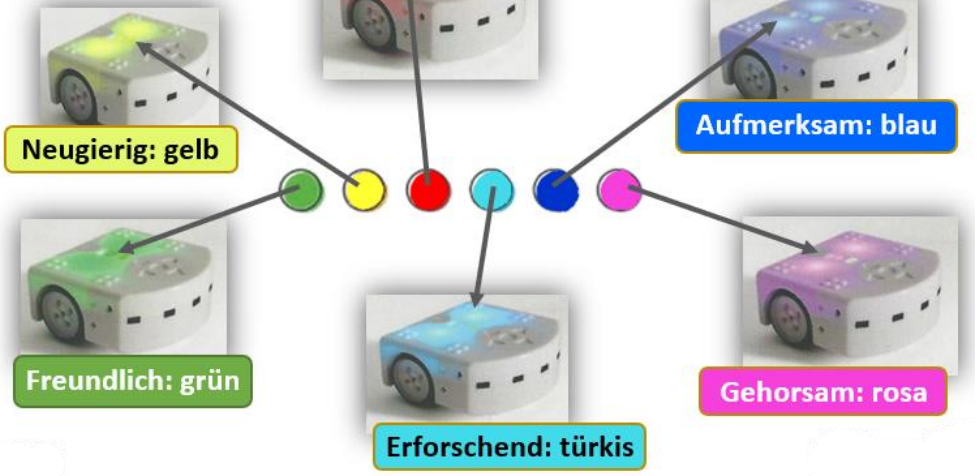
Thymio Verhaltensmuster

Nach dem Einschalten stehen dir 6 **vorprogrammierte Verhaltensweisen** zur Verfügung, die durch unterschiedliche Farben angezeigt und mit den **Pfeiltasten** ausgewählt werden.





Deine Auswahl musst du mit der **mittleren Sensortaste** bestätigen!



So können Kinder bereits im Vorschulalter entdecken, wie Thymio reagiert, wenn er mit seinen Sensoren ein Hindernis wahrnimmt. Sie lernen dabei bewusst sein Verhalten mit den Händen oder mit einem Parcours aus Bauklötzen oder Alltagsgegenständen zu beeinflussen.

Thymio AB1 Lösung

Nach dem Einschalten deines Thymio stehen dir **sechs** vorprogrammierte Verhaltensweisen zur Verfügung. Benutze die **Pfeiltasten** auf dem Roboter, um zwischen den einzelnen Verhaltensmustern zu navigieren. Über die **Taste in der Mitte** wird ein **Verhaltensmuster aktiviert** oder **deaktiviert**.

Gehe auf die Website **https://www.thymio.org/de/grundlegende-verhaltensweisen/** und informiere dich über die verschiedenen Verhaltensmuster und probiere sie aus.

GRÜN: Freundlich
Thymio folgt der Hand und reagiert auf einen anderen freundlichen Thymio.

GELB: Neugierig
Thymio vermeidet Hindernisse und stoppt automatisch am Tischrand oder wenn die Unterlage schwarz ist.

ROT: Ängstlich
Thymio entfernt sich, wenn du dich ihm mit deiner Hand näherst und er reagiert lautstark, wenn er in die Ecke gestellt oder in die Luft geworfen wird.

BLAU: Aufmerksam
Thymio ändert seine Farben und bewegt sich je nach Anzahl der erkannten Klatschtöne.
1x Klatschen: dreht nach rechts / fährt vorwärts
2x Klatschen: Start/Stopp **3x Klatschen**: fährt einen Kreis

TÜRKIS: Erforschend
Thymio folgt einer schwarzen Spur auf dem Boden. Die Spur sollte mindestens 3 cm breit sein.

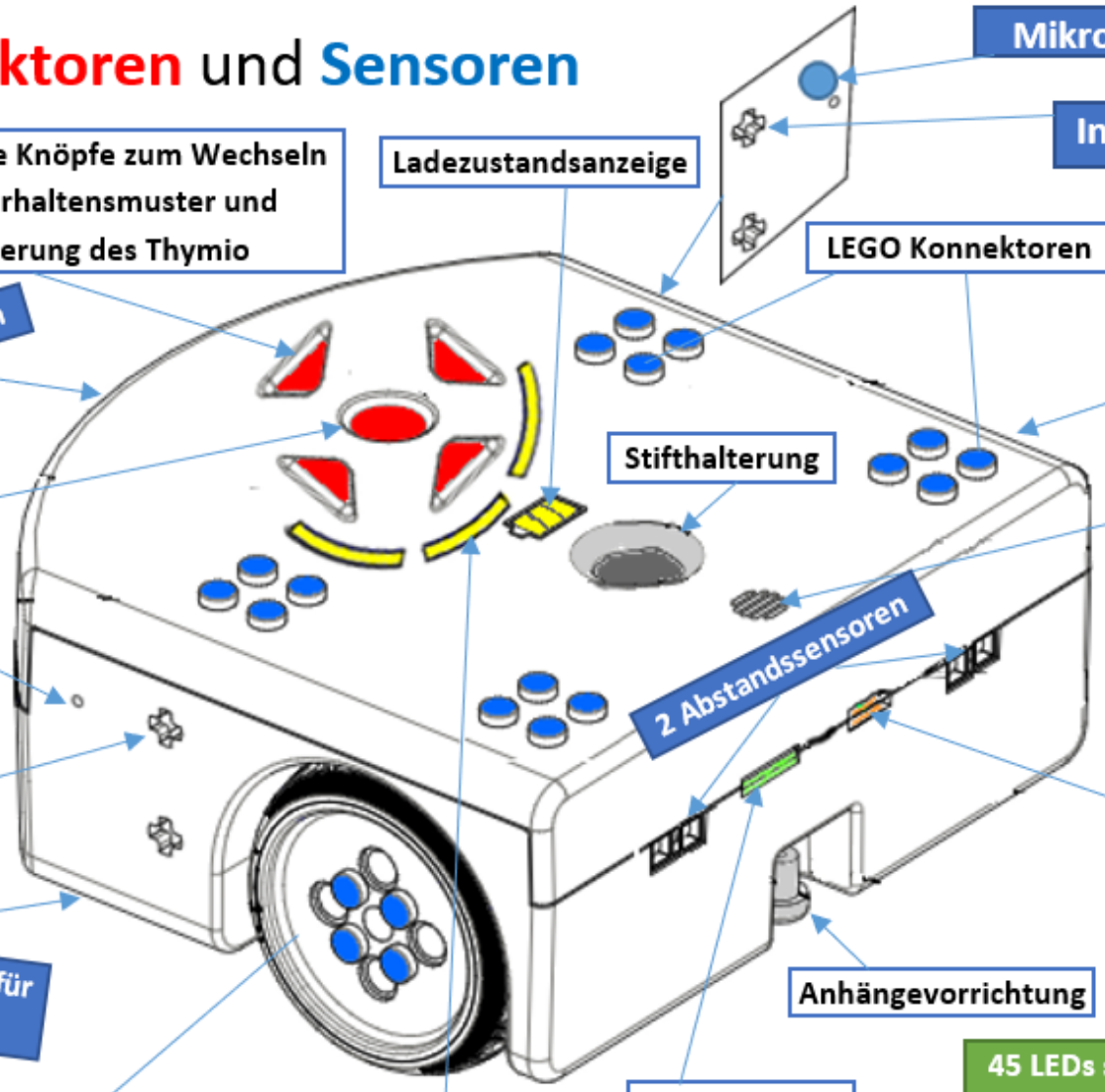
ROSA: Gehorsam
Thymio reagiert auf Tastenbefehle und die Fernbedienung. Bei mehrmaligem Betätigen der Tasten, beschleunigt oder verlangsamt sich Thymio.

Weitere Übungsaufgaben zu den verschiedenen Verhaltensweisen:

Thymio AB2

ACTIONcards for thymio Verhaltensmuster auf der Website **http://hemi.bplaced.net/Robotik/Roboter.htm**

Thymio Aktoren und Sensoren



Mikrofon

Infrarotempfänger

5 sensitive Knöpfe zum Wechseln der Verhaltensmuster und Steuerung des Thymio

Ladezustandsanzeige

LEGO Konnektoren

Rechtes Rad mit Motor

5 Abstandssensoren

Ein-/Ausschalt-Knopf

Stifthalterung

Lautsprecher

Resetknopf

2 Abstandssensoren

Beschleunigungssensor erkennt Erschütterungen und Schwerkraft

Temperatursensor

USB Anschluss zum Aufladen und die Übertragung von Programmen.

2 Bodensensoren für Linienverfolgung

Anhängevorrichtung

45 LEDs sorgen für unterschiedliche Farben

Slot für Micro SD-Card

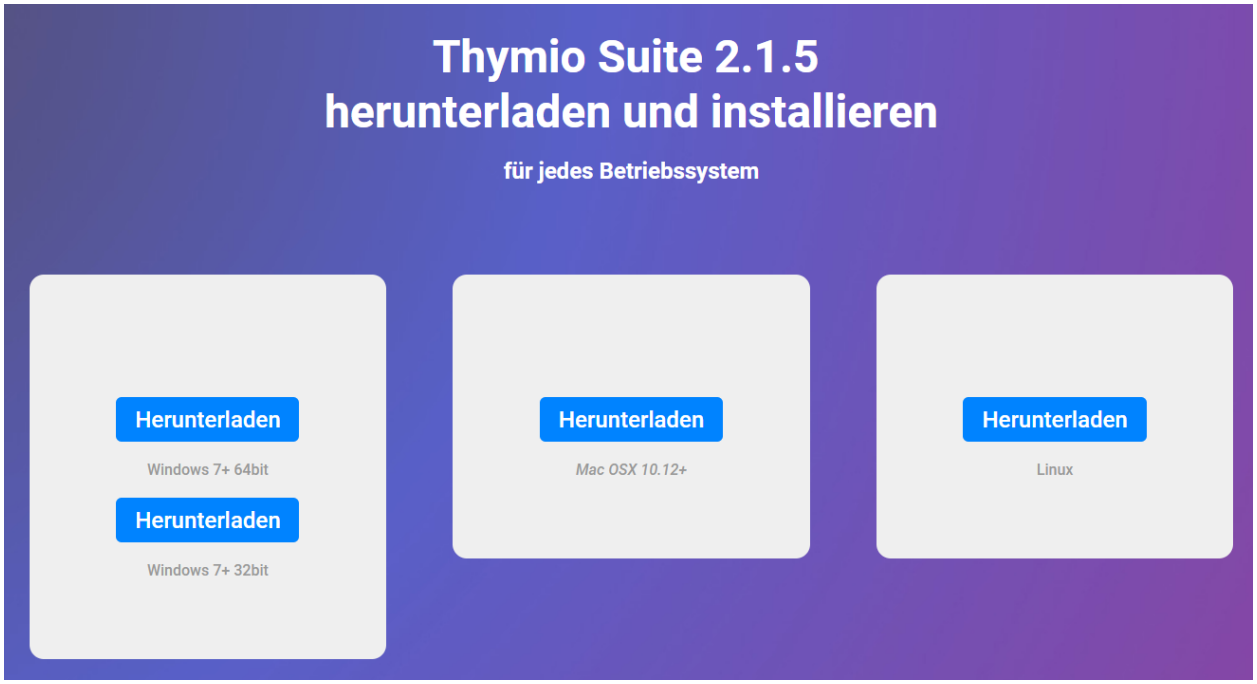
Linkes Rad mit Motor

Verhaltensmuster Aktivierungsanzeige

H. Milchram, Mai 2021

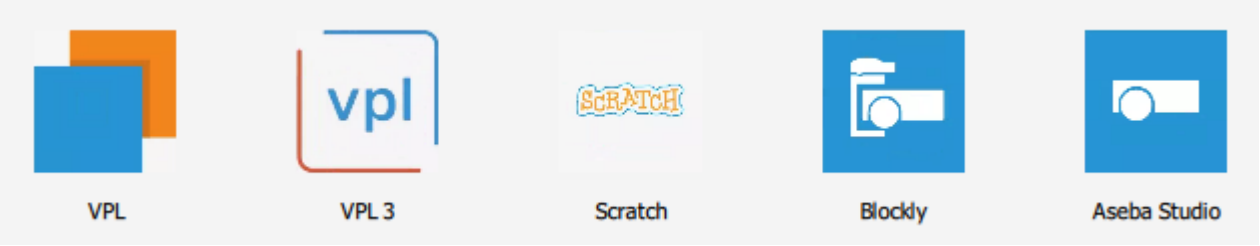
Wir lernen den Thymio programmieren

Das zentrale Programm für die Arbeit mit dem Thymio ist die **Thymio Suite**. Mit dieser Software kann der Thymio in 5 unterschiedlichen Programmiersprachen VPL, VPL3, Scratch, Blockly, ASEBA) gesteuert werden. Die Software steht für die Betriebssysteme Windows, MacOS und LINUX zur Verfügung und kann gratis von der Website https://www.thymio.org/ heruntergeladen werden.

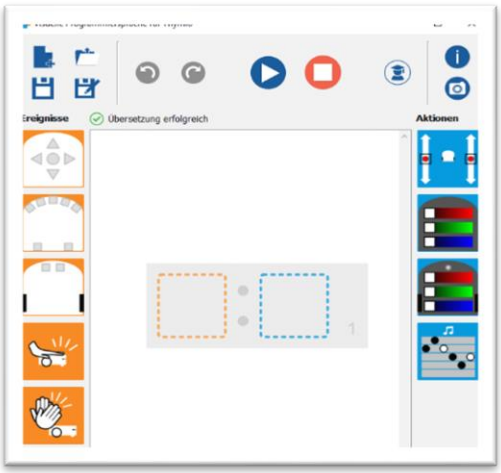


Thymio Suite

Die Thymio Suite ist eine Sammlung von Werkzeugen für die **Konfiguration** (wireless dongle), zum Starten von **Simulatoren** und die **Programmierung** in fünf verschiedenen Programmiersprachen.

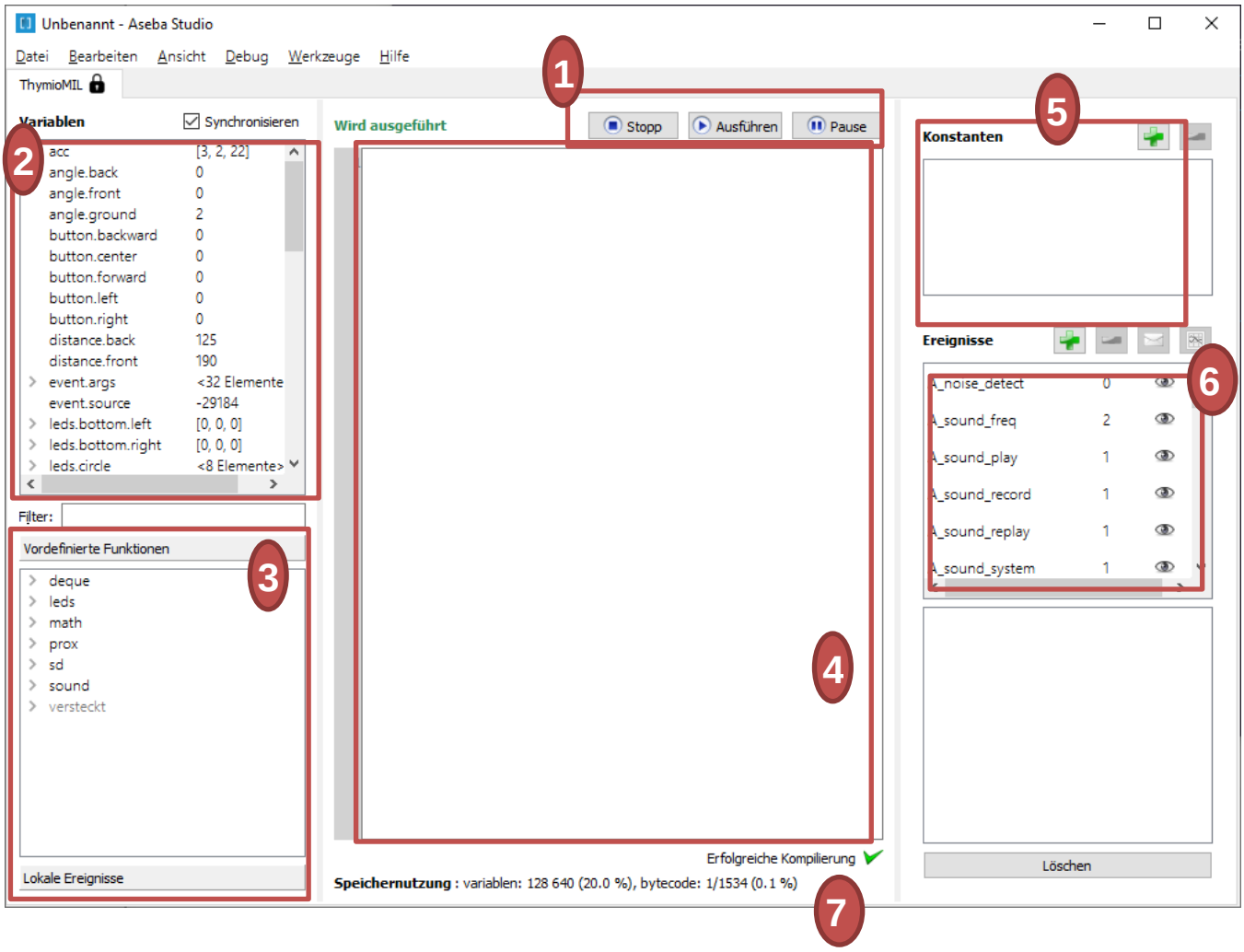


VPL (Visual Programming Language) und **VPL3** sind einfache, blockorientierte Programmiersprachen, die es schon Kindern ohne Lesekenntnisse ermöglicht, die Logik der Programmierung zu erlernen.



Für bereits fortgeschrittene Programmierer eignen sich die ebenfalls blockorientierten Programmiersprachen **BLOCKY** und **SCRATCH**.

Für die **Programmierprofis** steht mit dem **ASEBA Studio** eine textbasierte Programmiersprache zur Verfügung, die eine präzise Kontrolle über alle Möglichkeiten des Roboters ermöglicht.

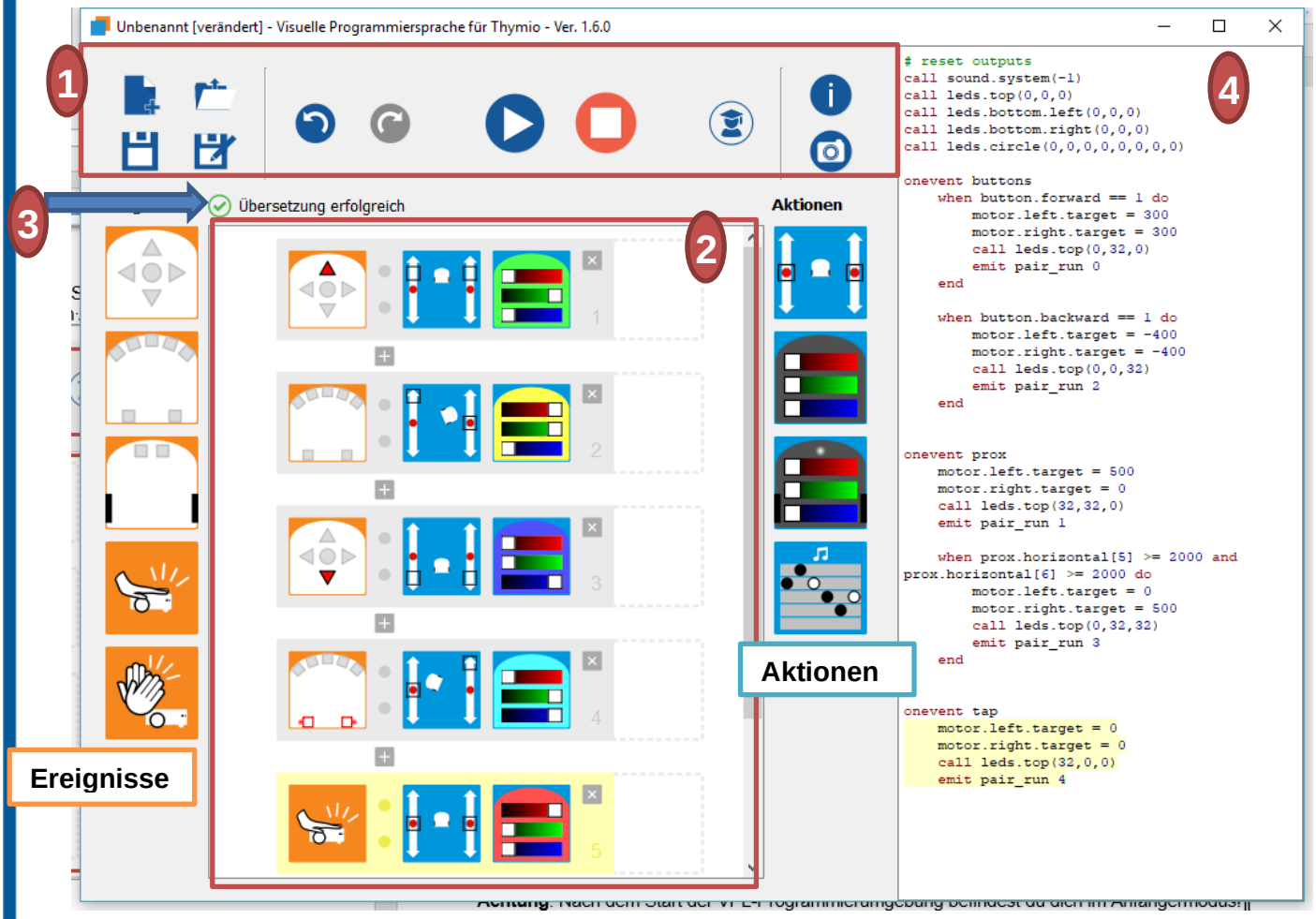


1. Mit diesen Knöpfen kannst du deinen eigenen Code auf den Roboter laden und starten.
2. Dieses Fenster zeigt den Speicher des Roboters an: die Werte der Sensoren, der Aktoren und der Variablen.
3. Hier findest du die vorprogrammierten Funktionen, lokale Ereignisse und Werkzeuge, die du in deinem Code verwenden kannst.
4. Hier wird der Code geschrieben. Dieser Code wird nachher das Verhalten vom Roboter steuern.
5. Hier können Konstanten bestimmt werden.
6. Hier hast du die Kontrolle über die Ereignisse und kannst eigene Ereignisse hinzufügen.
7. Diese Zeile zeigt, ob der Code richtig geschrieben wurde

Visuelle Programmierumgebung mit VPL

Bei der visuellen Programmierung mit **VPL** (Visual Programming Language) verwendet man auf intuitive Weise **Ereignis**- und **Aktionsblöcke**.

1. Thymio einschalten und dazugehörigen Wireless-Stick am PC in eine USB-Schnittstelle stecken.
2. **Thymio Suite**  starten und VPL  auswählen. (Sollte das Programm nicht auf deinem PC installiert sein, kannst du es unter der folgenden URL downloaden: **https://www.thymio.org/de:start**



1. **Werkzeugleiste (Toolbar)**: Die Werkzeugleiste enthält Tasten zum Öffnen und Speichern von Dateien, zum Laden und Stoppen des geschriebenen Programm-Codes, sowie zum Ändern des Programmier-Modus.
2. **Programmierfenster**: Hier werden die Blöcke für die Ereignisse und Aktionen abgelegt.
3. **Statuszeile**: Hier wird angezeigt, ob die eingegebenen Befehle vollständig sind.
4. **Programmcodefenster**: Hier wird der **Quellcode** angezeigt.

Erklärung der Symbole der Toolbar

 neu	Diese Taste löscht den bisher programmierten Code und stellt eine leere Programmierumgebung
 Dokument öffnen	Diese Taste öffnet eine bestehende Datei.
 speichern	Diese Taste speichert den Programmcode in einer neuen Datei ab.
 speichern als	Diese Taste speichert den Programmcode in einer neuen Datei mit neuem Namen ab.
 laden und ausführen	Diese Taste lädt den Code auf den Roboter und führt ihn aus.
 stoppen	Diese Taste hält den Roboter an. Sobald man den Roboter angehalten hat, muss der Code erneut auf den Roboter geladen werden.
 fortgeschrittener Modus	Diese Taste schaltet die Programmier-Umgebung in den fortgeschrittenen Modus um (**advanced**). In diesem Modus stehen zusätzliche Funktionen zur Verfügung.
 Information	Diese Taste lädt dieses Referenzblatt.
 Bildschirmfoto	Diese Taste macht ein Bildschirmfoto (Screenshot) des VPL-Programmcodes.

Achtung: Nach dem Start der VPL-Programmierumgebung befindest du dich im Anfängermodus!

Vor dem Schließen der **VPL-Programmierumgebung,** unbedingt ein eventuell noch laufendes Programm mit der Schaltfläche  beenden!

VPL Referenzkarte →Ereignisse (Reaktionen auf Sensoren)

Gedrückte Knöpfe:

Grau → ignorieren

Rot → Die zugeordnete Aktion wird ausgeführt.

Hinderniserkennung (Front- und Hecksensoren)

Grau → ignorieren

Weiß mit roter Umrahmung → Aktion, wenn Objekt in der Nähe.

Bodensensoren

Grau → ignorieren

Weiß mit roter Umrahmung → Aktion, wenn Boden vorhanden.

Klopfen auf den Roboter, Erschütterung

Bei Erschütterung des Roboters werden die zugeordneten Aktionen ausgeführt.

Klatschen

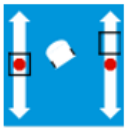
Der Roboter reagiert auf laute Geräusche und führt die zugeordnete Aktion aus.

Kombinieren verschiedener Sensoren

Werden mehreren Sensoren gleichzeitig Aktionen zugeordnet, müssen alle Bedingungen erfüllt sein, um zugeordnete Aktionen auszuführen.

Nur wenn beide Tasten gedrückt werden , stoppt der Motor!

VPL Referenzkarte →Aktionen (Aktoren steuern)



Geschwindigkeit der beiden Motoren regeln

Durch Verschieben des kleinen Quadrates wird die Geschwindigkeit des rechten und linken Motors eingestellt.

Farbauswahl für die LEDs an der Oberseite

Durch Verschieben der drei Regler für die Grundfarben ROT, GRÜN, BLAU kann die Farbe für die LEDs auf der Oberseite eingestellt werden (**additive**

Farbauswahl für die LEDs an der Unterseite

Musik abspielen

Linie**: Tonhöhe**

Farbe**: Tonlänge**

Diese Aktion spielt eine Melodie bestehend aus **sechs benutzerdefinierten Noten**. Die **Tonhöhe** wird durch Linien angezeigt. Um die Tonhöhe festzulegen, klickt man auf den gewünschten Balken. Je höher der Punkt, desto höher die Tonlage. Ein **weißer** Punkt erzeugt einen **doppelt so langen Ton** wie ein schwarzer Punkt. Ein fehlender Punkt bedeutet eine **Pause.**

Additiver Farbmischer:
https://bit.ly/2gp1TnV

Zusätzliche Informationen zu den einzelnen Ereignissen und Aktionen findet man auf der Website: https://www.thymio.org/de:thymiovpl

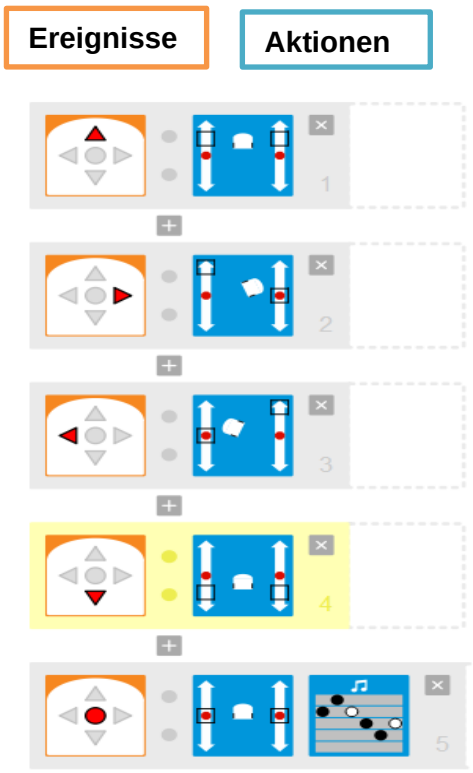
Bei der Programmierung werden die **Ereignisse** per „Drag & Drop“ in das linke und **Aktionen** in das rechte Quadrat gezogen. Findet das Ereignis statt, führt der Roboter die Aktion aus.

Unter der folgenden URL findest du eine ausführliche Anleitung zur visuellen Programmierung:
https://www.thymio.org/de:thymiovpl

Mein erstes Programm

ThymioAB3 Lösung

Programmiere deinen Thymio so, dass du ihn mit den Sensortasten steuern kannst. Durch Drücken der mittleren Sensortaste soll der Roboter gestoppt werden und eine kurze Melodie abspielen.



Quelltext

```
# variables for notes
var notes[6]
var durations[6]
var note_index = 6
var note_count = 6
var wave[142]
var i
var wave_phase
var wave_intensity

# compute a sinus wave for sound
for i in 0:141 do
wave_phase = (i-70)*468
call math.cos(wave_intensity, wave_phase)
wave[i] = wave_intensity/256
end
call sound.wave(wave)
# reset outputs
call sound.system(-1)
call leds.top(0,0,0)
call leds.bottom.left(0,0,0)
call leds.bottom.right(0,0,0)
call leds.circle(0,0,0,0,0,0,0,0)

# when a note is finished, play the next note
onevent sound.finished
if note_index != note_count then
call sound.freq(notes[note_index], durations[note_index])
note_index += 1
end

onevent buttons
when button.forward == 1 do
motor.left.target = 300
motor.right.target = 300
emit pair_run 0
end
```

```
utton.right == 1 do
motor.left.target = 500
motor.right.target = 0
emit pair_run 1
end

when button.left == 1 do
motor.left.target = 0
motor.right.target = 500
emit pair_run 2
end

when button.backward == 1 do
motor.left.target = -500
motor.right.target = -500
emit pair_run 3
end

when button.center == 1 do
motor.left.target = 0
motor.right.target = 0
call math.copy(notes[0:5], [440, 524, 440, 370, 311, 370])
call math.copy(durations[0:5], [7, 7, 14, 7, 7, 14])
note_index = 1
note_count = 6
call sound.freq(notes[0], durations[0])
emit pair_run 4
end
```

Thymio AB4 Lösung

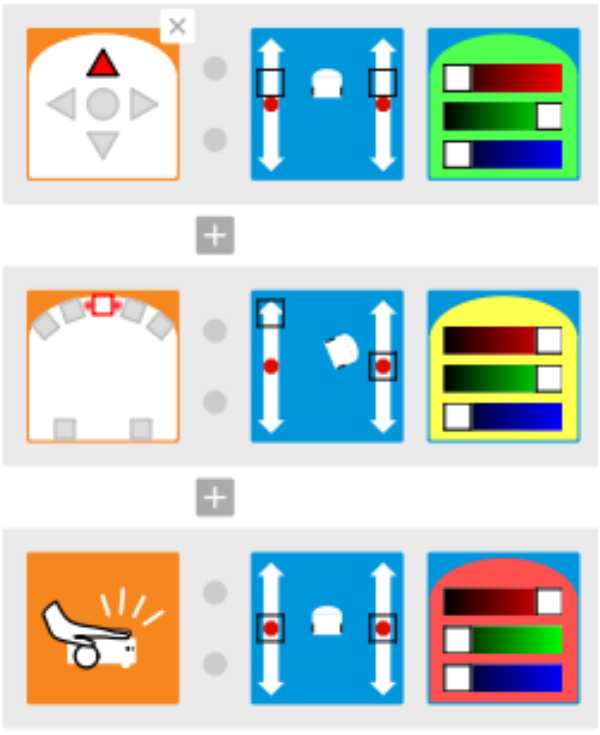
Beim Drücken der **vorderen Sensortaste** beginnt dein Thymio **langsam vorwärts zu fahren**, die oberen **LEDs leuchten** dabei **grün**.
Sobald der **mittlere Frontsensor** ein Hindernis erkennt beginnt er **rechtsherum im Kreis** zu fahren. Die oberen **LEDs leuchten** nun **gelb**.
Durch **Klopfen auf die Oberseite** des Thymio wird dein Roboter **gestoppt** und die **LEDs** beginnen **rot** zu leuchten.

Für die **Sensoren** benötigst du folgendes Ereignis:

Hinderniserkennung:
grau → ignorieren
Weiß mit roter Umrahmung:
→ Aktion, wenn Objekt in der Nähe
Schwarz → Aktion, wenn kein Objekt vorhanden

Am PC oder bei RGB-LEDs kann eine Farbe durch ihre Anteile an den drei **Primärfarben** Rot, Grün und Blau definiert werden. Betrachte den Farbkreis um zu ermitteln, wie du die Farbe „gelb“ einstellen kannst!

Klopfereignis
Wird der Roboter erschüttert (z.B. durch Klopfen auf die Oberfläche, oder anstoßen an ein Hindernis) erfolgt eine Aktion.



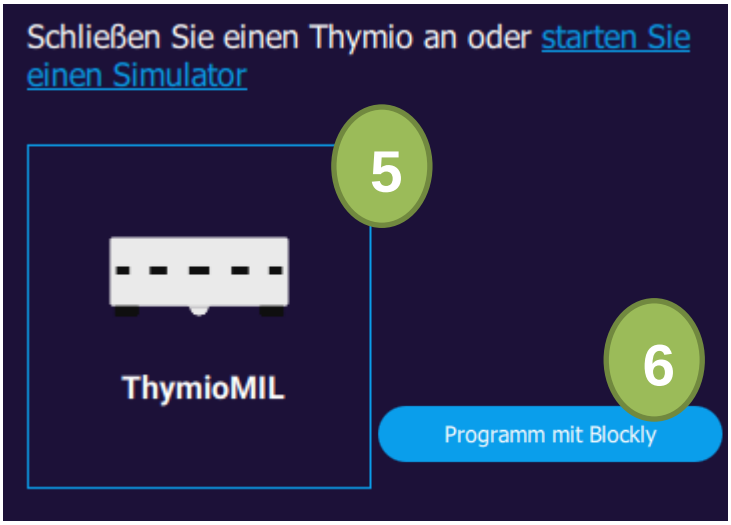
Thymio VPL Tutorial
→https://bit.ly/2RQFzIS

ACTIONcards for thymio VPL-Grundlagen

Weitere **Übungsaufgaben:**
zu finden auf der folgenden **Website** →http://hemi.bplaced.net/Robotik/Roboter.htm

Programmierung mit Blockly

1. Thymio-Suite starten
2. **Blockly**  auswählen
3. Thymio einschalten
4. Wireless Dongle am USB-Anschluss des Computers einstecken.
5. Thymio auswählen
6. Programm starten

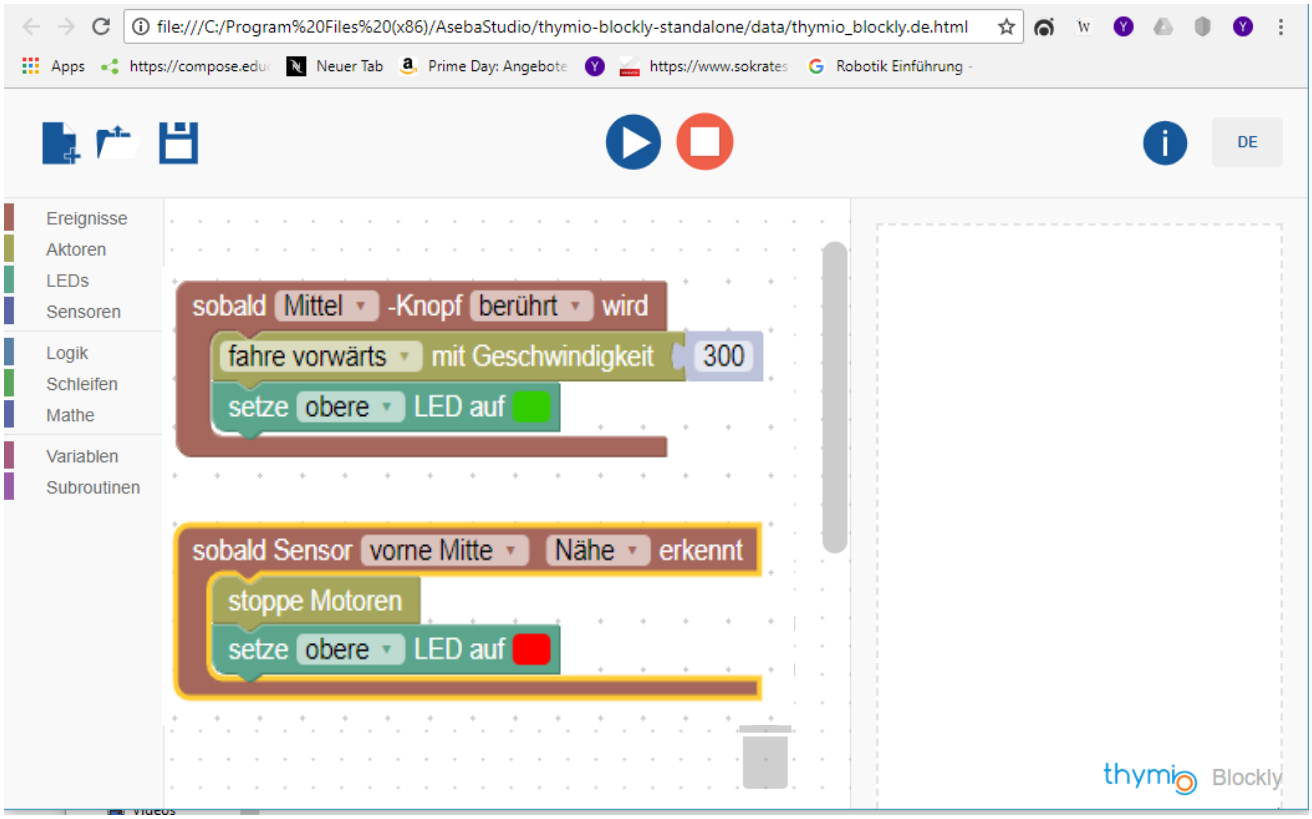


Thymio 5 Lösung

Beispiel für ein kleines Programm:

Sobald der **mittlere Knopf** am Thymio **berührt** wird, **fahre vorwärts**.

Die oberen **LEDs** sollen **grün** leuchten. Wenn der **mittlere Frontsensor** ein **Hindernis erkennt,** sollen die **Motoren stoppen** und die oberen **LEDs rot** leuchten.



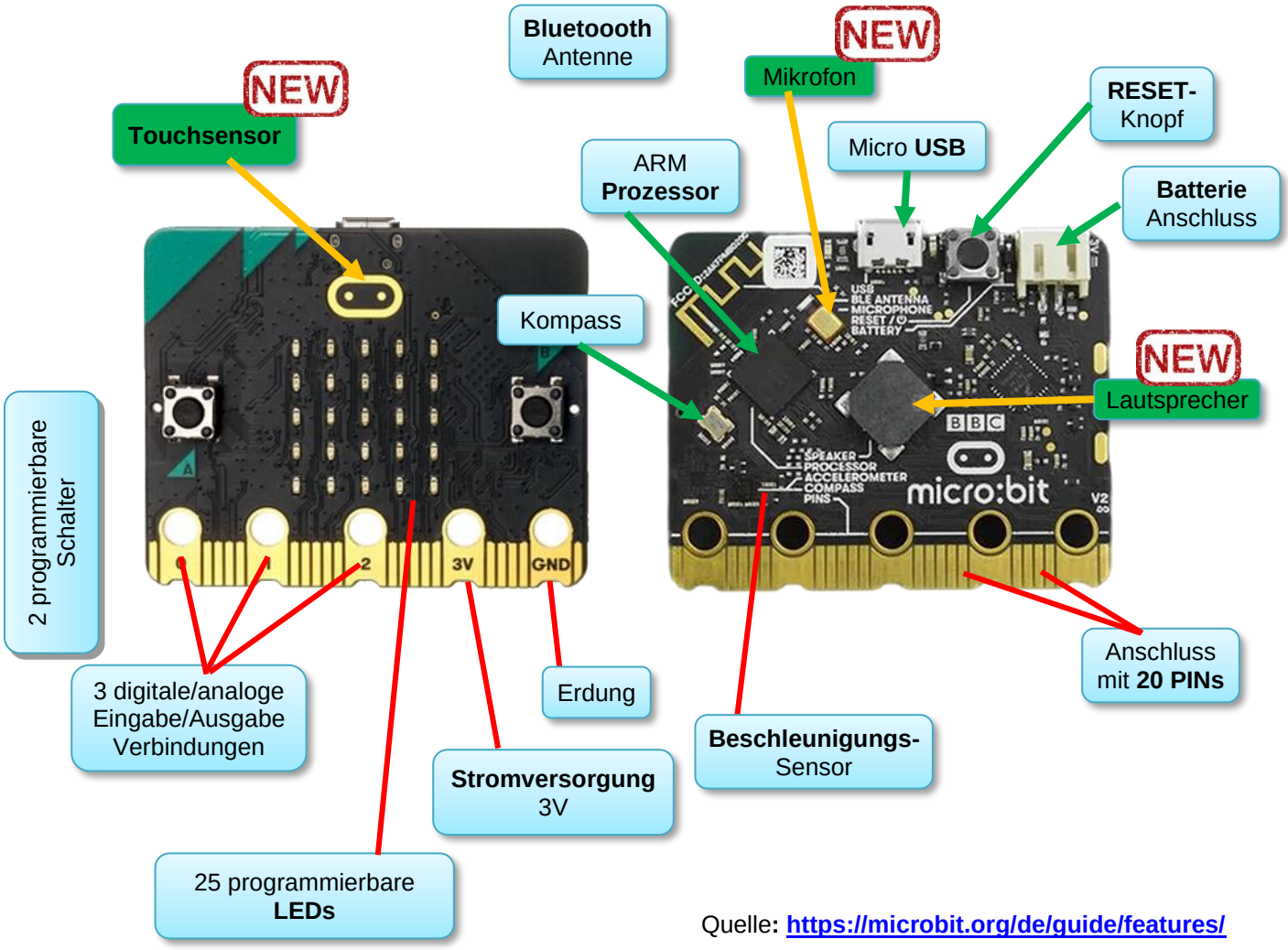
4. Micro:bit

Alter: 8+ (Micro:bit bei Amazon ca. 21€ https://austro-tec.at/bbc-microbit)

Der **micro:bit** ist ein nur etwa kreditkartengroßer, mit **Blockly**, **JavaScript, Scratch** und **Python** programmierbarer Computer.

Technische Details https://microbit.org/de/

- 25 individuell programmierbare LEDs zum Anzeigen von Zahlen, Buchstaben und Bildern
- 2 programmierbare Schalter
- 20 physikalische Anschlusspins: *Über diese PINs können Motoren, LEDs oder andere elektrische Komponenten oder zusätzliche Sensoren angeschlossen und gesteuert werden!*
- Licht- und Temperatur-Sensoren: *Durch die **Umkehrung der LEDs** des Bildschirms zu einem Eingang wird der LED-Bildschirm zu einem **Basis-Lichtsensor** der es ermöglicht, das Umgebungslicht zu erkennen. Der **Temperatursensor** ist im Prozessor integriert!*
- Bewegungsmelder (Beschleunigungssensor und Kompass)
- Drahtlose Kommunikation über Funk und Bluetooth
- USB-Schnittstelle
- **Lautsprecher/Mikrofon** + **Touch Sensor** (nur in der neuen Version **micro:bit v2**)

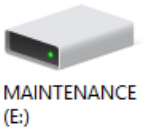


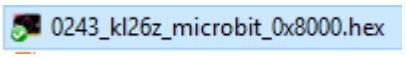
Quelle: https://microbit.org/de/guide/features/

Firmware-Update für den micro:bit

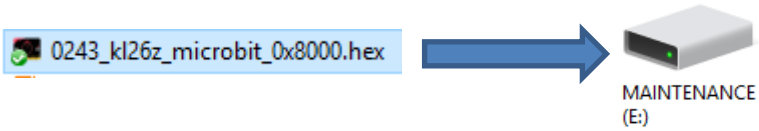
Download der aktuellen Firmware: **https://bit.ly/2OH8Lgp**
Anleitung für das Firmware Update: **https://bit.ly/2pv4QbI**

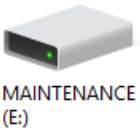
- ✓ Firmware (HEX-Datei) herunterladen und am PC speichern
- ✓ Micro-USB-Kabel am micro:bit anstecken
- ✓ **RESET-Knopf am Micro:bit gedrückt halten** und das Micro-USB-Kabel mit dem PC verbinden



- ✓ Am PC erscheint ein neues Laufwerk mit dem Namen **„Maintenance“**
- ✓ Nun muss die Datei 

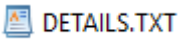
mit der aktuellen Firmware per **Drag&Drop** auf das Laufwerk **„Maintenance“** kopiert werden!



- ✓ Die gelbe **System-LED** beginnt zu blinken
- ✓ Nach erfolgreichem Flash-Update verschwindet das Laufwerk 

und es wird wieder das Laufwerk 

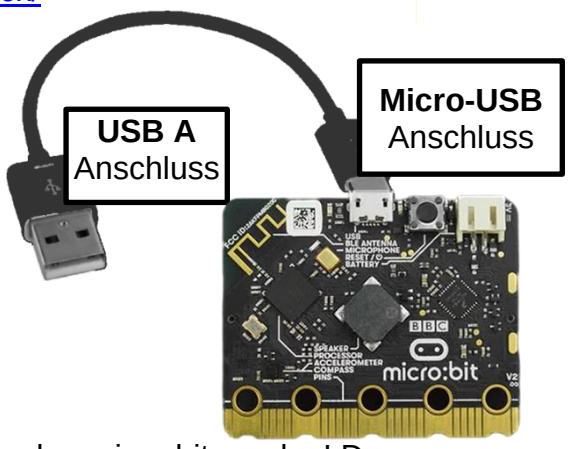
angezeigt.
- ✓ Laufwerk 

öffnen und überprüfen ob die Versions-Nr. in der Datei 

mit der Versions-Nr. des Firmware-Updates übereinstimmt!

```
# DAPLink Firmware - see https://mbed.com/daplink
Unique ID: 9900000048154e45003b9013000000510000000097969901
HIC ID: 97969901
Auto Reset: 1
Automation allowed: 0
Overflow detection: 0
Daplink Mode: Interface
Interface Version: 0243
Git SHA: b403a07e3696cee1e116d44cbdd64446e056ce38
Local Mods: 0
USB Interfaces: MSD, CDC, HID
Interface CRC: 0x14256f44
Remount count: 0
```

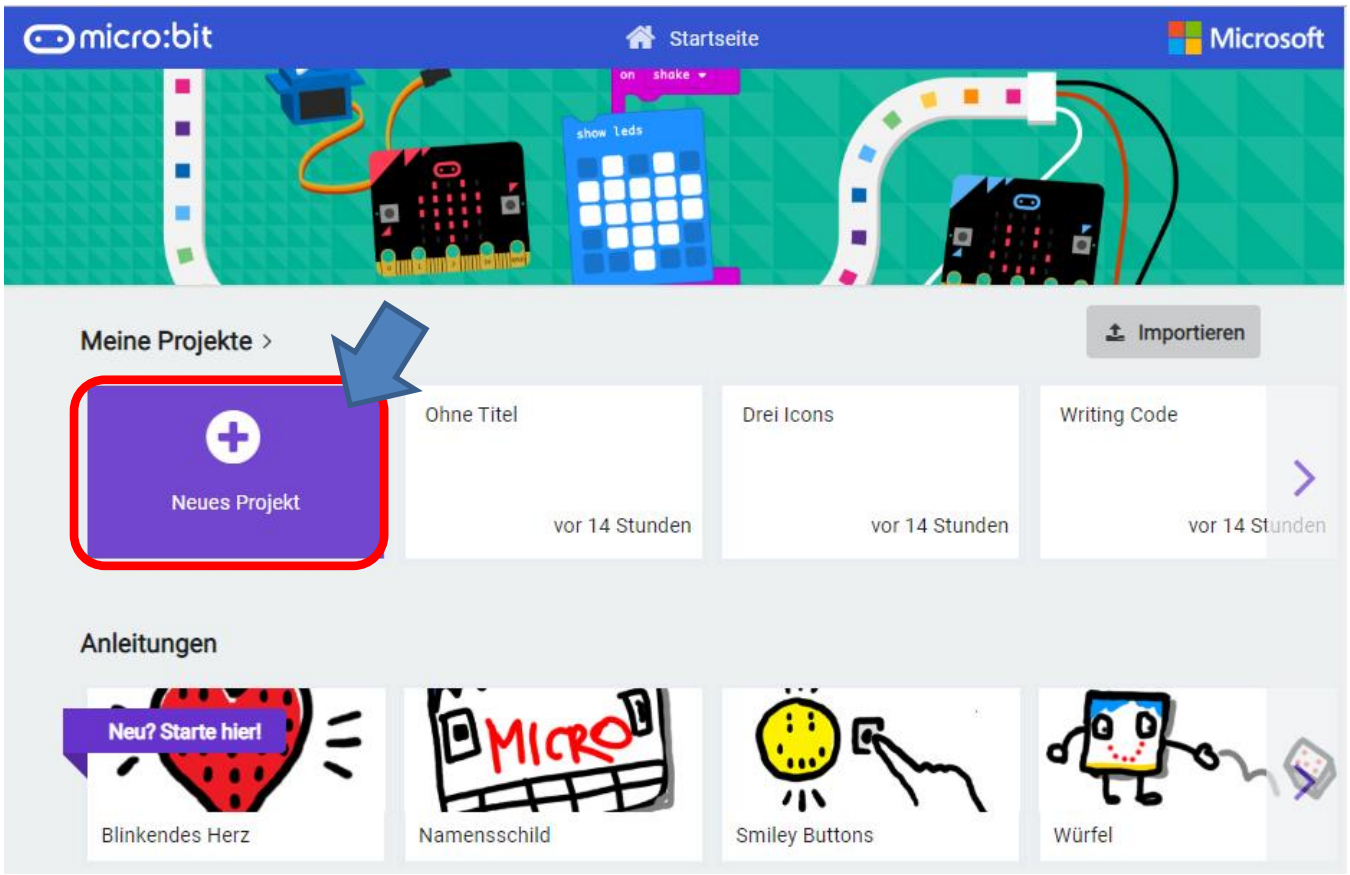
Erste Schritte https://microbit.org/de/guide/quick/



Schließe den **micro:bit** über ein **Micro-USB-Kabel** an deinen Computer an. MACs, PCs, Chromebooks und Linux-Systeme (einschließlich Raspberry Pi) werden unterstützt.

Der angeschlossene micro:bit wird auf deinem Computer als ***Laufwerk*** angezeigt.

Nun können wir unser **erstes Programm** schreiben und an den micro:bit senden! Dazu starten wir einen **BROWSER** und geben folgende URL ein: **https://makecode.microbit.org/** und starten ein **neues Projekt**!

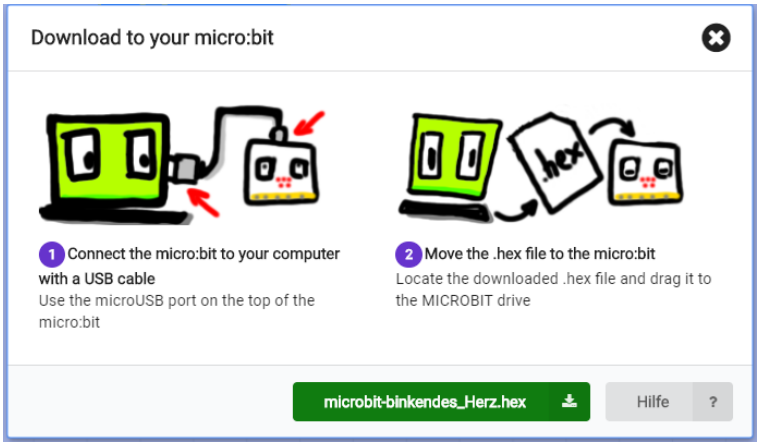


Die Programmieroberfläche mit auswählbaren Programmblöcken (Programmiersprache „**Blockly**“) wird geöffnet!

Die Auswahl der Sprache für die Programmierbefehle kann mit dem Zahnrad vorgenommen werden. Da die meisten professionellen Programmiersprachen aber englische Befehle verwenden, ist aus didaktischen Gründen auch bei der Verwendung von Blockly diese Sprache vorzuziehen!

micro:bit Programmierumgebung

- ✓ Programm-Blöcke per **Drag & Drop** in den Arbeitsbereich ziehen und das Programm mit dem **Simulator** testen.
- ✓ Über die Schaltfläche  wird das Programm als **HEX-Code** in deinem **Download-Ordner** gespeichert!



- ✓ **HEX-Datei** per Drag & Drop auf den micro:bit kopieren
  Der micro:bit pausiert und die gelbe LED auf der Rückseite des micro:bit blinkt, während dein Code gespeichert wird. Sobald der Code vollständig übertragen ist, wird er automatisch ausgeführt!

Auf dem micro:bit wird immer nur ein Programm gespeichert. → Bei der Übertragung eines neuen Programmes wird das alte Programm gelöscht!

Mein erstes Programm

Schreibe ein Programm „**Herzschlag**“, das auf deinem micro:bit ein schlagendes Herz anzeigt!

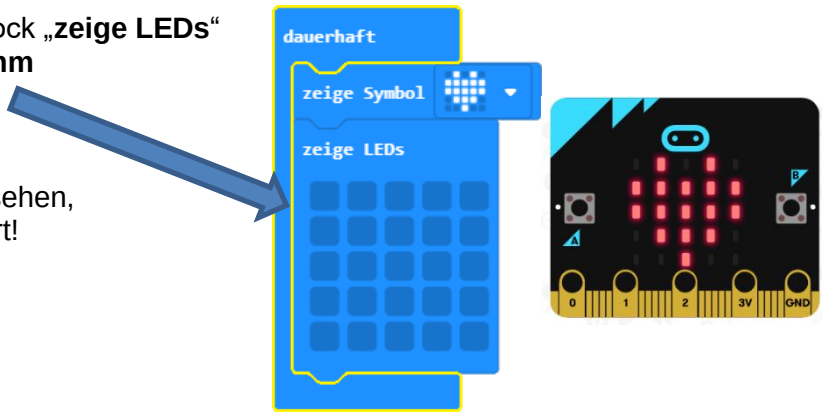
- ✓ **Programmierumgebung** für den micro:bit in deinem Browser öffnen: **https://makecode.microbit.org**
- ✓ Die Schaltfläche 

anklicken
- ✓ Aus dem Register 

den Block 

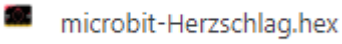
per **Drag&Drop** in den Arbeitsbereich auf den bereits vorhandenen Block „**dauerhaft**“ 

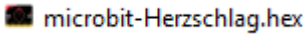
ziehen!
- ✓ Ergänze dein Programm um den Block „**zeige LEDs**“
- ✓ So sollte nun dein **fertiges Programm** aussehen!



Im **Simulationsbereich** kannst du sehen, ob dein Programm richtig funktioniert!

- ✓ Alle **nicht erforderlichen Blöcke** kannst du **löschen**, indem du sie in den Registerbereich ziehst!
- ✓ Gib deinem Programm den Namen 

und lade es durch einen Mausklick auf das **Diskettensymbol** auf deinen PC! Das Programm wird als HEX-Dateiim Download-Ordner gespeichert! 

- ✓ Die Datei 

per Drag&Drop auf das Laufwerk 

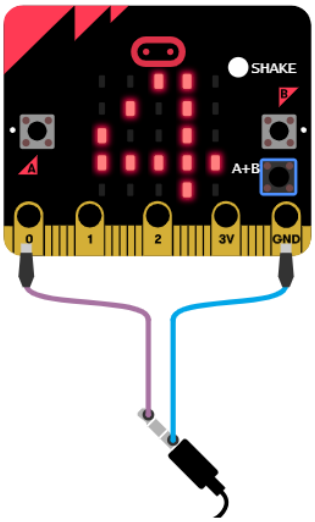
ziehen.

micro:bit Programmierung im Browser

Schreibe ein Progamm, dass auf deinem micro:bit durch Drücken der Schalter „**A**“, „**B**“, „**AB**“ und durch „**Schütteln**“ unterschiedliche Melodien abspielt! Dabei soll bei jeder Melodie eine andere Ziffer am Display eingeblendet werden!
A→(1), B→(2), AB→(3), Schütteln→(4)

micro:bit AB2 Lösung

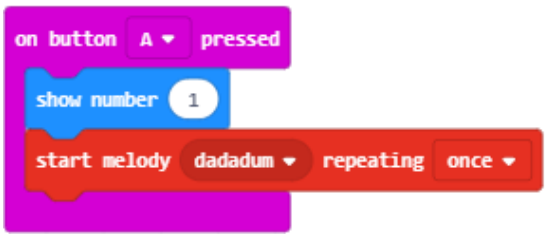
Achtung: Für die beschriebene Aufgabenstellung sind insgesamt vier Teilprogramme erforderlich. Alle vier Teilprogramme können gemeinsam im Arbeitsbereich abgelegt werden und als eine HEX-Datei heruntergeladen und am micro:bit gespeichert werden!



Beachte: Der **micro:bit v1** hat keine **Lautsprecher** eingebaut. Für die Ausgabe von Tönen, muss daher am micro:bit zusätzlich ein Lautsprecher angeschlossen werden. Im Simulationsbereich wird angezeigt, wie ein Lautsprecher über **Krokodilklemmen** und ein Standard **Audiokabe**l an den micro:bit angeschlossen werden muss.

Der neue **micro:bit v2** hat bereits einen Lautsprecher auf der Platine integriert, der Anschluss eines externen Lautsprechers ist daher nicht mehr erforderlich!

Die erforderlichen **Programmierblöcke** befinden sich in den **Registern** Basic, Input und Music.



Krokodilklemme

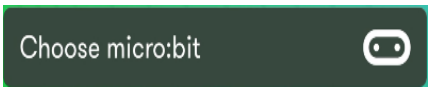
ACTIONcards for micro:bit Erste Schritte

Weitere **Übungsaufgaben:**
zu finden auf der folgenden **Website** →http://hemi.bplaced.net/Robotik/Roboter.htm

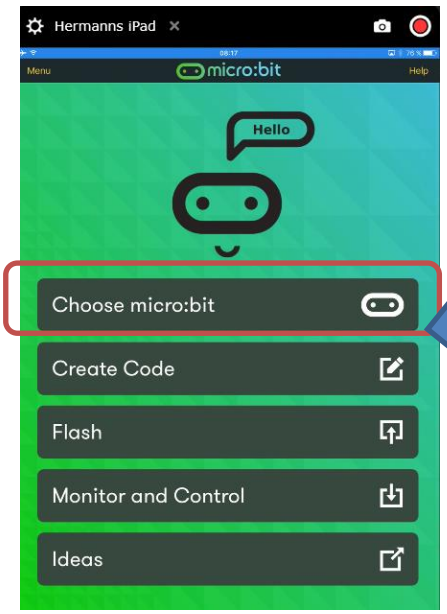
Drahtlose Datenübertragung mit der micro:bit APP

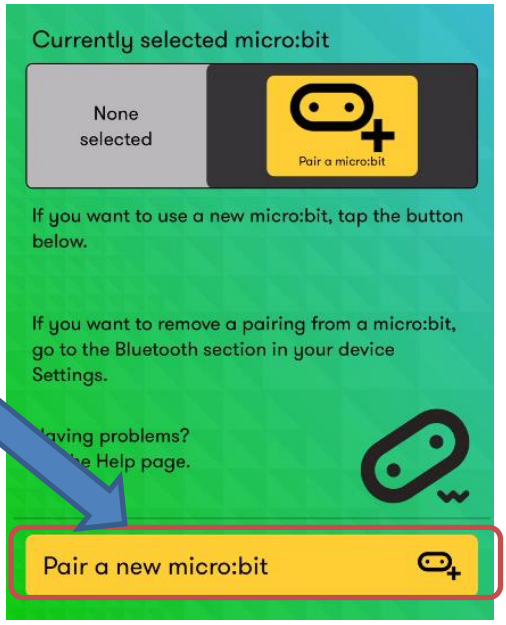
Mit der für IOS, Android und Windows10 (beta) verfügbaren **micro:bit APP**, kannst du Programme drahtlos über Bluetooth® an deinen micro:bit übermitteln.

Um den Bluetooth® micro:bit per Bluetooth® mit einem anderen Gerät (Tablet, Smartphone) nutzen zu können, muss es zunächst mit dem micro:bit gekoppelt werden.

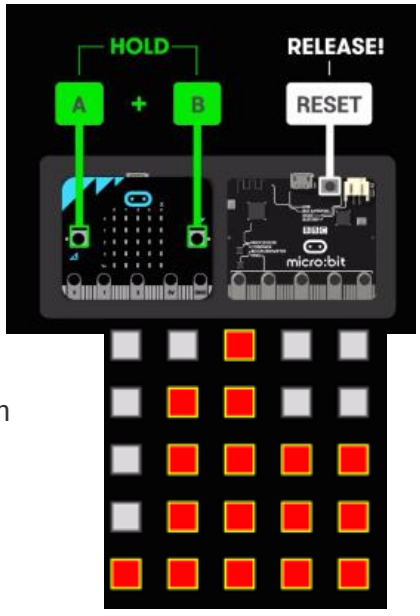
- ✓ micro:bit an die **Stromversorgung** (Micro-USB oder Battery-Pack) anschließen.
- ✓ **micro:bit APP** öffnen und die Schaltfläche 

auswählen!





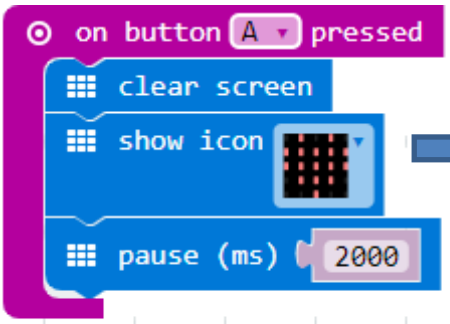
- ✓ Die beiden **Schalter an der Vorderseite des micro:bit gedrückt halten** und anschließend den **RESET-Knopf kurz drücken**, die beiden Schalter dürfen dabei nicht losgelassen werden! Nun sollte am Display des micro:bit der Schriftzug "**PAIRING MODE!**" als Laufschrift erscheinen. Nun erst dürfen die beiden Schalter losgelassen werden. Das am **micro:bit Display** angezeigte Muster am Notebook, Tablet ... eingeben und den weiteren Anweisungen am Bildschirm folgen! Nach erfolgreicher Koppelung erscheint am **micro:bit Display** ein **Häkchen**!



micro:bit APP - Beispielprogramme

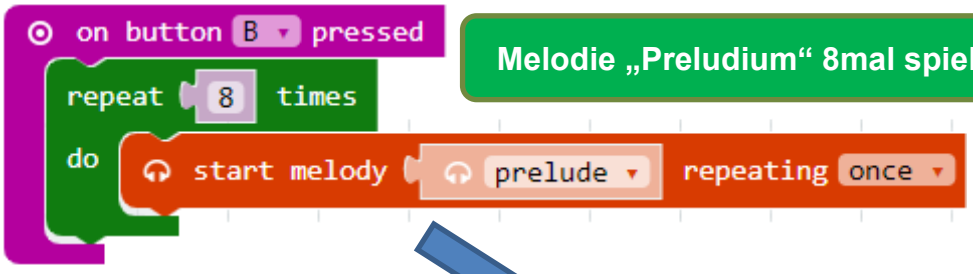
Verfügbar für **iOs** und **Android**

Herz-Icon am Display für 2 sec anzeigen

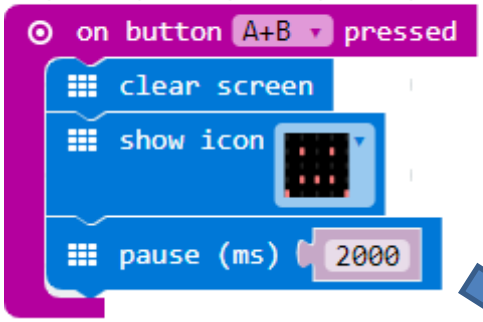



```
input.onButtonPressed(Button.A, () => {
    basic.clearScreen()
    basic.showIcon(IconNames.Heart)
    basic.pause(2000)
})
```

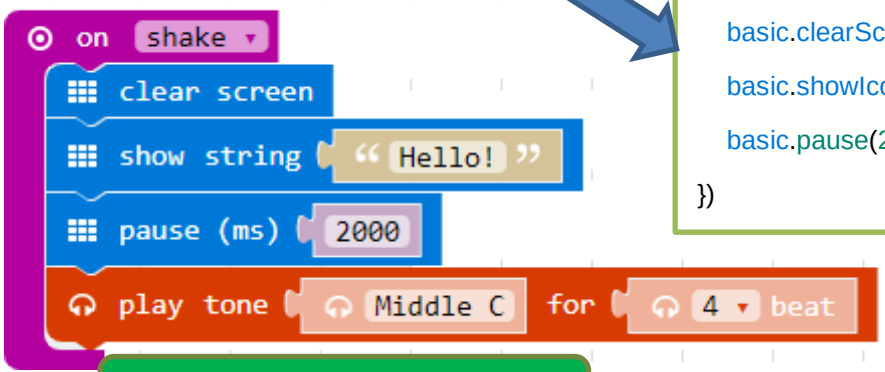
Melodie „Preludium“ 8mal spielen




```
input.onButtonPressed(Button.B, () => {
    for (let i = 0; i < 8; i++) {
        music.beginMelody(music.builtInMelody(Melodies.Prelude),
MelodyOptions.Once)
    }
```

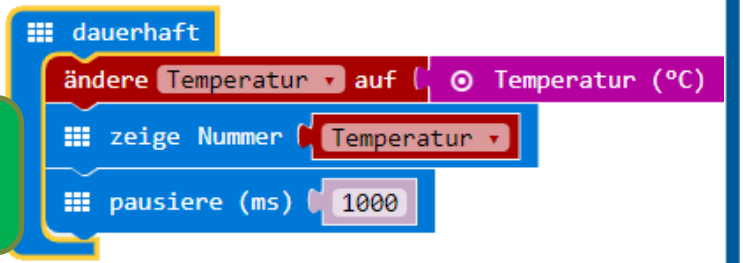



```
input.onButtonPressed(Button.AB, () => {
    basic.clearScreen()
    basic.showIcon(IconNames.Sad)
    basic.pause(2000)
})
```



Text „Hello“ als Laufschrift

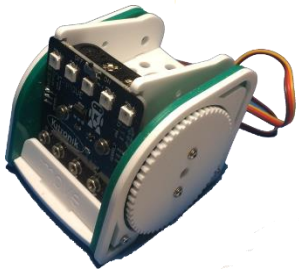
Thermometer mit digitaler Anzeige am Display des micro:bit



H. Milchram Oktober 2021

MOVE mini

https://www.kitronik.co.uk/5624-move-mini-buggy-kit-excl-microbit.html



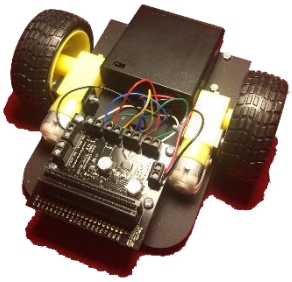
Der Kitronik „**MOVE mini buggy“** für den micro:bit ermöglicht einen lustvollen Einstieg in die Robotik. Der Buggy besitzt zwei Servomotoren für den Antrieb der Räder und 5 RGB LEDs für Sonderfunktionen oder Statusanzeigen. Außerdem bietet er die Möglichkeit einen Stift einzusetzen und er kann zusätzlich mit diversen Teilen (Add-On) erweitert werden.

- ✓ Bulldozer Add-On
- ✓ Tipper truck Add-On
- ✓ Bumper Add-On

Für den Zusammenbau des in Einzelteilen gelieferte „MOVE mini Buggy“ sind keine besonderen Kenntnisse oder Werkzeuge erforderlich! Der micro:bit-Controller ist nicht im Lieferumfang enthalten.

Line Following Buggy

https://www.kitronik.co.uk/blog/bbc-microbit-line-following-buggy/



Der „Line Following Buggy“ wird als Bausatz geliefert und eignet sich besonders für die Verbindung von Robotik und dem technischen Werkunterricht. Für den Zusammenbau sind ein Elektronik-Lötkolben und etwas handwerkliches Geschick erforderlich. Der für die Steuerung erforderliche micro:bit ist nicht im Bausatz enthalten und muss extra zugekauft werden.

Inventor‘s Kit

https://www.kitronik.co.uk/5603-inventors-kit-for-the-bbc-microbit.html

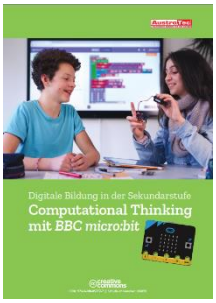
Das Inventor‘s Kit ist ein Bausatz, der ein lötfreies Experimentieren auf einem Breadboard ermöglicht. In diesem Set sind neben der Grundplatte, das Breadboard, eine Adapterplatine für den micro:bit, ein Anleitungsheft für 10 Versuche sowie viele Bauteile enthalten:
4 Drucktaster, 1 Motor, 1 Transistor, 9 LEDs, eine davon RGB, 15 Widerstände, 1 LDR, 1 Piezo-Summer, 1 Kondensator, 1 Potenziometer und diverse mechanische Bauteile

micro:bot PACK

https://www.techwillsaveus.com/shop/microbit-microbot-pack/
Das „micro:bit-microbot-pack enthält alle Teile, die benötigt werden, um drei unterschiedliche Roboter zu bauen.

micro:bit - Das Schulbuch

Das Buch stellt exemplarisch 20 Projekte vor, die in der Sekundarstufe I in einem fächerübergreifenden, projektorientierten Unterricht eingesetzt werden können. Es ist **kein** reines Informatik- oder "Programmierlernbuch". Es verwendet den BBC micro:bit als Grundlage für spannende und kreative Projekte. Der notwendige Code kann, wenn man einmal eine Idee für den Algorithmus gefunden hat, ganz einfach direkt im Webbrowser erstellt, getestet und danach bei Bedarf heruntergeladen und auf einem realen micro:bit installiert werden.



Quelle: https://microbit.eeducation.at/wiki/Hauptseite
Download-Link für das Buch: https://bit.ly/2PWsIVq

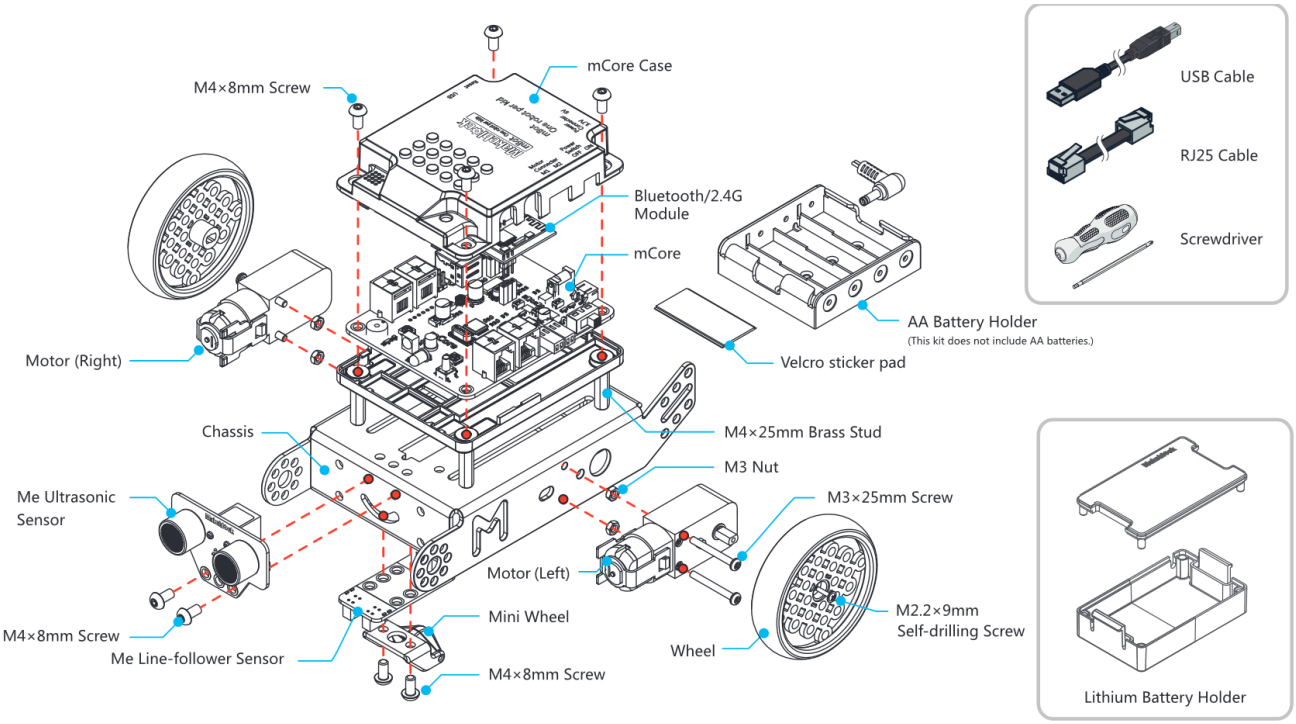
5. mBot-S Education Robot von Makeblock

Alter: 8+ (mBot Bluetooth Version bei ca. 95€
https://austro-tec.at/makeblock oder https://www.reichelt.at/)

Der **mBot** ist ein **Roboter-Bausatz**, der in zwei verschiedenen Versionen zur Verfügung steht:

- ✓ **Bluetooth-Variante** (Bluetooth 4.0)
  das 2,4G Modul kann nachgerüstet werden
- ✓ **2.4G-Variante** mit **USB-Dongle** (unterstützt Windows 32&64Bit Systeme)

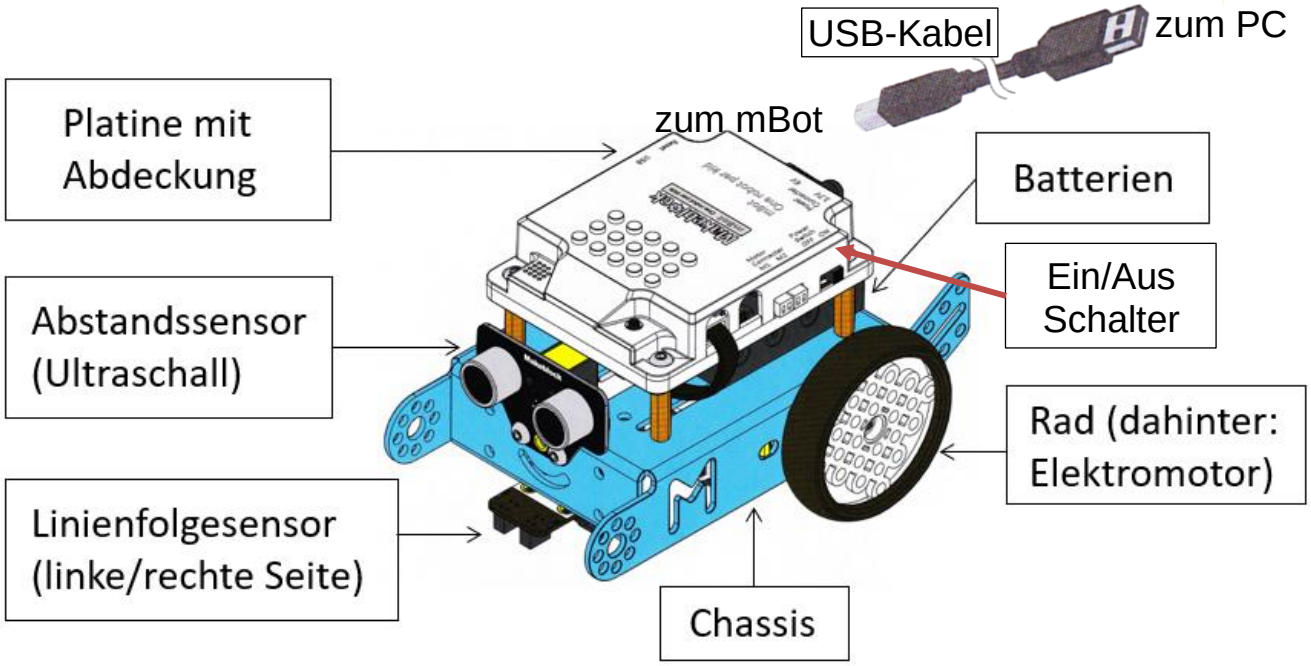
Bestandteile des Bausatzes



Die für die Steuerung verantwortliche **mCore-Platine** basiert auf dem **Arduino** Uno. Als **Programmiersprache** kommt eine an **Scratch** angelehnte, blockorientierte Programmier-Software (**mBlock**) zum Einsatz. Das Basis-Set ist wahlweise in einer Bluetooth- und einer 2.4G-Variante erhältlich. Die folgenden Funktionalitäten können programmiert werden:

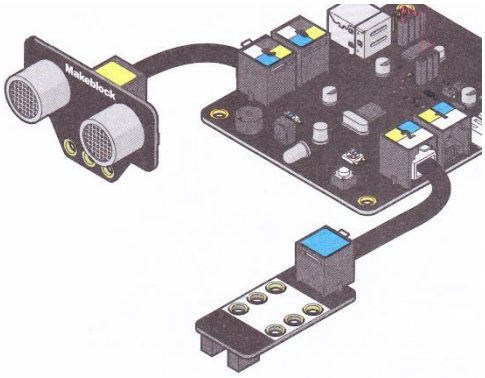
- ✓ **Bewegung** erfolgt über zwei Motoren (bis zu 4 Motoren können angeschlossen werden)
- ✓ **Audioausgabe**
- ✓ **Ultraschallsensor** (Erkennen von Hindernissen)
- ✓ **Linienfolgesensor**
- ✓ **Helligkeitssensor**
- ✓ **LED**-Lichter (RGB)
- ✓ **LED Matrix Display**
- ✓ 4 **RJ45 Ports** zum Anschließen der Motoren und von Hardware-Erweiterungen
- ✓ **1 Schalter** zur Auswahl dreier verschiedener Modi (Remote Steuerung, Hinderniserkennung, Linienverfolgung)
- ✓ **Lego Konnektoren**

Hauptbestandteile des mBots



Ports und Module
An den Ports (RJ25-Buchsen) sind standardmäßig drei Module angeschlossen.

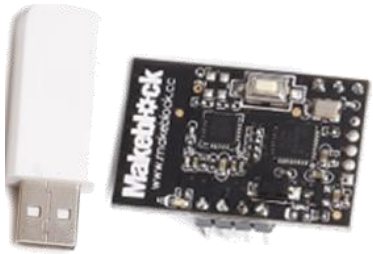
- ✓ Abstandsensor (Port 3)
- ✓ Linienverfolgungssensor (Port 2)
- ✓ LED-Matrix Modul (Port 1 oder Port 4)



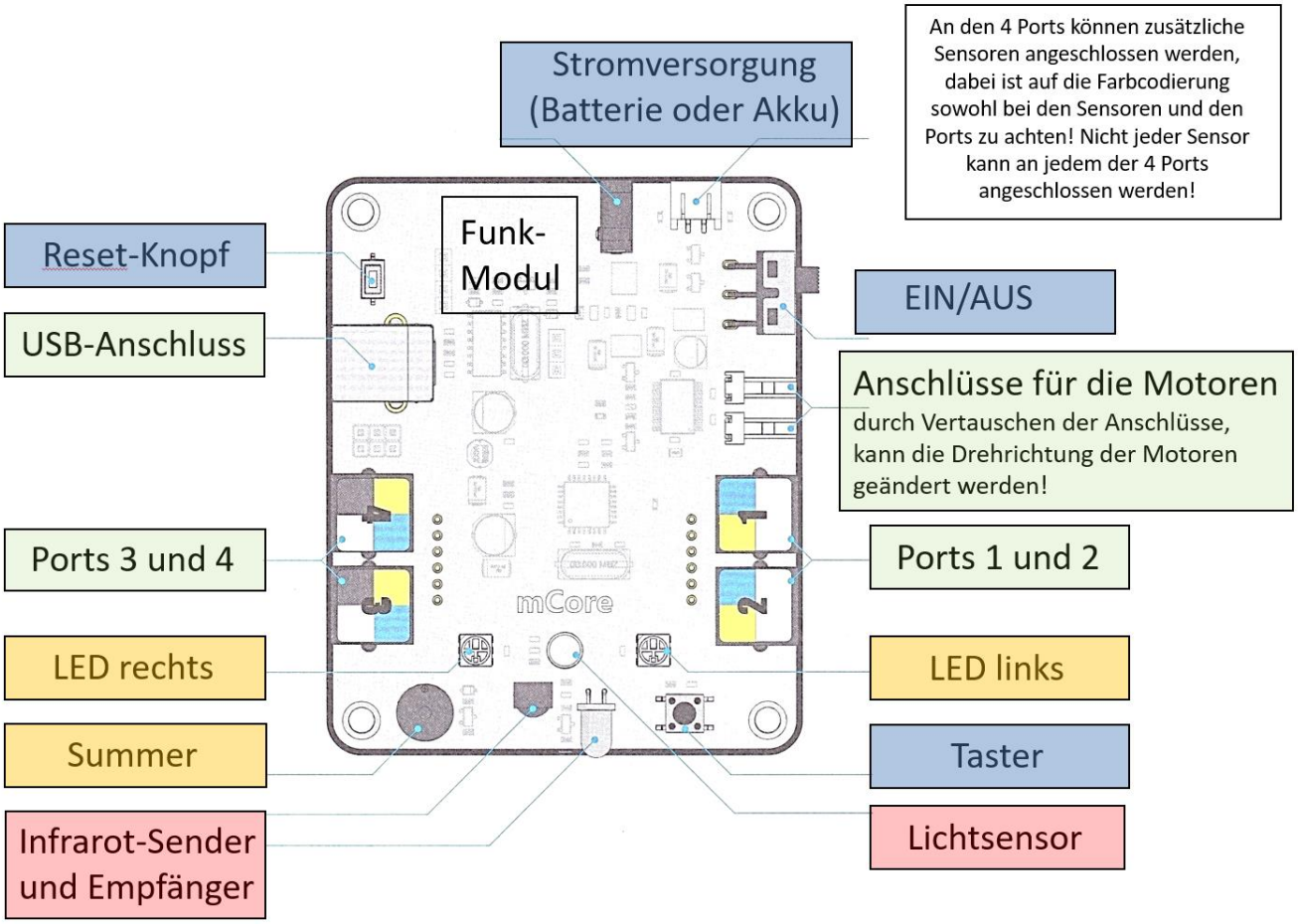
Nach dem Einschalten erklingen 3 Töne und die LEDs leuchten nacheinander in unterschiedlichen Farben (rot, grün, blau und weiß) auf. Die rote LED in der Mitte zeigt die Stromversorgung an. Die LED am **Bluetooth** bzw. **WLAN 2.4G-Modul** blinkt. Sobald eine Verbindung mit dem PC, Tablet hergestellt ist, leuchtet die LED konstant.

Verbindung mit dem PC oder Tablet

1. Verbindung mit dem **USB-Kabel** (nur PC)
2. Verbindung **Bluetooth**
   Notebooks und Tablet-PCs haben meist standardmäßig bereits Bluetooth integriert, für PCs ist meistens ein zusätzlicher Bluetooth-Dongle erforderlich!
   - ✓ Bluetooth am PC, Tablet … einschalten
   - ✓ Am Windows PC muss der mBot vor der ersten Verwendung über die **Einstellungs-APP** hinzugefügt werden!
3. Verbindung mit dem WLAN **2.4G-Modul**
   https://bit.ly/2Z8OYxY



Aufbau der Plantine

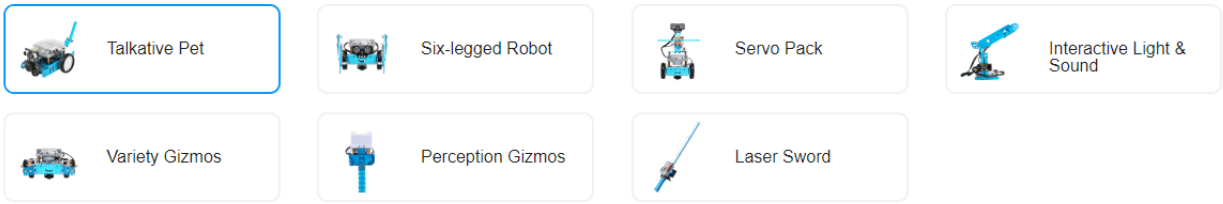


Ein paar Beispiele für Sensoren die über die 4 Ports angeschlossen werden können:

- ✓ Display
- ✓ Geräuschsensor
- ✓ Temperatur- und Luftfeuchtigkeitssensor
- ✓ Bewegungssensor
- ✓ Kompasssensor
- ✓ Schieberegler/Potentiometer

Die Farbe am Sensor gibt Auskunft darüber, an welchen Port er angeschlossen werden darf!

Zusätzlich gibt es eine Fülle von Erweiterungsbausätzen mit denen unterschiedliche Roboter gebaut werden können. https://www.makeblock.com/mbot-add-on-packs-talkative-pet



Erste Inbetriebnahme

Beim mBot handelt es sich um einen Roboter-Bausatz, der in einzeln verpackten Teilen geliefert wird. Geleitet durch eine Kurzanleitung montiert man mit Hilfe des mitgelieferte Schraubenziehers, die beiden Motoren, Sensoren und Räder an das ALU-Chassis und schließt sie laut Anleitung an das mCore-Board an.

Der mBot kann wahlweise mit vier Stück **AA-Batterien** oder einem **Akku** betrieben werden. Beides ist jedoch nicht im Lieferumfang enthalten und muss extra gekauft werden.

Nach dem Zusammenbau kann der mBot auf Grund der **drei einprogrammierten Funktionsmodi** sofort einige grundlegende Funktionen ausführen. Die 3 Modi können direkt am mBot über einen kleinen schwarzen Taster an der Oberseite ausgewählt oder über die Fernbedienung (Knöpfe A B C) gestartet werden. (Firmware der Fabrik muss eingespielt sein siehe → **Aktualisierung der Firmware**!)

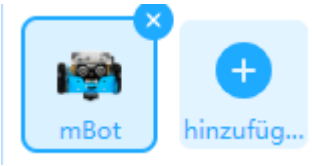
- ✓ **Remote Control** (Fernbedienung → **A**, LED-Farbe „weiß“) Steuerung über die mitgelieferte Fernbedienung oder mit Hilfe der Makeblock APP für Android und iOS-Geräte

- ✓ **Hinderniserkennung** (Fernbedienung → **B**, LED-Farbe grün) gesteuert durch die eingebauten Abstandssensoren kann der mBot selbständig die Umgebung erkunden und dabei Hindernissen ausweichen

- ✓ **Linienverfolgung** (Fernbedienung → **C**, LED-Farbe blau)

Achtung: Vor der ersten Verwendung ist es sinnvoll, eventuell vorhandene Firmware-Updates zu installieren.

- ✓ mBot mit dem mitgelieferten **USB-Kabel** an den PC anschließen
- ✓ Programm **mBlock** am PC öffnen
- ✓ **Register Geräte** auswählen und aus der Bibliothek **mBot** auswählen und als „meistgenutztes Gerät“ hinzufügen (Cody kann entfernt werden)



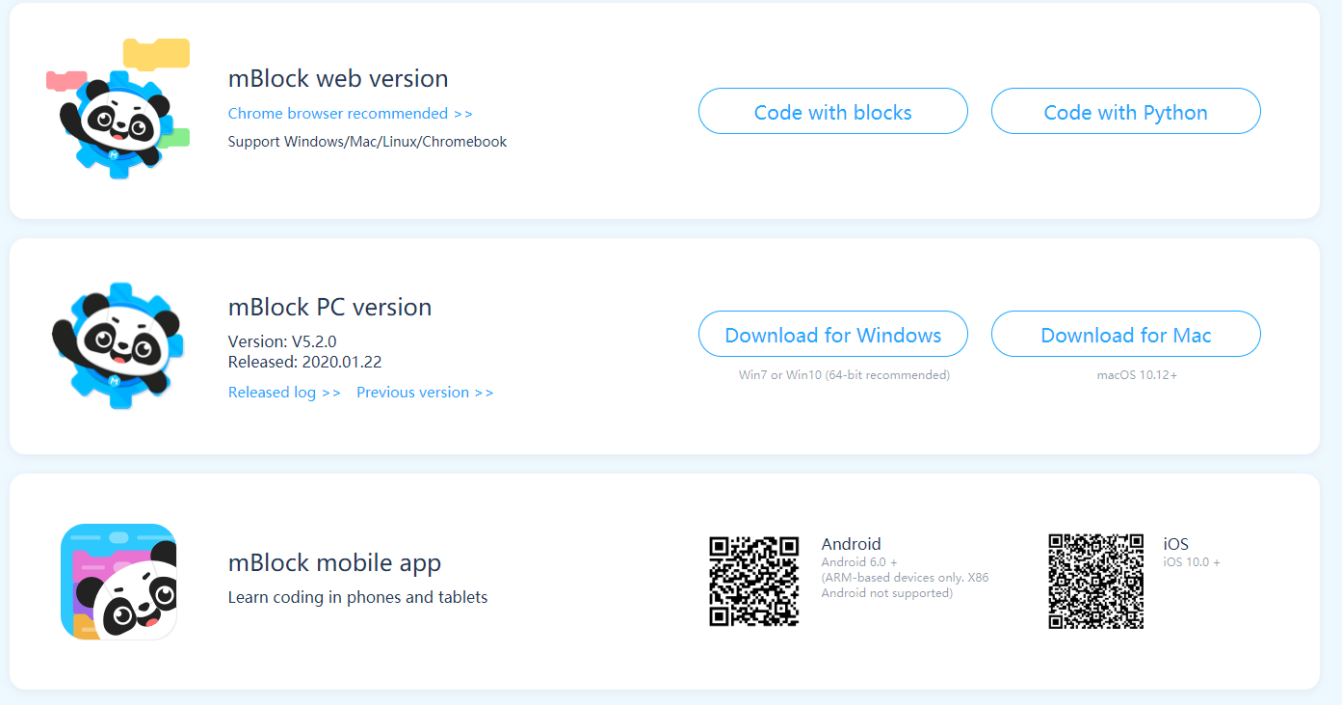
Aktualisierung der Firmware

Für die **Aktualisierung der Firmware** stehen zwei Varianten zur Verfügung:

- ✓ **Online Firmware**
- ✓ **Firmware der Fabrik**
  Nur hier stehen drei einprogrammierte Funktionsmodi (**Remote Control**, **Hinderniserkennung** und **Linienverfolgung**) zur Verfügung!

Programmierung

Es stehen grundsätzliche 3 Arten von Programmiersoftware für den mBot zur Verfügung:
https://mblock.makeblock.com/en-us/download/



Die **Makeblock APP** und die **mBlock Blockly APP** bieten eine einfache Benutzeroberfläche, die es den Kindern ermöglicht, ihren mBot per Knopfdruck zu steuern oder die voreingestellten Steuerungseinstellungen beliebig zu kombinieren. Durch einfaches Ziehen, Ablegen und Kombinieren von Befehlsblöcken, genau wie bei Bausteinen, können Kinder jede Bewegung des mBots steuern und ihren eigenen persönlichen Roboter erstellen.

Über die **Makeblock-APP** stehen mehrere Lernpfade zur Erkundung des mBots zur Verfügung:

- ✓ **Spielen** (Fahren, Zeichnen und Laufen, Musiker, Sprachsteuerung)
- ✓ **Erstellen** (eigene interaktive Bedienoberfläche gestalten)
- ✓ **Coding (**vorgegebene Lernpfade und freies Programmieren mit der mBlock Blockly APP**)**

Coding am PC mit Scratch

Lokal am PC

- ✓ Aktuelle Software herunterladen und installieren
  https://mblock.makeblock.com/en-us/download/

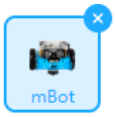
Online im Browser (mLink)

- ✓ **mLink installieren**
  https://mblock.makeblock.com/en-us/download/

  →**Other mBlock software** (es stehen Versionen für verschiedene Betriebsysteme zur Verfügung (Windows, Mac, Linux, Chromebook)

mBot – Programmierung mit Scratch

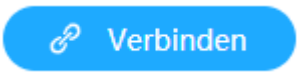
1. **mBlock** kann in der **Offline-Version** (PC-Version) oder über **mLink** als **Online-Version** (Internetverbindung erforderlich) gestartet werden.

2. In der Register-Karte „Geräte“ den mBot 

auswählen

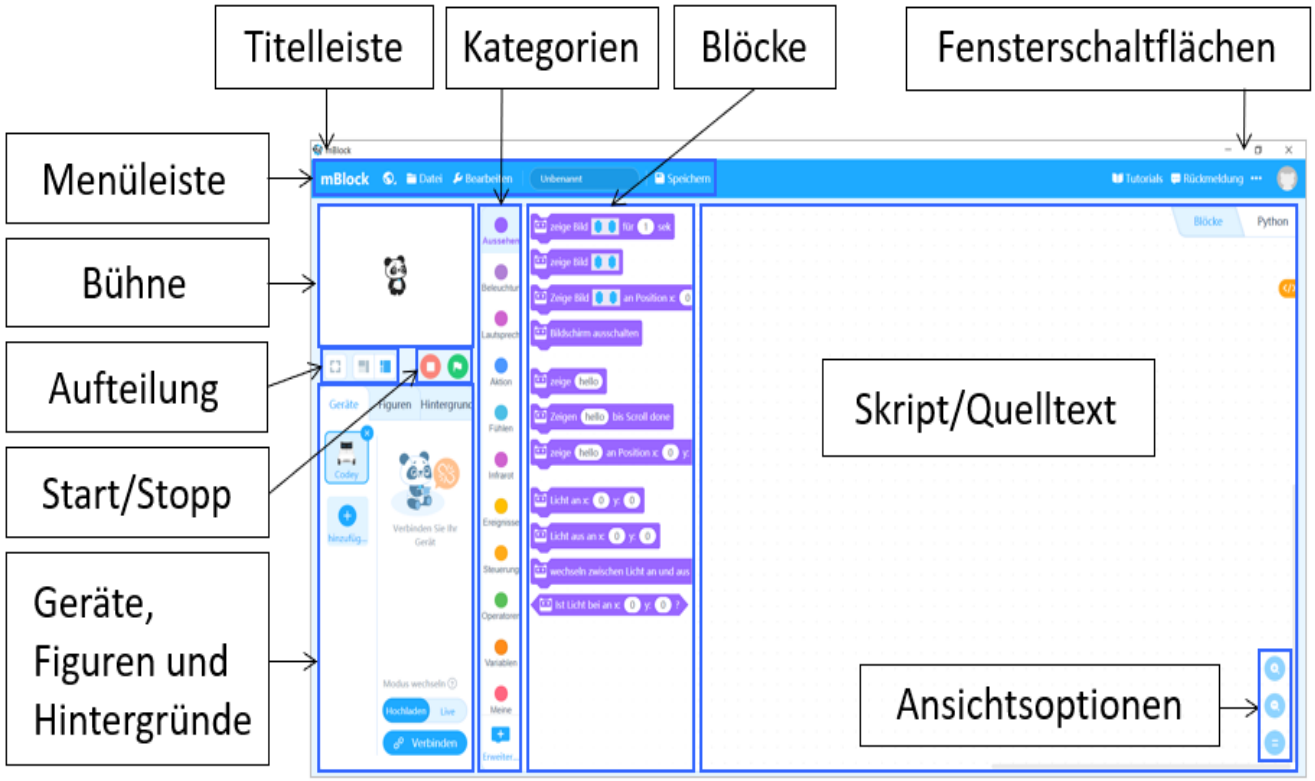
3. Modus für die Programmausführung auswählen 

(bei Bluetooth Verbindung steht nur der Live-Modus zur Verfügung)

4. Verbindung mit dem mBot herstellen 

(USB, **Bluetooth**, 2,4G)
Jeder mBot ist eindeutig über seine **MAC-Adresse** (14 stellige Buchstaben-Zahlenkombination erkennbar!)
Sobald mehrere mBots verwendet werden ist es sinnvoll, diese jeweils mit der **MAC-Adresse** zu beschriften um eine schnelle Zuordnung zu ermöglichen!
Im Klassenverband ist die Verwendung von **2,4G Dongles** zu überlegen, da jeder mBot eindeutig einem WLan-Dongle zugeordnet ist, mit dem ers sich automatisch verbindet!

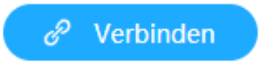
5. Gewünschte Programmblöcke per Drag&Drop ins Skriptfenster ziehen



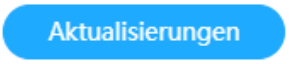
Die Programmierung mit mBlock basiert auf der visuellen, **objektorientierten** Programmiersprache **Scratch**.

Grundfunktionen und Blöcke

Vor der Programmierung wird empfohlen die vorinstallierten Programme zu löschen:

- ✓ Verbindung zum mBot mit **USB-Kabel** herstellen
- ✓ Auf 

klicken und USB auswählen
- ✓ 

(mBot muss eingeschaltet sein!)
- ✓ Online Firmware auswählen und auf 

klicken!

Steuerung der Motoren (Register Aktionen) 

Im Register „Aktionen“ stehen sieben Blöcke zur Steuerung der Elektromotoren bereit.

- ✓ Der Prozentsatz (0% - 100%) bestimmt die Leistung (**Drehgeschwindigkeit**) der Motoren
- ✓ Die ersten 4 Blöcke ermöglichen eine zeitliche Begrenzung der Bewegung
- ✓ Der letzte Block dient zum Unterbrechen der Bewegung beim Verwenden von Blöcken ohne zeitlicher Begrenzung oder beim Erkennen eines Hindernisses

Dein mBot soll beim Anklicken des Fähnchens  folgende Aufgaben erledigen:

(1) mit 50% der Leistung für 3 Sekunden vorwärts fahren.
(2) mit 30% der Leistung für 5 Sekunde rückwärts fahren.
(3) mit 90% der Leistung für 1 Sekunden nach links drehen.
(4) mit 75% der Leistung für 1 Sekunde vorwärts fahren dann 2 Sekunden warten und mit 100% der Leistung für 1 Sekunde rückwärts fahren.

Tipp: Um ein Programm mit dem Fähnchen starten zu können, muss das Ereignis 

am Beginn eines Skripts stehen! Probiere auch andere Ereignisse zum Starten aus

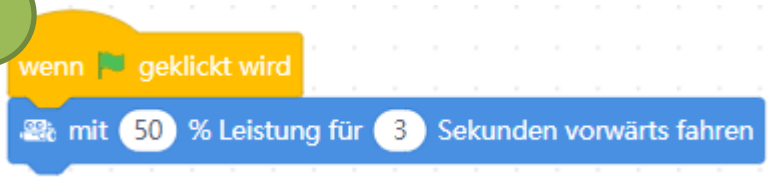
Der Befehl fürs Warten befindet sich im Register 

→ 

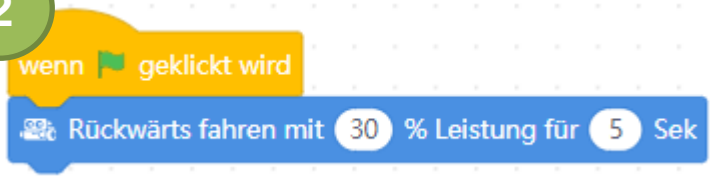
Versuche die Skripts sowohl im **Live-Modus** als auch durch **Hochladen** auszführen!

mBot AB1 Lösung

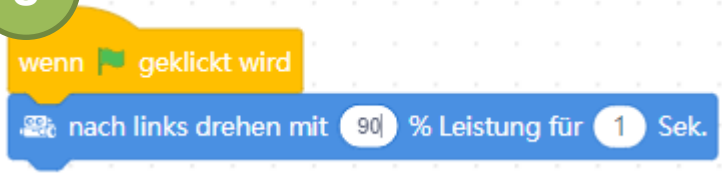
1



2



3

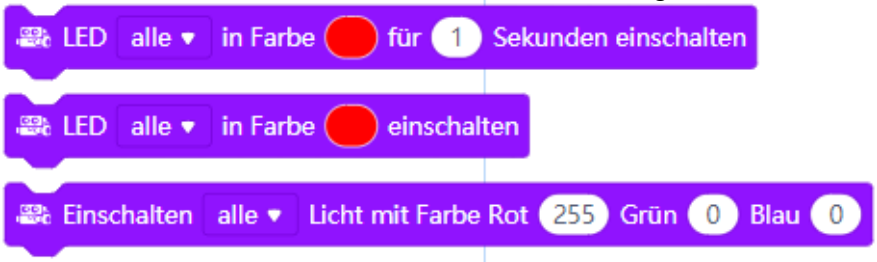


4

Kommunikation mit der Umwelt →Aussehen: RGB LEDs)

Register Zeigen OnBoard-LEDs:

Im vorderen Bereich der Platine sitzen zwei **RGB-LEDs**. Die beiden LEDs können einzeln angesteuert werden. Die Farbeinstellung erfolgt dabei entweder über **Schieberegler** oder durch die Eingabe der sogenannten **RGB-Werte** (Zahlen von 0-255. → siehe InO-Bot: additive Farbmischung →. Bei den beiden Blöcken ohne zeitliche Begrenzung, müssen die LEDs durch einen Block mit den Werten 0-0-0 ausgeschaltet werden.



(1) beide LEDs leuchten 2 Sekunden rot
(2) die beiden LEDs leuchten in unterschiedlichen Farben
(3) beide LEDs sollen zuerst 1 Sekunde gelb leuchten, dann für 2 Sekunden ausgeschalten werden und dann für drei Sekunden blau leuchten.
(4) die beiden LEDs sollen abwechselnd in 2 unterschiedlichen Farben 5x blinken
Tipp: Verwende dazu den Block „wiederhole Anzahl“ → Register Ereignisse
(5) Programmiere eine Verkehrsampel
https://www.wien.gv.at/verkehr/ampeln/ampelkunde.html

mBot AB2 Lösung

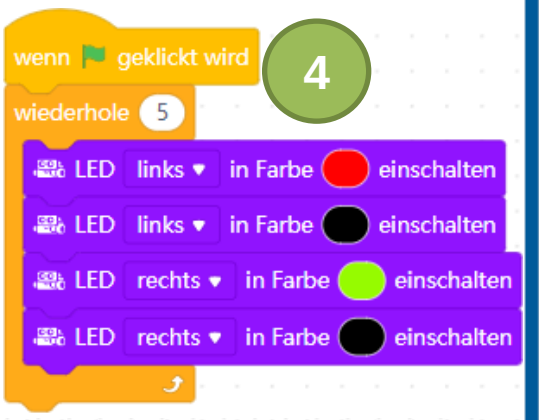
Wenn du hier mit der linken Maustaste anklickst kommst du zu den Schiebereglern zum Einstellen der Farbwerte!

1

2

3

4



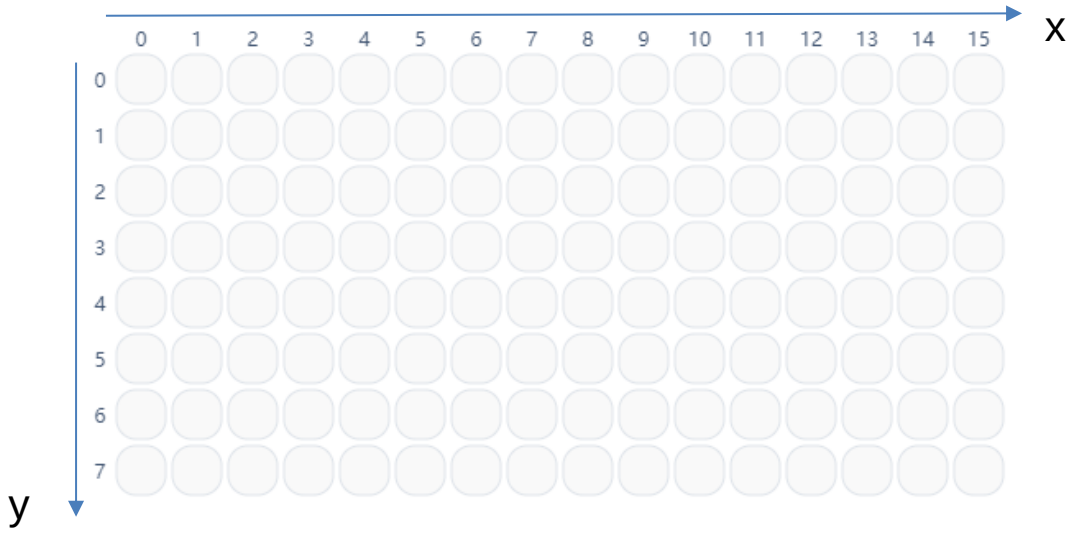
5

H. Milchram Oktober 2021

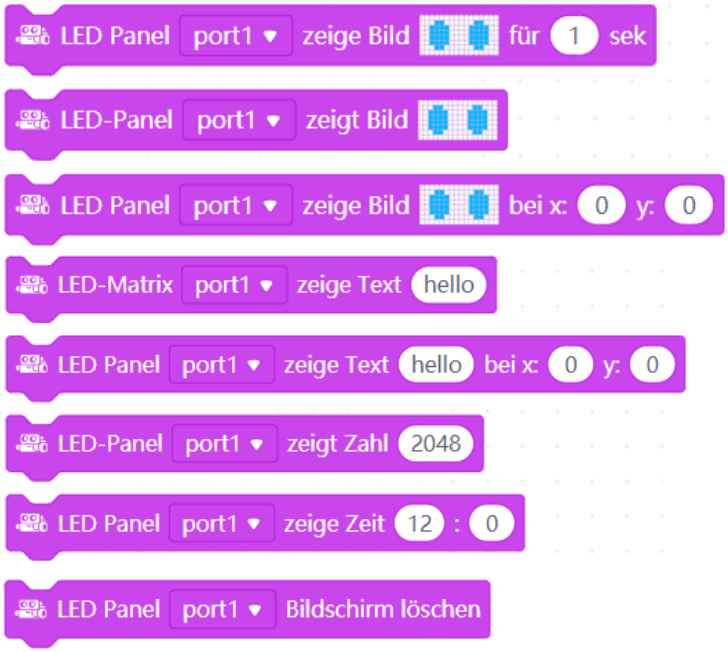
Kommunikation mit der Umwelt →Anzeigen: LED Matrix Display)

✓ LED-Matrix-Display (Register Aussehen)

Das LED-Matrix-Display kann an den freien Ports 1 oder 4 angeschlossen werden. Sie besteht aus einer Matrix von 16x8 Pixel (x=0-15, y=0-7, 128 LEDs) die einzeln angesteuert werden können. Es können damit sowohl kurze Texte, Zahlen oder Pixelbilder angezeigt werden.

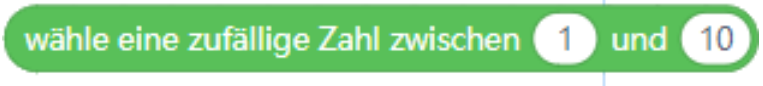


In der Kategorie „Aussehen“ gibt es hierfür acht Blöcke.



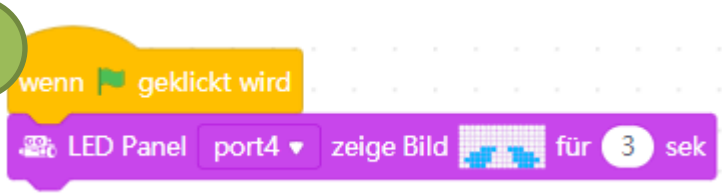
(1) Zeige für 3 Sekunden dein Lieblingssymbol aus der vorhandenen Auswahl an
(2) Zeichne dein eigens ICON und zeige es für 5 Sekunden am Display
(3) Lasse einen Punkt in der Mitte des Display von oben nach unten wandern
(4) Lasse für jeweils 0.2 Sekunden einen Punkt zufällig am Display aufleuchten

Verwende dazu „Zufallszahl“ aus dem Register Operatoren

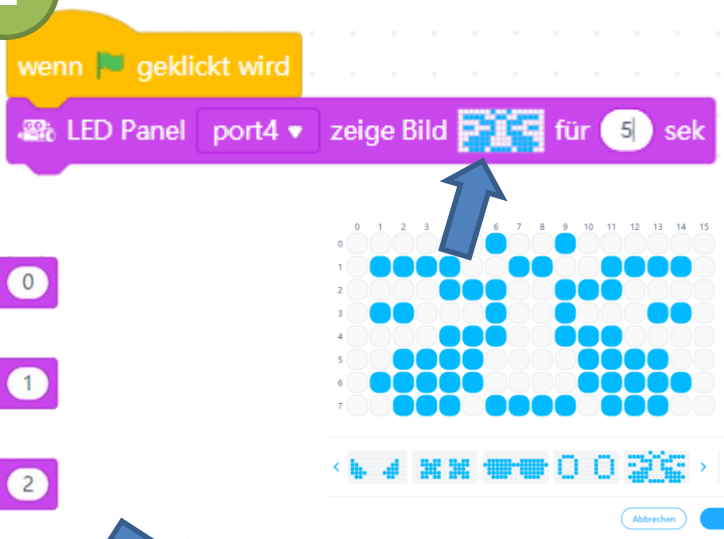


mBot AB3 Lösung

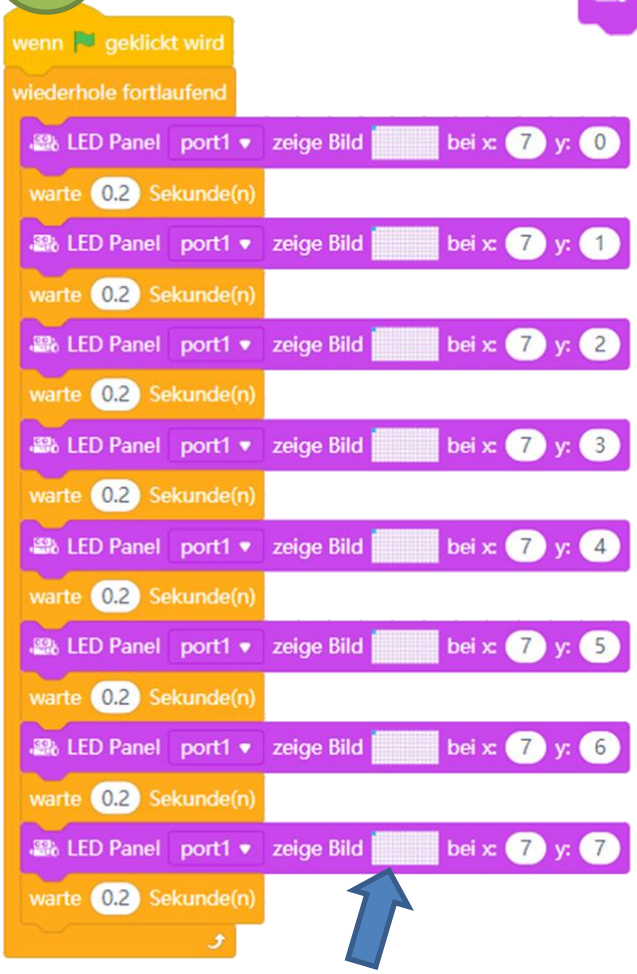
1

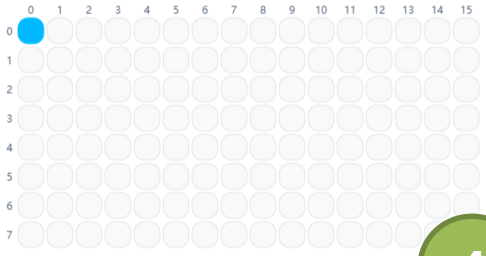


2



3





3a So geht es auch!!

Am virtuellen Display wird 1 Punkt im Koordinatenursprung (0/0) gezeichnet!

4

H. Milchram Oktober 2021

Kommunikation mit der Umwelt →Audio-Signale

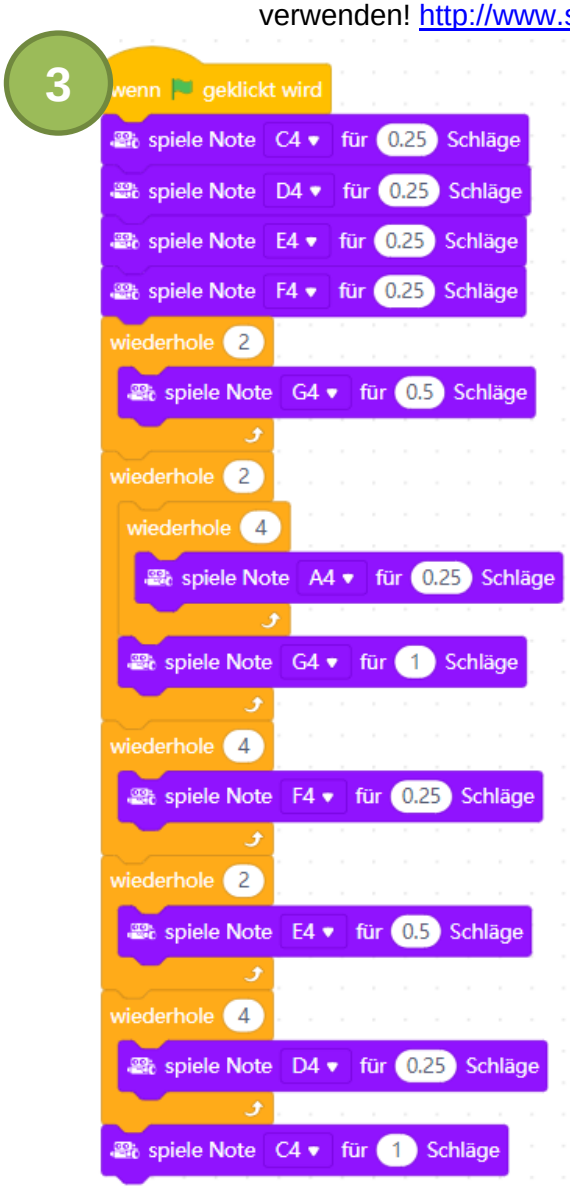
Die SOUNDs befinden sich im Register 

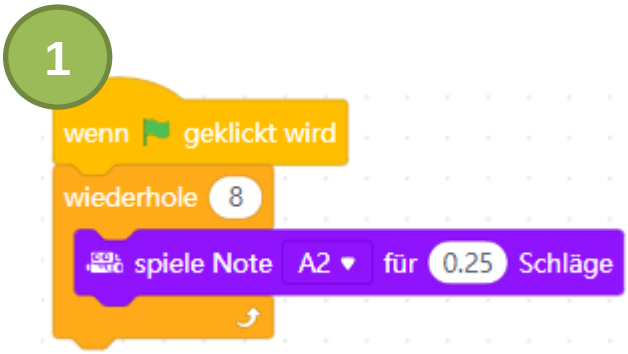
Der **Summer/Buzzer** befindet sich auf der Platine deines mBot und kann Töne erzeugen und über den eingebauten Lautsprecher wiedergeben!

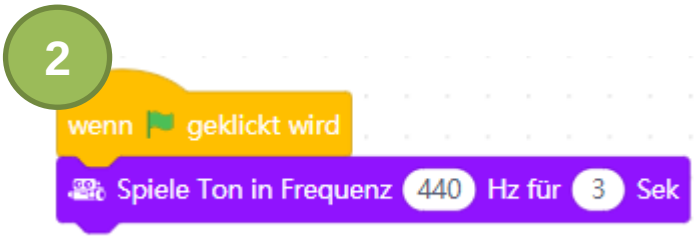
Die Tonhöhe wird dabei über die Frequenz (Anzahl der Schwingungen pro Sekunde) oder über die Notennamen und die Oktavbezeichnung angegeben. Die Tonlänge kann in Sekunden bzw. Schlägen (ganze Note = 1, halbe Note = 0.5, Viertelnote = 0.25, Achtelnote = 0.125) angegeben werden. **Achtung**: Die Bezeichnungen in mBlock stimmen nicht mit den in der Musiklehre verwendeten Oktavbezeichnungen überein! Kammerton $\mathbf{a^1}$ →mBlock **A4**

(1) Dein mBot soll 8 Viertelnoten C2 spielen.
(2) Erzeuge den **Kammerton**, er soll für 3 Sekunden erklingen.
(3) Versuche die Melodie von „Alle meine Entchen zu spielen“ (Beginne mit C4). Zum Schreiben von Noten kannst du den Online-Noteneditor „Scorio“ verwenden! http://www.scorio.com/de_DE/web/scorio/new-score

mBot AB4 Lösung







Kommunikation mit der Umwelt → Helligkeitssensor

Der Helligkeitssensor befindet sich vorne auf der Platine.

Der zugehörige Block ist im Register 

zu finden.

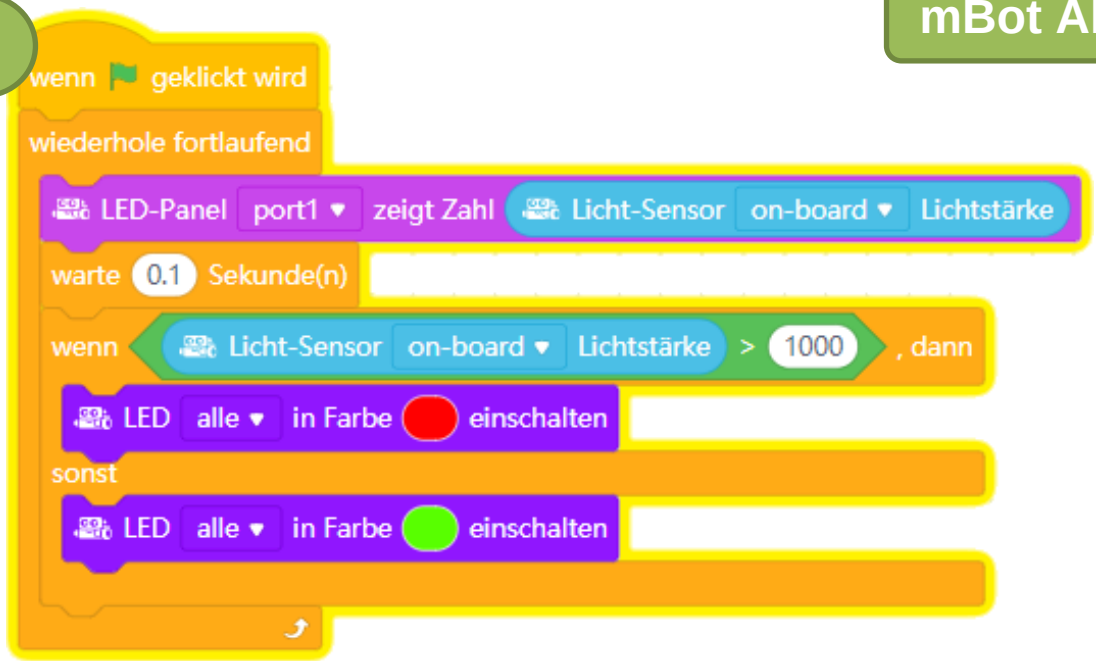
Wertebereich: 0 (ganz dunkel) bis ca. 1030 (sehr hell)

Mit folgendem Skript wird der aktuelle Wert der Helligkeit abgefragt und auf dem LED-Matrix-Modul angezeigt:

(1) Setze den Sensor unterschiedlichen Helligkeiten (abdecken; aufdecken) aus und beobachte die angezeigten Werte. Abhängig von der Helligkeit, sollen die LEDs unterschiedlich leuchten.

mBot AB5 Lösung

1



H. Milchram Oktober 2021

Kommunikation mit der Umwelt →Ultraschallsensor

Der Ultraschallsensor (Abstandssensor) befindet sich vorne am Roboter und sollte mit dem Port 3 verbunden sein.

Der zugehörige Block ist im Register Fühlen zu finden.

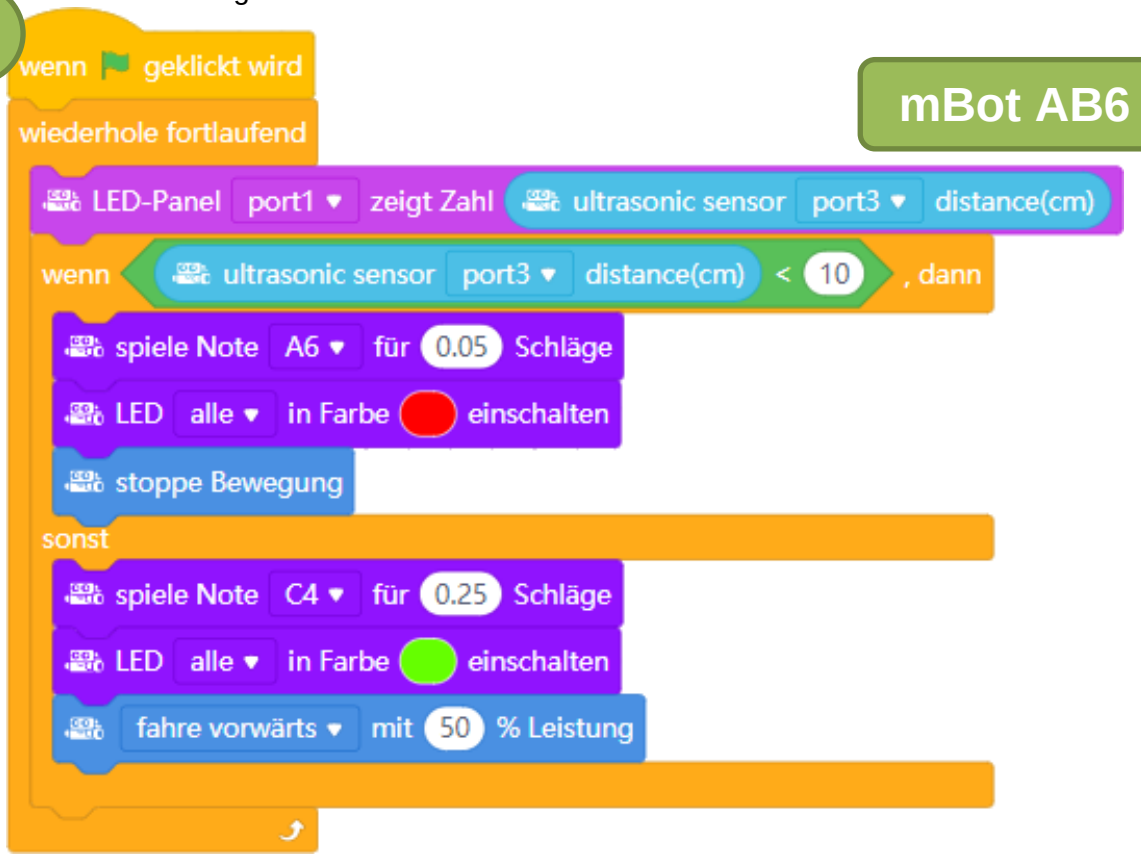
Die Entfernungen werden in cm angegeben

✓ Mit dem folgendem Skript, wird die Entfernung am LED-Matrix-Display anzeigt.

(1) Schreibe ein Programm das die jeweilige Entfernung auf dem LED-Matrix-Display anzeigt und abhängig von der Entfernung einen hohen Ton (Abstand < 10) bzw. einen tiefen Ton (Abstand > 10) ausgibt. Solange kein Hindernis in der Nähe ist, soll dein mBot vorwärts fahren. Taucht ein Hindernis auf, soll er bei einem Abstand <10 stoppen und die LEDs sollen rot leuchten, während der Vorwärtsfahrt sollen die LEDs grün leuchten.

1

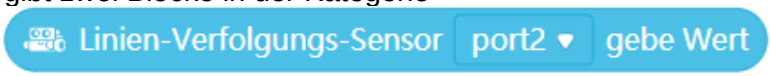
mBot AB6 Lösung



Kommunikation mit der Umwelt →Linienverfolgungssensor

Der Linienverfolgesensor befindet sich vorne an der Unterseite deines mBots und sollte an Port 2 angeschlossen sein. Der Linienverfolgungssensor besteht aus zwei Sensoren (links/rechts), die unabhängig voneinander überprüfen, ob der Untergrund hell oder dunkel ist.

Es gibt zwei Blöcke in der Kategorie Fühlen

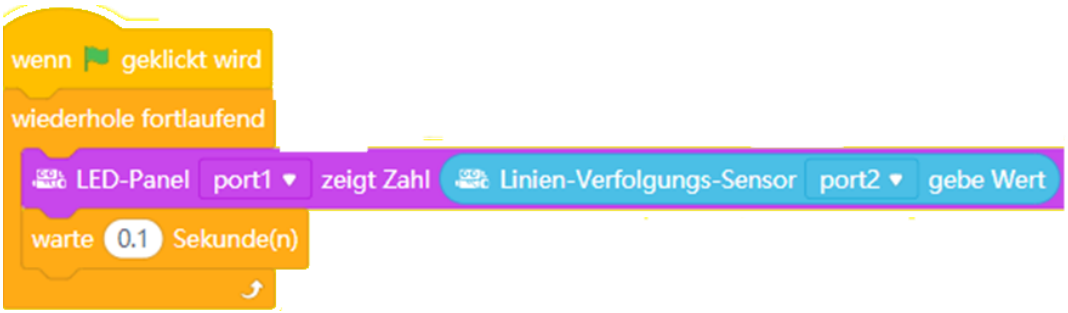


Rückgabewerte:

- ✓ 0 li: schwarz re: schwarz →Roboter ist in der Spur
- ✓ 1 li: schwarz re: weiß → Roboter zu weit rechts
- ✓ 2 li: weiß re: schwarz→ Roboter zu weit links
- ✓ 3 li: weiß re: weiß →Roboter ist neben der Spur

Die **Entfernungen** werden in cm angegeben

Mit dem folgendem Skript werden die **Sensorwerte am LED-Matrix-Display** anzeigt.



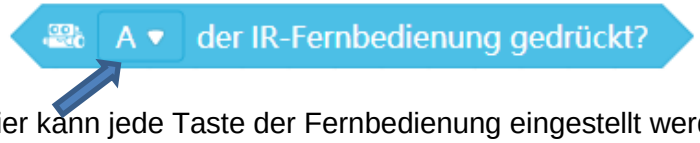
(1) Schreibe ein Programm das deinen mBot auf einer schwarzen Linie dahinfahren lässt.

mBot AB7 Lösung

Steuerung mit der Fernbedienung

Der Infrarot-Sender befindet sich in der Fernbedienung, der Empfänger auf der Platine.

Der zugehörige Block ist unter Fühlen zu finden.



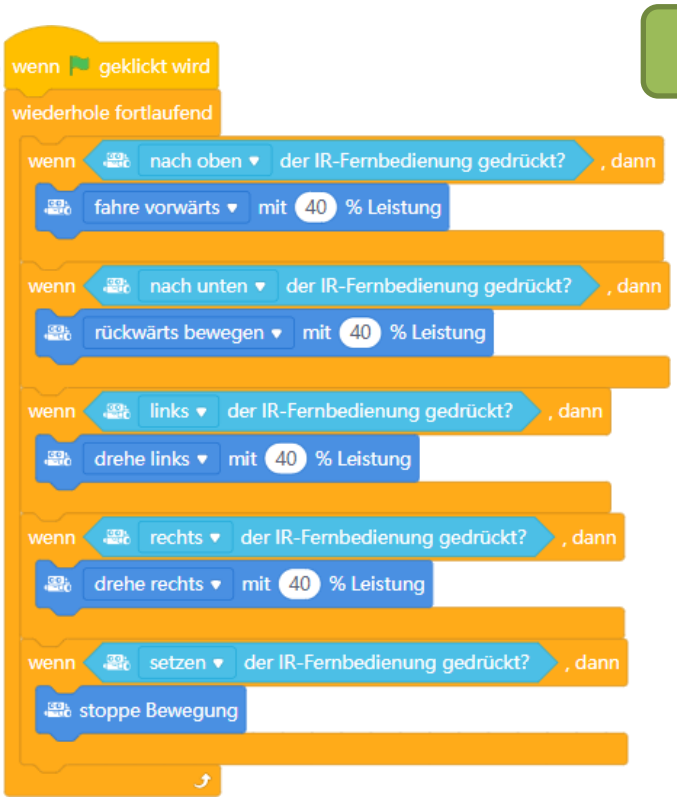
Hier kann jede Taste der Fernbedienung eingestellt werden.

✓ Teste die Fernbedienung mit folgendem Skript. Das LED-Matrix-Display soll „A“ zeigen, wenn die Taste A gedrückt wird, sonst soll das Display leer bleiben.

(1) Schreibe ein Programm, mit dem du deinen Roboter über die Fernbedienung steuern kannst.

mBot AB8 Lösung

1

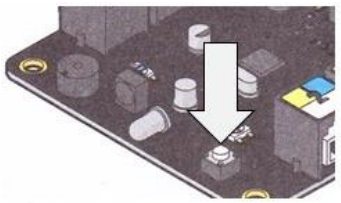


Achtung: Um den mBot über die Fernbedienung steuern zu können, muss die Firmware der Fabrik(06.01.009) eingespielt sein!

Taster

Auf dem mBot befindet sich ein Taster, mit dem im Auslieferungszustand die drei verschiedenen Modi (Remote-Control, Linienverfolgung und Hinderniserkennung) eingestellt werden können.

Der zugehörige Block ist unter Fühlen zu finden.



(1) Teste den Taster mit folgendem Skript. Beim Drücken des Tasters (OnBoard-Taste) soll auf dem LED-Matrix-Modul „ups“ zu lesen sein. Wird der Taster losgelassen, soll das Display leer bleiben.

mBot AB9 Lösung

1

Achtung: Um den mBot über den Taster steuern zu können, muss die Firmware der Fabrik(06.01.009) eingespielt sein!

Linksammlung

Allgemein

https://www.inf-schule.de

https://www.inf-schule.de/programmierung/robotik/lejoseinstieg/planundstruktur

https://www.techtools21.ch/de/wissen/anwendung-ino-bot

https://hos.se/files/custom/ProductTemplate/ino-bot-user-guide---pc.pdf

OzoBot

https://ozobot.com/support/manuals

https://learninglab.tugraz.at/informatischegrundbildung/oer-schulbuch/ozobot-unterrichtsbeispiele/

InO-Bot

https://www.tts-group.co.uk/ino-bot-scratch-programmable-bluetooth-floor-robot/1009821.html

BlueStacks und **NOX**
sind Android-Emulatoren für Windows PCs zum Ausführen der **InO-Bot APP**
https://www.bluestacks.com/download.html?utm_campaign=homepage-dl-button-en

https://www.bignox.com/en/download/fullPackage

Thymio

https://www.thymio.org/home-de:home

https://www.generationrobots.com/blog/de/programmierung-roboter-thymio/

Micro:bit

https://microbit.org/de

http://robotik1.menu.baa.at/ (Coding & Roboting Alois Bachinger PH der Diözese Linz)

https://play.google.com/store/apps/details?id=com.kitronik.blemove

Inhalt Robotik-Koffer

3 Ozobot evo

12 Stifte (rot, grün, blau, schwarz) für Ozobot Farbcodierung

3 mBots

3 Wireless-Dongle (USB)
3 Fernbedienungen

3 Thymio

3 Fernbedienungen
3 Wireless-Dongle (USB)
3 kurze Micro USB-Kabel weiß (10cm)

3 micro:bit incl. Plastik Schutzhüllen

3 Batterie-Packs leer + 6 Batterien AAA
3 Micro USB-Kabel schwarz mit Klettbänder
10 Krokodilklemmen

1 iPad

1 USB Ladegerät für iPad
10fach USB-Ladestation fix im Koffer verbaut
1 Stromkabel für USB-Ladestation

1 Skriptum „Hilfe die Roboter kommen“
1 Mappe mit Arbeitsblättern (Kopiervorlagen!)

Weitere Informationen und Materialien:

http://hemi.bplaced.net/Robotik/Roboter.htm