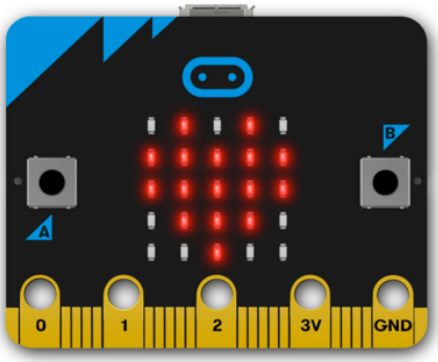
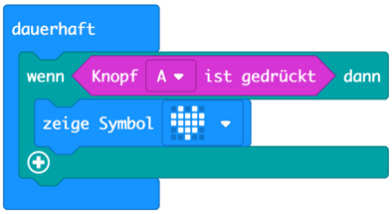


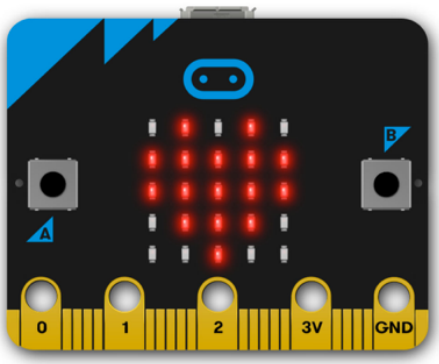


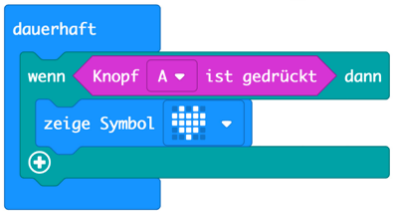
DigiMe Aufbau Workshop

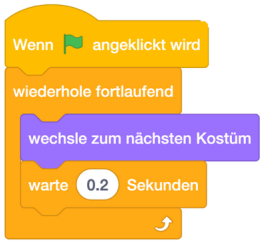
Interreg
EUROPÄISCHE UNION
Österreich-Tschechische Republik
DigiMe
Europäischer Fonds für regionale Entwicklung

Project co-funded by the European Regional Development Fund (ERDF)

DigiMe Aufbau Workshop







Workshopablauf

- Inhalte
  - Kurzeinführung (WH) micro:bit und Makecode Oberfläche
  - Kennenlernen des Funkmoduls (Senden von Nachrichten)
  - Making -> Coding und Technik (Elektronik + Sensoren)
- Beispiele und Einsatzmöglichkeiten im Unterricht
- Aufbau Microbit (Pins, Stromversorgung, ...)

Praxisteil

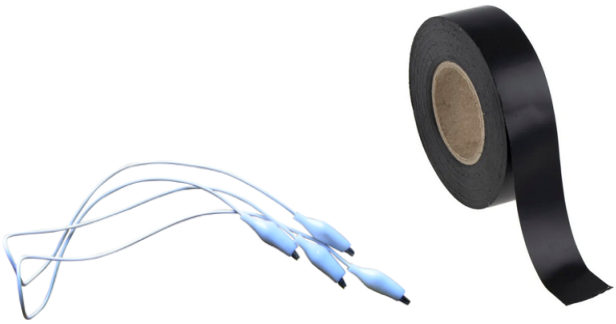
- Verwendung der Pins des Microbits:
  - Ausgabe (LEDs, Servo, Motor, LC Display)
  - Eingabe (Joystick, Taster)
  - Sensoren (Licht, Bewegung, Entfernung, ...)
- Elektronik Basics (3Volt vs 5Volt, Motortreiber, Pullups, ...)
- Verwendung und Möglichkeiten des Funk Moduls
- Beispiele für mögliche Projekte

Heißer Draht – Aufbau 1

Du kennst sicher das Geschicklichkeitsspiel "Der heiße Draht". Dieses Spiel wollen wir mit Hilfe des Microbits, etwas Elektronik und Bastelarbeit nachbauen. Je nachdem wie fit du beim Coding bist, kannst du eigene Features bzw. Erweiterungen einbauen.

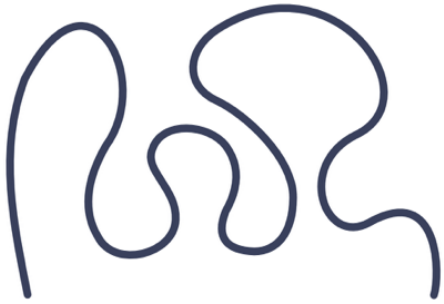
Zutaten

- 1 Stück Draht
- 2 Kabel mit Krokoklemmen
- 1 Holzbrett oder Plastilin
- 1 micro:bit
- Isolierband



Aufbau der Drahtstrecke

Biege den Draht mit einer Zange nach deinem Geschmack. Je enger die Drahtstrecke – desto schwieriger wird es, den Parcours fehlerfrei zu meistern.

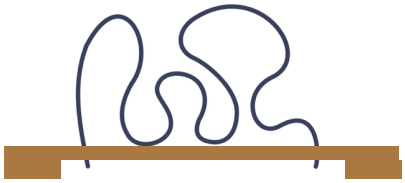


Zeichne am Holzbrett den Anfangs- und Endpunkt der Drahtstrecke und bohre ein Loch mit dem Durchmesser des Drahtes.

Tipp: Wenn du den Abstand der Löcher 10mm geringer machst, klemmt die Drahtstrecke besser im Brett.

Nun steckst du die Drahtstrecke durch die soeben gebohrten Löcher. Danach biegst du auf der Unterseite des Holzbretts beide Enden des Drahtes ca. 10mm um und schneidest die überschüssigen Enden mit einem Seitenschneider ab.

Montiere 2 schmale Holzstreifen auf der Unterseite des Holzbretts, damit das Spiel plan am Tisch stehen kann.



Alternativ kannst du anstelle des Holzständers 2 Klumpen Plastilin nehmen, in die du die 2 Enden der Drahtstrecke drückst.

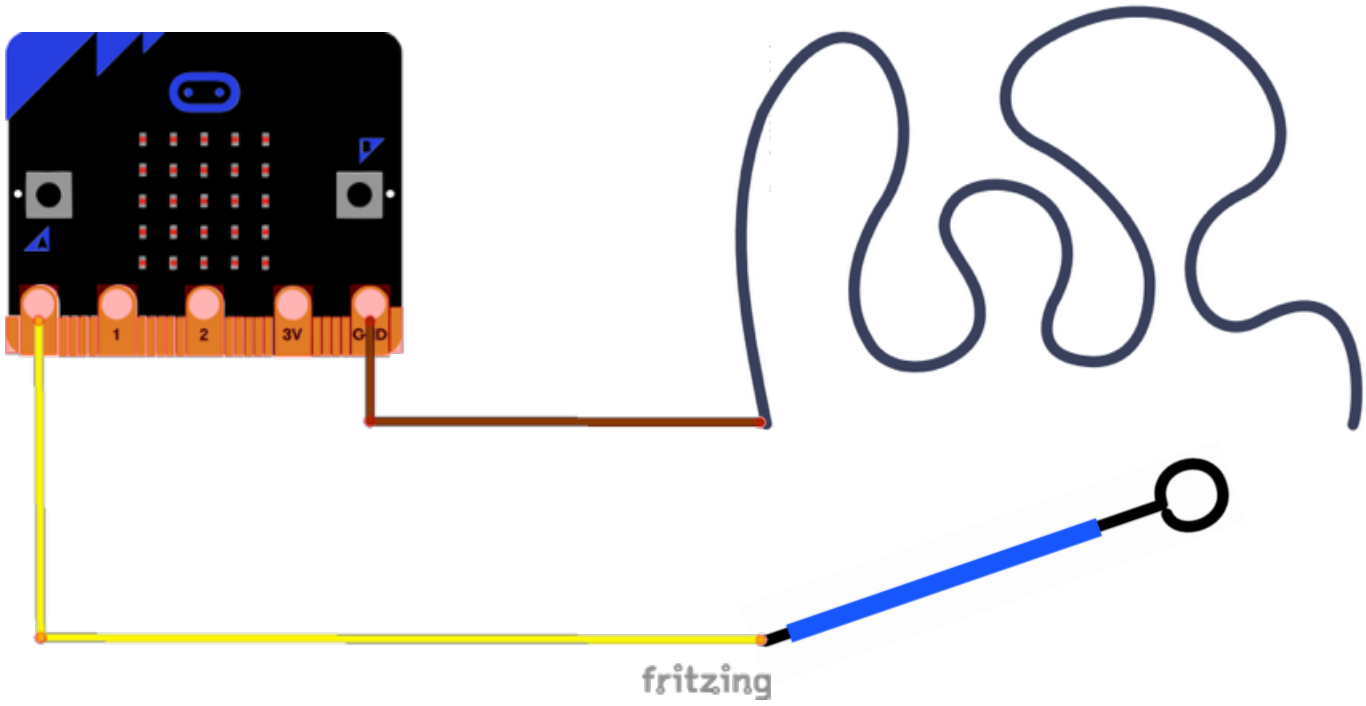
Heißer Draht – Aufbau 2

Aufbau der Drahtschlaufe

Biege am Ende eines ca. 15cm langen Drahtstücks eine Schlaufe mit einem Durchmesser von ca. 15mm. Je kleiner der Durchmesser ist – desto höher wird der Schwierigkeitsgrad des Spiels.
Den Schaft umwickelst du anschließend mit Isolierband, die letzten 10mm des Schafts bleiben jedoch unisoliert.

Verschaltung

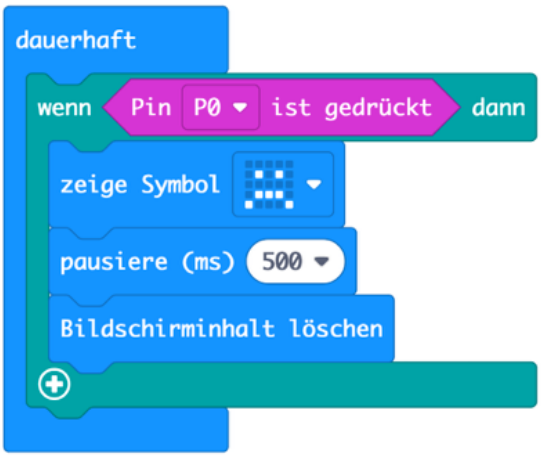
Verbinde mit einem Krokoklemmenkabel ein Ende der Drahtstrecke mit dem Minuspol (GND) des Microbits.
Das andere Krokoklemmenkabel führst du anschließend vom Pin0 des Microbits zum unisolierten Schaft der Drahtscheife.



Heißer Draht – Coding

Programmierung

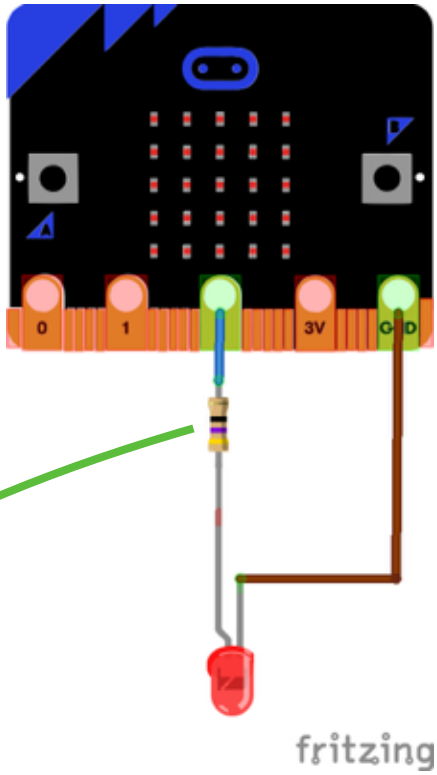
Da deine Drahtschlaufe mit dem Pin0 verbunden ist, musst du den Pin0 abfragen, ob eine Berührung erkannt wurde.
Ist dies der Fall, wird der traurige Smiley angezeigt und anschließend wieder gelöscht.



https://makecode.microbit.org/_a44akyhLFPyH

Fehleranzeige durch LED (optional)

Zusätzlich kannst du dir jeden Fehler auch durch das Leuchten einer LED (Leuchtdiode) anzeigen lassen.
Du benötigst dazu lediglich eine Leuchtdiode in der Farbe deiner Wahl und einen Vorwiderstand (die Kosten belaufen sich auf unter 1€).



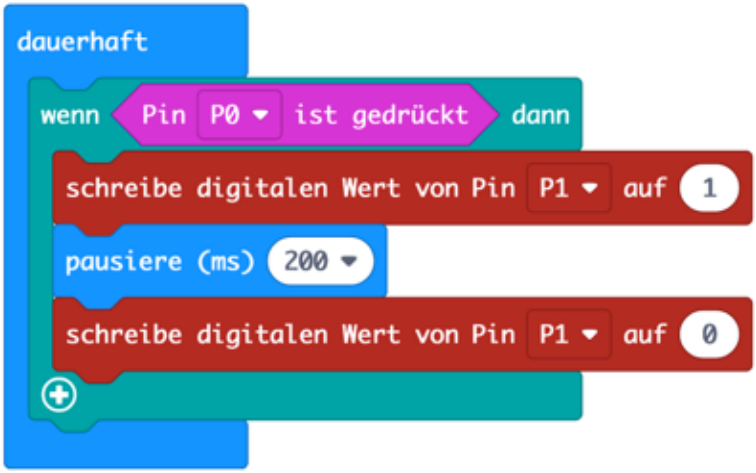
Hinweis

Damit die Leuchtdiode beim Betrieb nicht zerstört wird, benötigst du einen sogenannten Vorwiderstand. Obwohl dieser so heißt, spielt es keine Rolle, ob dieser im Stromkreis vor oder nach der Leuchtdiode verbaut wird. Für "herkömmliche" im Handel erhältliche LEDs benötigst du für unsere 3V Spannungsquelle des Microbits einen 47 Ohm Widerstand mit den Farben schwarz-violett-gelb.

Nähere Details bezüglich Berechnung von Vorwiderständen findest du im Arbeitsblatt "Ohmsche Gesetz".

Heißer Draht – Buzzer

Erweiterung Summer



https://makecode.microbit.org/_LfhCLTHuXPY3

Tipp

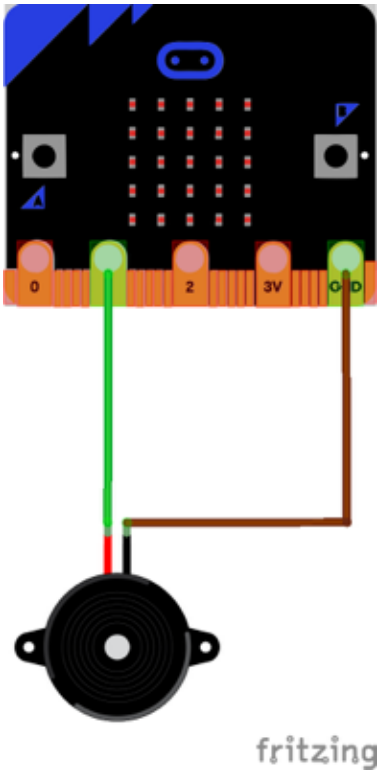
Den Block 

findest du in der Kategorie 

im Bereich 

.

Für eine akustische Anzeige kannst du einen Minibuzzer verwenden.
Schließe den Minuspol (kurzer Pin) über ein Krokoklemmenkabel an den GND des Microbits bzw. an die Drahtstrecke (da diese ja bereits mit dem GND verbunden ist).
Den Pluspol des Buzzers (langer Pin) verbindest du über ein weiteres Krokoklemmenkabel mit dem Pin1 des Microbits.



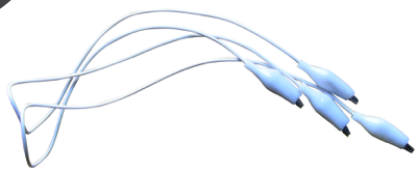
Spiele Buzzer – Aufbau

Wer kennt die Situation nicht? Mehreren Spielern wird eine Frage gestellt und man muss entscheiden, wer als Erstes geantwortet hat.

Diese Entscheidung übernimmt ab sofort eine selbst programmierte Schaltung. Bei der Person, die zuerst drückt, erscheint ein Licht und alle anderen MitspielerInnen haben keine Möglichkeit mehr zu drücken. Erst nach Freigabe des Spielleiters wird ein erneutes Drücken gewertet.

Zutaten

- Taster
- 2 Kabel mit Krokoklemmen
- 1 micro:bit



Funk

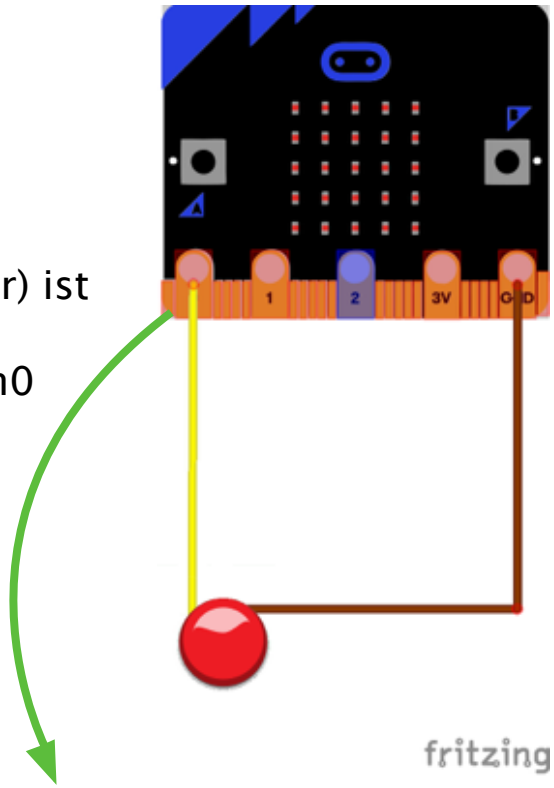
Damit alle SpielerInnen drahtlos miteinander kommunizieren können, bedarf es der Verwendung des Funkmoduls, welches im Microbit verbaut ist.

Voraussetzung:
Die Microbits, die miteinander kommunizieren, müssen sich in der gleichen Funkgruppe (gleicher Kanal) befinden.

Spiele Buzzer – Sender

Buzzer (Sender)

Die Verdrahtung des Buzzers (Funksender) ist sehr einfach.
Ein Kontakt des Buzzers wird mit dem Pin0 verbunden – der andere Kontakt mit dem Minuspol (GND) des Microbits.

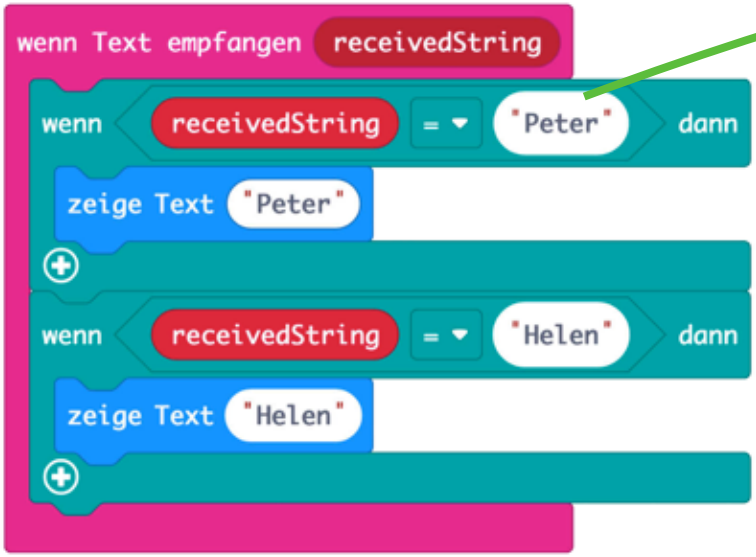


Im nicht-gedrücktem Zustand liegen am Pin0 des Microbits 3V (Zustand 1) an.
Drückt man den Buzzer – verbindet man den Pin0 über den Taster mit dem Minuspol (GND) und es liegen 0V an.

Im Codingteil wird der Zustand des Pins0 abgefragt.
Sobald dieser gedrückt wird – wird der Text "Peter" per Funk verschickt.
Schicke für jeden der verwendeten Buzzer den Namen des(r) jeweiligen Spielers(in).

Spiele Buzzer – Empfänger

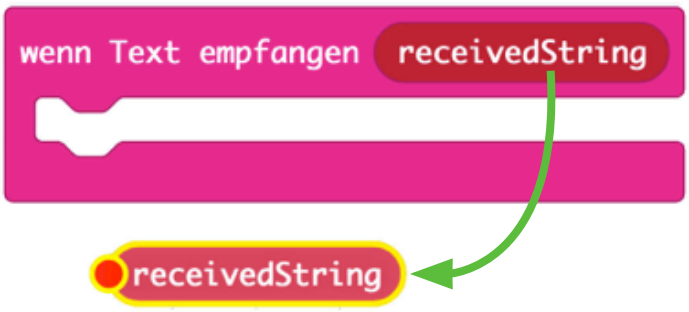
Buzzer (Empfänger)



Um hier einen Namen eintragen zu können, benötigst du aus den Kategorien "Fortgeschritten" und 

diesen Block 

Sobald ein Text empfangen wird, wird dieser automatisch im Buffer "receivedString" gespeichert und kann wie eine herkömmliche Variable abgefragt werden.



Um die Variabe "receivedString" verwenden zu können, ziehst du sie einfach aus dem Block "wenn Text empfangen ..." heraus.

Spiele Buzzer – advanced

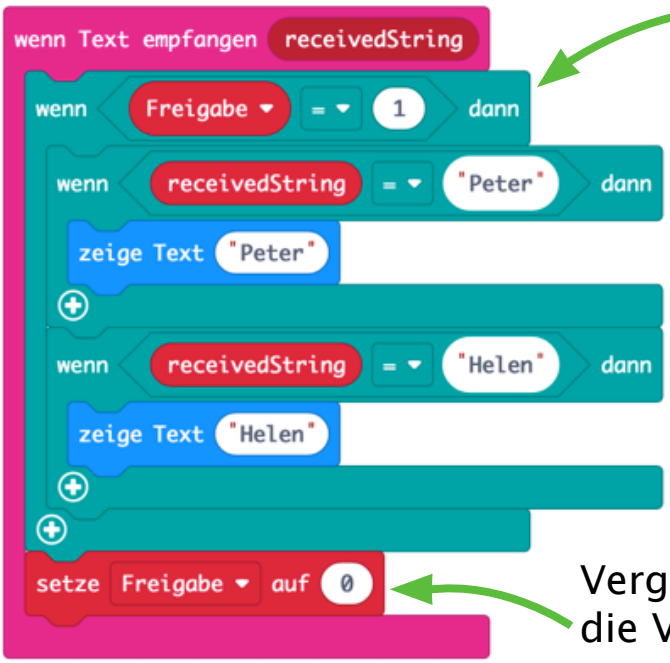
Problemstellung

Der Funkempfänger zeigt zwar den Namen an, dessen Buzzer als Erstes gedrückt wurde – alle weiteren Buzzerdrücke werden aber in einem Buffer gespeichert. Dies hätte zur Folge, dass nacheinander auch alle weiteren Namen angezeigt werden würden.

Dies soll mit Hilfe der Zusatzvariable (Freigabe" verhindert werden. Die Buzzerdrücke werden nur dann ausgewertet, wenn die Variable "Freigabe" auf 1 steht.

Bei Programmstart wird die Freigabevariable auf 0 gesetzt, sodass ein Drücken auf den Buzzer zwecklos ist.

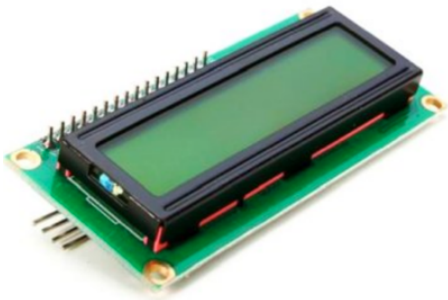
Mit einem Klick auf den Button B wird die Variable "Freigabe auf 1 gesetzt, sodass der nächste Buzzerdruck ausgewertet kann.



Solange die Freigabevariable 1 ist – werden empfangene Funktext ausgewertet.

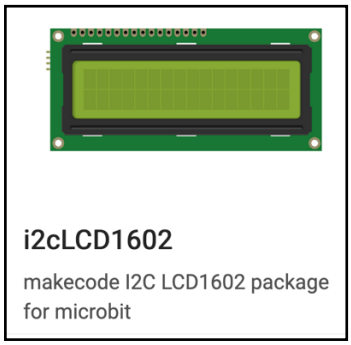
Vergiss nicht am Ende der Auswertung die Variable "Freigabe" wieder auf 0 zu setzen, damit Buzzerdrücke nicht mehr ausgewertet werden (erst nach erneutem Reset durch den Taster B).

LCD Buzzeranzeige

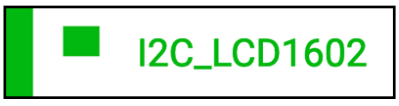


Wir erweitern das vorhergehende Beispiel, indem ein LC Display beim Spielleiter den Spielernamen anzeigt, dessen Buzzer als Erstes gedrückt wurde.

Voraussetzung dafür ist die Implementierung des LCDs in der Makecode Oberfläche. Dies erreichst du, indem du in der Kategorie "Fortgeschritten" und "Erweiterungen" den Begriff "LCD" suchst.



Von den vorgeschlagenen Option wählst du folgendes LC Display: **i2cLCD1602**.

Danach steht dir folgende Kategorie 

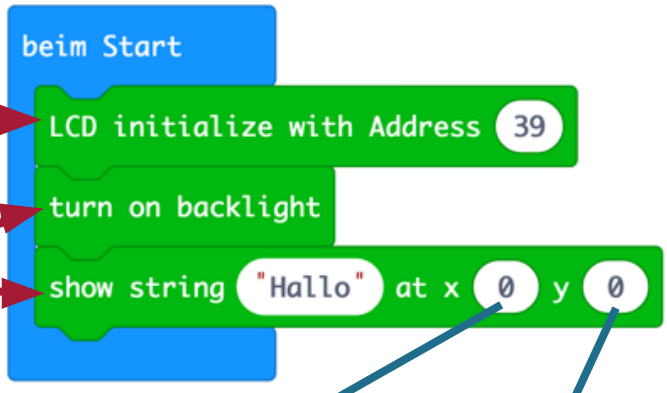
zur Verfügung.

Die wichtigsten Blöcke für die Verwendung des LC Displays:

Damit das Display verwendet kann, muss es mit dieser Zeile initialisiert werden.

Die Hintergrundbeleuchtung schaltest du auf diese Weise ein.

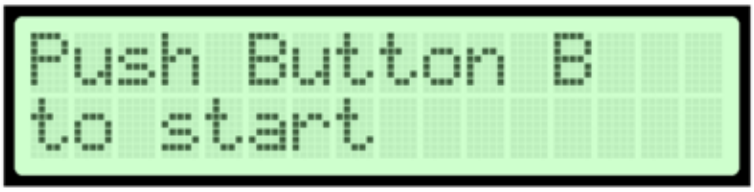
Mit dieser Zeile gibst du an, was auf dem Display angezeigt werden soll.

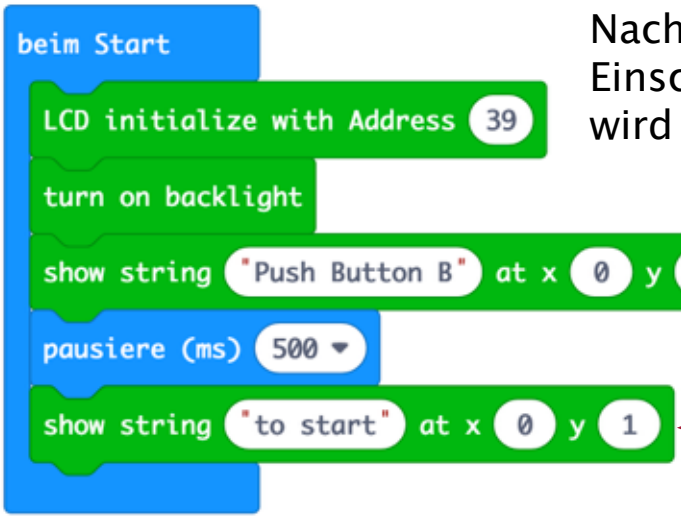


1. Zeichen

1. Zeile

LCD Buzzeranzeige - Coding

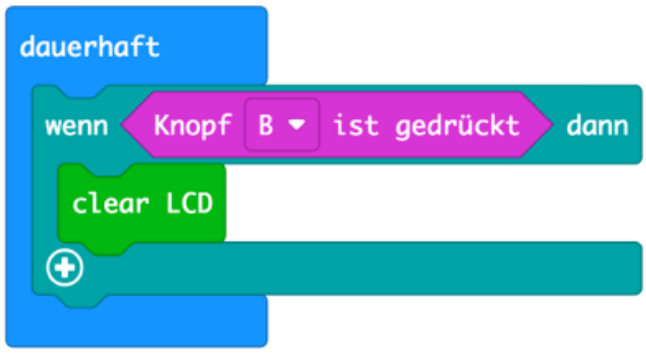




Nach dem Initialisieren des Displays und Einschalten der Hintergrundbeleuchtung wird in Zeile1 "Push Button B" ausgegeben.

Nach einer halben Sekunde Pause wird "to start" in Zeile2 angezeigt.

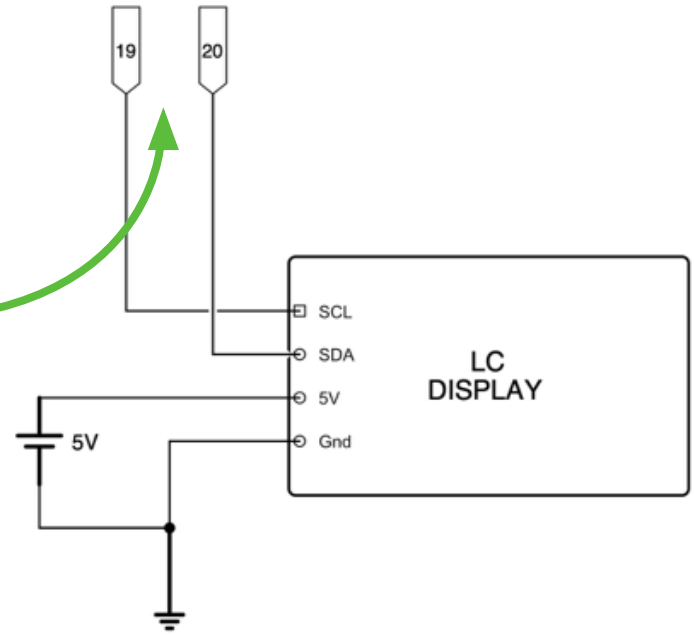
Mit einem Druck auf den Taster B wird das Display gelöscht und du bist bereit für die zweite Runde.



LCD Buzzeranzeige – Verdrahtung

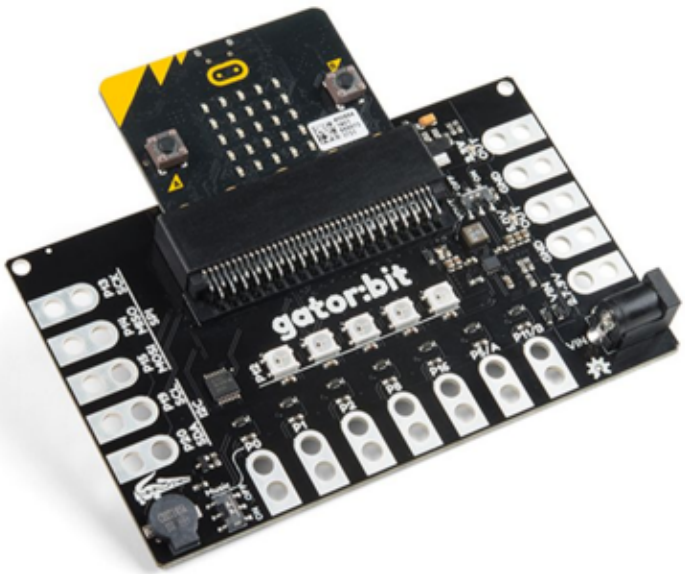
Im Schaltplan rechts siehst du wie die Verdrahtung des LC Displays am Microbit vorgenommen werden muss.

Verbinde SCL mit dem Pin19 und SDA mit PIN20.

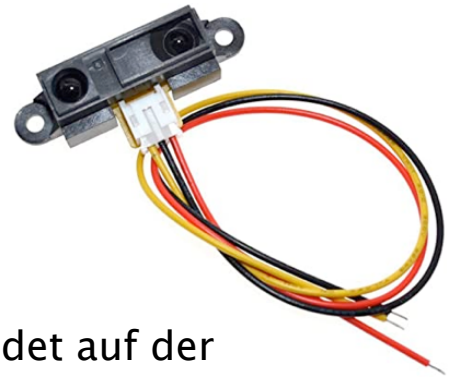


Wichtig!

Du benötigst eine externe 5V Spannungsquelle, damit das LC Display zuverlässig arbeiten kann. Dies kann eine Powerbank sein, ein externes Netzteil oder so wie in unserem Fall ein Add On Board für den Microbit, welches neben einer 5V Spannungsquelle auch die wichtigsten Pins zum Abgreifen mit einer Krokoklemme ermöglicht.



IR Entfernungssensor – Theorie

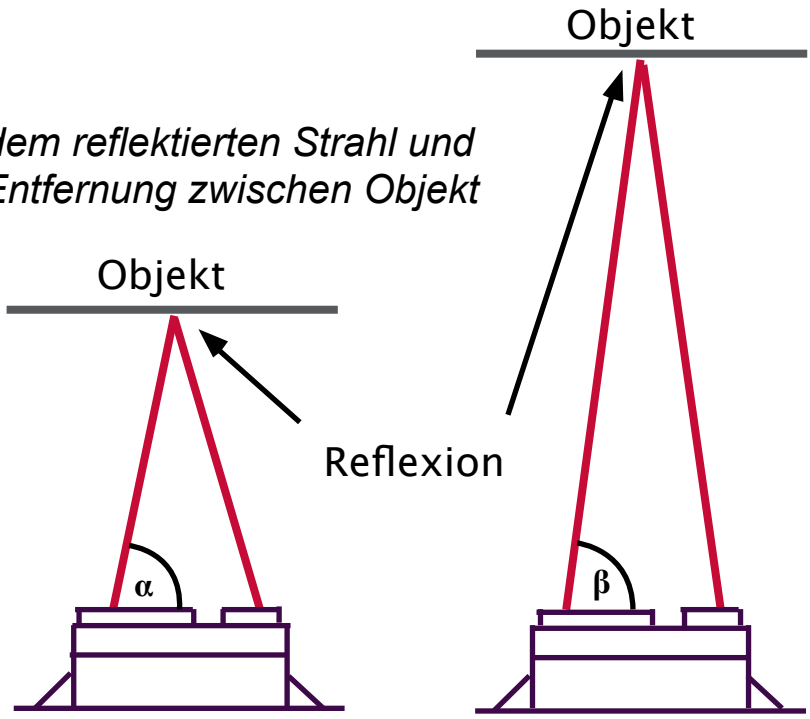


Theorie

Der Infrarot Entfernungssensor von Sharp sendet auf der einen Seite einen Infrarotstrahl aus.
Trifft der Strahl auf kein Hindernis, wird dieser nicht reflektiert und der Empfänger auf der anderen Seite bekommt keine Information.

Wird der IR Strahl durch ein Hindernis reflektiert, trifft dieser in einem bestimmten Winkel (abhängig von der Entfernung zum Hindernis) auf den Empfänger und die Entfernung wird intern berechnet.

Mit Hilfe des Winkels zwischen dem reflektierten Strahl und dem Empfänger wird intern die Entfernung zwischen Objekt und Sensor bestimmt.



Anwendungungsmöglichkeiten

- Einparkhilfe
- Roboter
- Händetrockner
- Alarmanlage
- Toilettenspülung
- Türöffner

IR Entfernungssensor – Coding

Verdrahtung

Die zwei Stromversorgungskabel des IR Sensors kannst du direkt mit dem Microbit verbinden. *[rot -> 3V / schwarz -> GND]*
Das gelbe Kabel verbindest du mit dem Pin 0,1 oder 2.

In der Kategorie 

findest du den Block 

.
Dieser Wert liefert dir, je nach Entfernung, einen Wert von 0 bis 1023.
Je höher der Wert – desto geringer ist die Entfernung zum Hindernis.

Mit dem Vergleichsoperator 

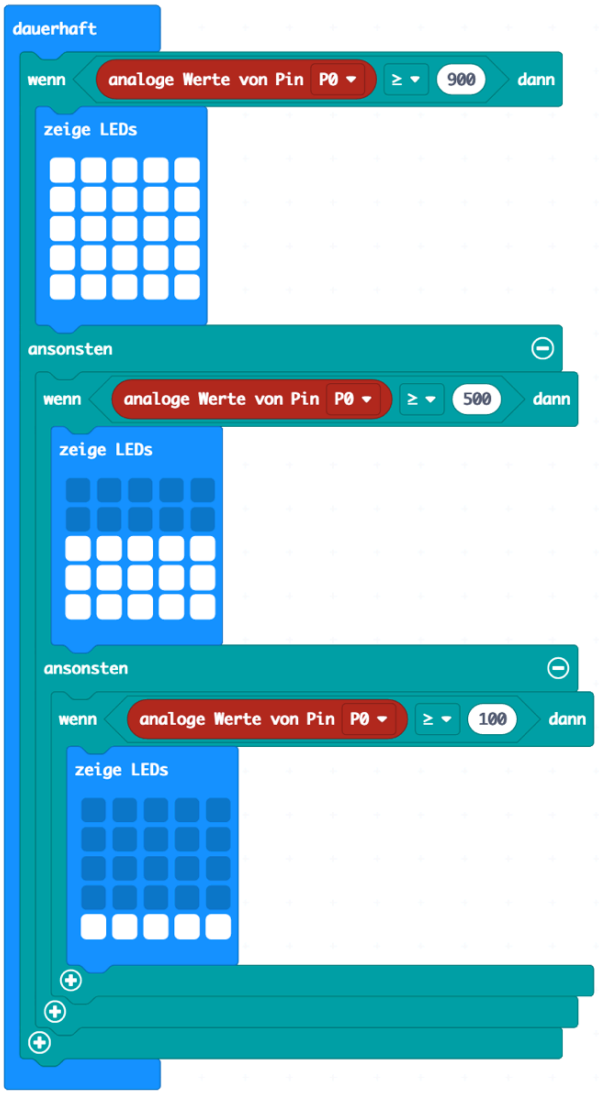
aus der Kategorie Logik, kannst du überprüfen, ob ein bestimmter Wert überschritten wird und dementsprechende Anweisungen ausführen.

Einparkhilfe

Für ein zentimetergenaues Einparken wäre es hilfreich ein optisches Feedback auf einem Display zu erhalten.

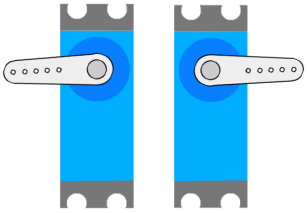
Dafür musst du lediglich den Pin0, an dem der IR Sensor angeschlossen ist, auf dessen Wert abfragen.

Je höher der Wert – desto mehr Balken (geringere Entfernung) werden angezeigt.



Servo – Allgemeines

Funktionsweise



Im Gegensatz zu einem herkömmlichen Motor dreht sich ein Servomotor nicht um 360°, sondern nimmt eine gewisse Stellung in einem Bereich von 0° bis 180° ein.

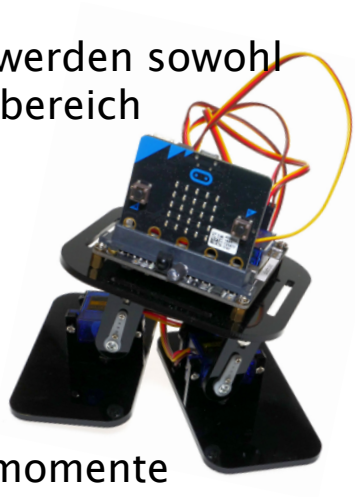
Ein gewöhnlicher Servo kann sich mechanisch nur im Bereich von 0° bis 180° drehen, da er einen Anschlag hat (Ausnahme: 360° Continuous Rotation Servo).

Neben der Stromversorgung (braun – 0V, rot – 5V) steht eine Steuerleitung (gelb) zur Verfügung. Die Dauer des Impulses, die an dieser Leitung angelegt wird, bestimmt den Winkel des Servos. Alle 20ms (0,02s) erwartet der Servo einen Impuls, der von 1ms [0°] bis 2ms [180°] den Winkel bestimmt.

Info Einen Servo SG90 kann man mit verminderter Leistung direkt am Microbit betreiben. Mehrere Servos verwendet man am einfachsten mit einem dafür vorgesehenen Servoboard bzw. externer Energiequelle.

Einsatzmöglichkeiten

Servomotoren haben einen breiten Einsatzbereich. Sie werden sowohl in der Industrie, im Maschinenbau aber auch im Hobbybereich verwendet:

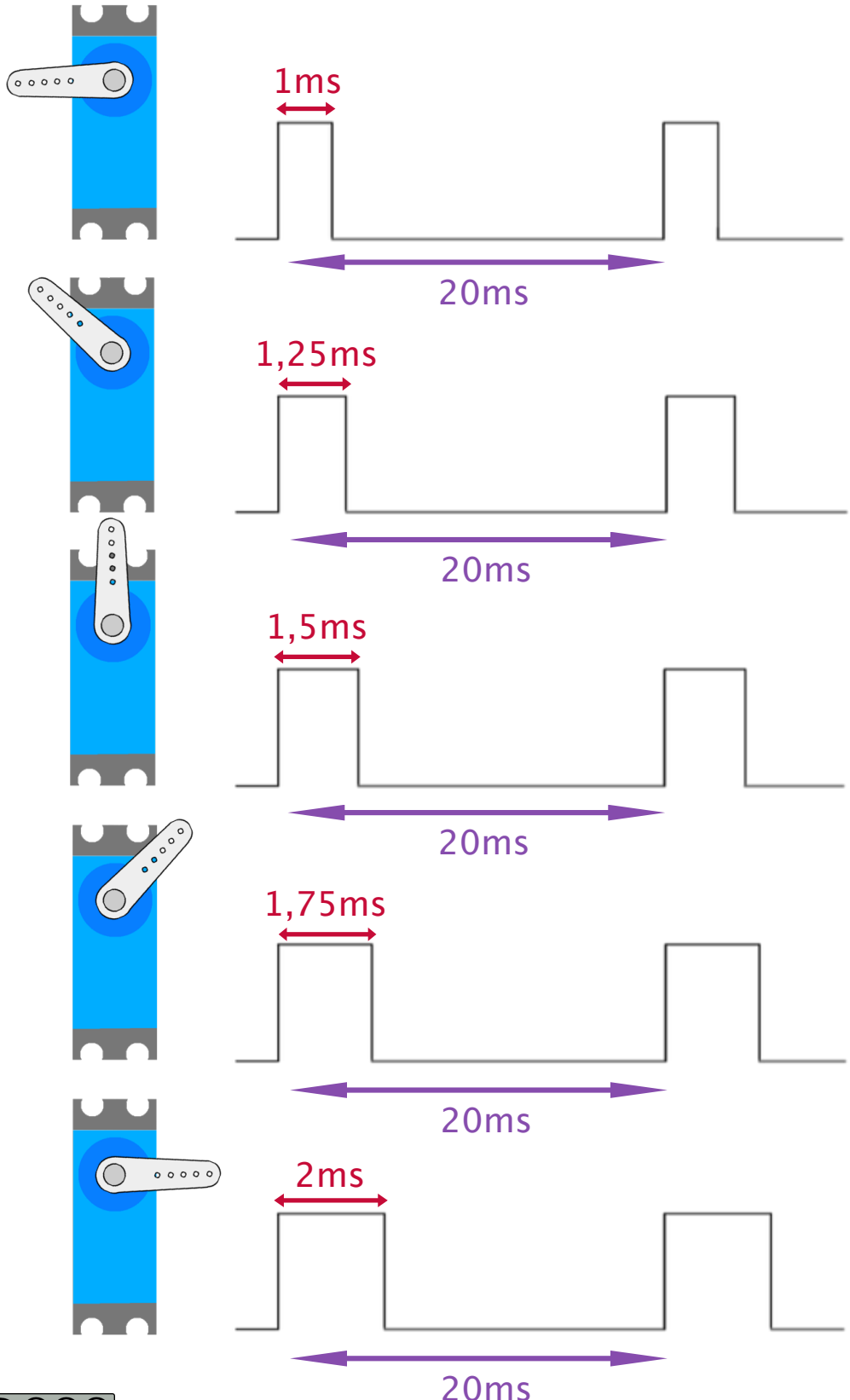


Industrie:	Roboterarm
Freizeitbereich:	Modellbau
Kraftfahrzeug:	automatische Sitzverstellung
Sensorik:	Positionieren von Sensoren

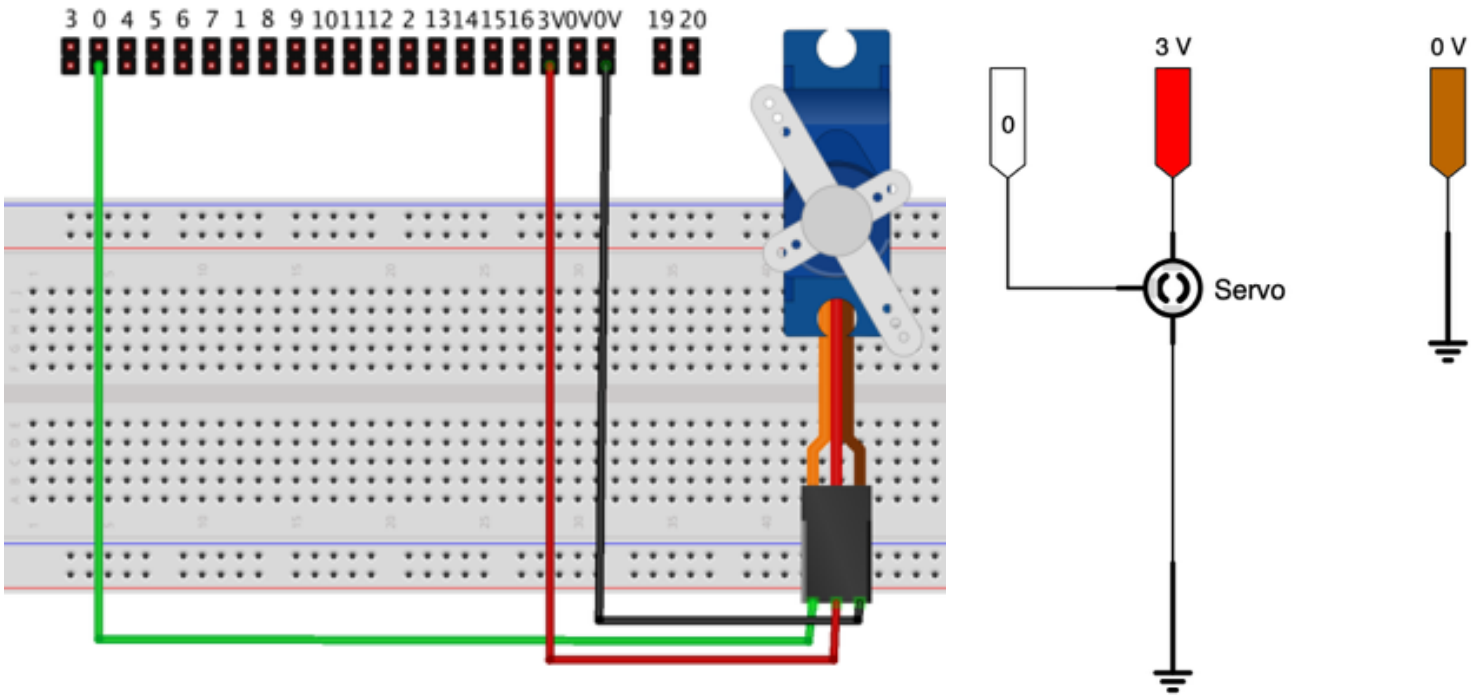
Servomotoren werden oft eingesetzt, wenn hohe Drehmomente und präzise, schnelle Bewegungen eine große Rolle spielen.

Servo – Theorie

Innerhalb einer Periodendauer von 20ms gibt die Pulsbreite des Steuersignals die Winkelstellung des Servos an. Bei einer Pulsbreite von 1ms nimmt der Servo den Winkel von 0° (ganz links) ein und wandert mit dem Ansteigen auf 2ms zu 180° (ganz rechts). Diese Werte bilden lediglich einen Richtwert und müssen dem Datenblatt entnommen werden.
Das Timing übernimmt der Mikrocontroller (Microbit) und wird bei der Programmierung durch die sogenannte Pulsweitenmodulation umgesetzt.



Servo – Verdrahtung



- Verkabelung des Servos: den Anschluss ganz links (Steuerleitung) führst du über das **grüne Kabel** zum **Pin0** des Microbits.
- Der mittlere Anschluss wird über das **rote Kabel** mit dem **3V Pin** verbunden.
- Zuletzt wird der Gnd des Servos über das **schwarze Kabel** mit dem 0V Pin des Microbits verbunden.

Info

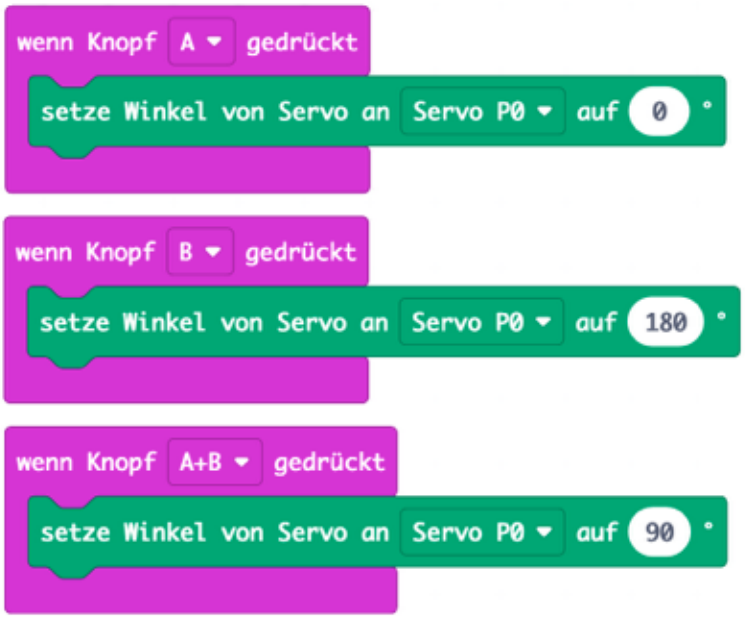
In den meisten Fällen benötigt ein Servo mehr Strom als der Microbit liefern kann. Sollte dies der Fall sein (Servo bewegt sich nicht oder "zittert"), musst du den Servo mit einer eigenen Stromquelle versorgen. Vergiss dabei nicht, die externe Masse (Minuspol) mit dem 0V Pin des Microbits zu verbinden.

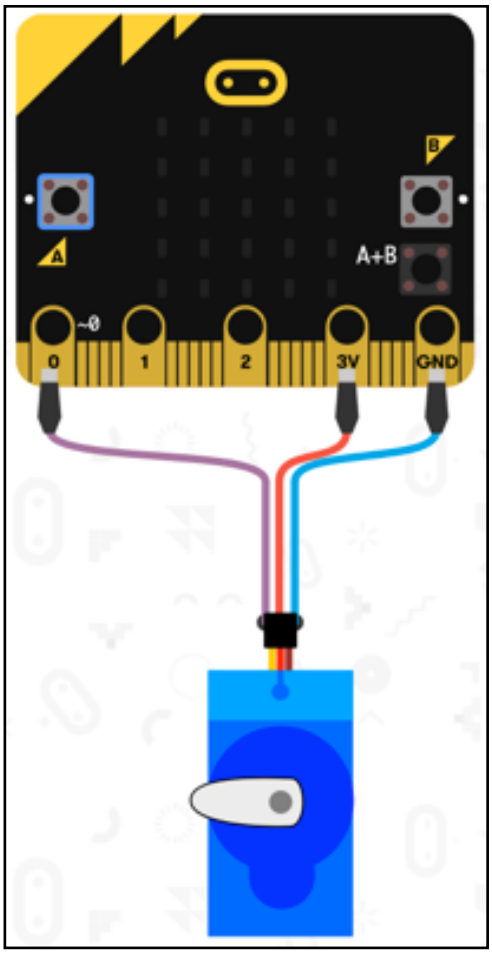
Servo - Coding

Voraussetzung dafür ist die Implementierung des Servos in der Makecode Oberfläche. Dies erreichst du, indem du in der Kategorie "Fortgeschritten" und "Erweiterungen" auf "Servo" klickst.

Danach steht dir folgende Kategorie  zur Verfügung.

Teste die Funktion des Servos, indem du 3 Winkelstellungen den Tastern zuweist.





Info

Du kannst die Funktion auch ohne Hardware wunderbar mit dem Online Simulator in der Makecode Oberfläche testen.

Servo – Helligkeitsanzeige

In diesem Beispiel soll die Umgebungshelligkeit mit einem am Servo angebrachten Zeiger angezeigt werden.

Dazu benötigst du den Block 

und den Block, mit dem du den Servowinkel angibst.

Problem

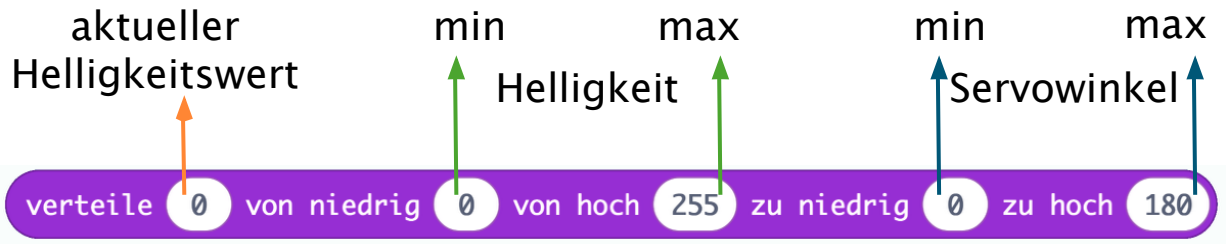
Wenn man wie im obigen Beispiel die Lichtstärke als Servowinkel ausgeben möchte, würde der Wertebereich nicht übereinstimmen.

Der Wertebereich des Servos beträgt 0–180 (Grad) während der Wertebereich des Helligkeitssensors im Simulator 0–255 und am Microbit selbst 0–1023 beträgt.

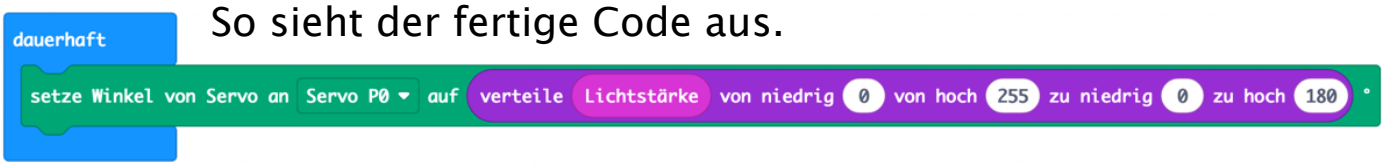
Lösung

Um einen Zahlenwert von einem Bereich in einen anderen Bereich abzubilden, kannst du die sogenannte "verteile" Funktion aus der Kategorie "Mathematik" verwenden.

In unserem Fall sollen die möglichen Helligkeitswerte 0–255 des Simulators den möglichen Servowerten 0–180 zugeordnet werden.



So sieht der fertige Code aus.



Wichtig: Solltest du den "echten" Microbit verwenden und nicht den Online Simulator, musst du den max. Helligkeitswert von 255 auf 1023 setzen.

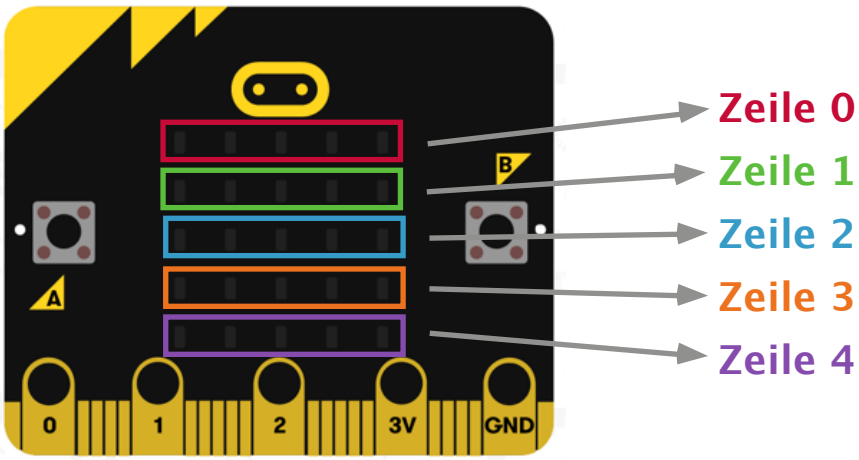
Zeichnen mit dem Joystick - 1

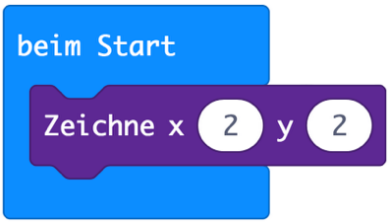
Bewege mit Hilfe des Joysticks das LED Pixel am Microbit. Mit jedem Tastendruck (Taster A) wird die aktuelle Position des Pixels gespeichert. Der Druck auf den Taster B zeigt dir das Gesamtbild aller gespeicherten Pixel.
Das können Icons, Emojis, Buchstaben oder einfache Bilder sein.

Dazu benötigst du den Block 

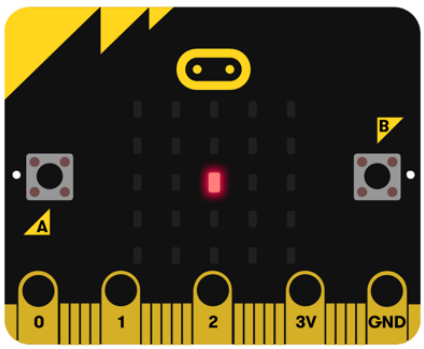
aus der Kategorie LED.

Wie in einem Koordiantensystem sind mit x die Spalten und mit y die Zeilen gemeint.
Bei Programmiersprachen ist es üblich beim Zählen bei 0 zu beginnen. Demnach ist die erste Zeile die Zeile 0 und die letzte die Zeile 4.

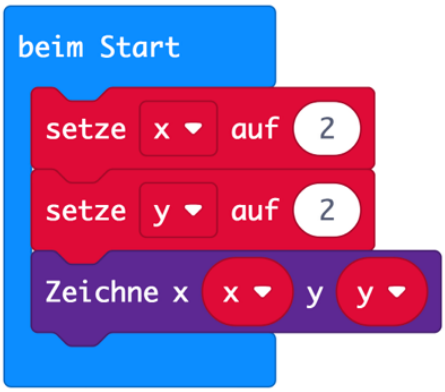




Soll das Pixel beim Start in der Mitte erscheinen, erreicht man dies mit den Koordinaten x=2 und y=2.



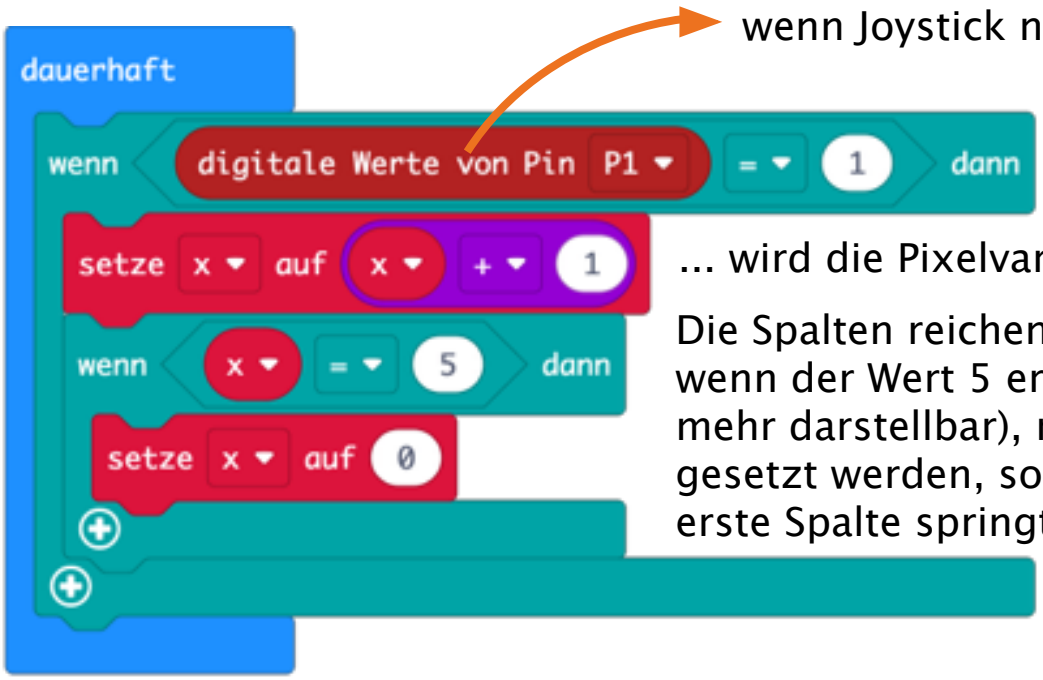
Da das leuchtende Pixel mit Hilfe des Joysticks "verschiebbar" sein soll, darf dieses nicht statisch, sondern muss dynamisch sein, d.h. mit Variablen angegeben werden.



Die Ausgabe am Display ist gleich - von nun an kann aber mit Hilfe des Joysticks der LED Wert erhöht bzw. verringert werden, sodass sich das LED Pixel bewegt.

Zeichnen mit dem Joystick - 2

Folgender Programmblock lässt das LED Pixel nach rechts wandern, sobald der Joystick nach rechts (Pin 1) gedrückt wird (vorausgesetzt die entsprechende Verdrahtung wurde vorgenommen).



... wird die Pixelvariable x um 1 erhöht

Die Spalten reichen von 0 bis 4, d.h. wenn der Wert 5 erreicht wird (nicht mehr darstellbar), muss dieser auf 0 gesetzt werden, sodass das Pixel in die erste Spalte springt.

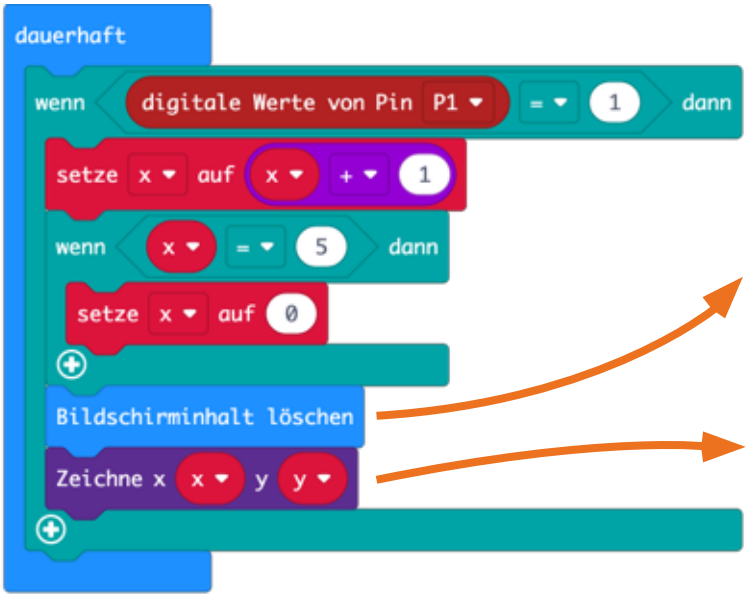
Verdrahtung:

Ein Joystick ist nichts anderes als 4 Taster, die jeweils einen Stromkreis bei entsprechender Bedienung schließen.
Da der Microbit am digitalen Eingang eine Spannung von 0 Volt als "gedrückt" erkennt, wird ein Kabel mit GND (0 Volt) verbunden und das andere mit dem jeweiligen Pin des Microbits z.B. Pin1. In dem Moment, in dem der Taster (z.B. Joystick rechts) gedrückt wird, wird der Stromkreis geschlossen und es liegen 0 Volt am Pin1 an, die vom Microbit als Tastendruck erkannt wird.

Zeichnen mit dem Joystick – 3

Um das Pixel auch anzeigen zu lassen (momentan wurde nur die entsprechende Variable angepasst), muss folgender Block hinzugefügt werden.

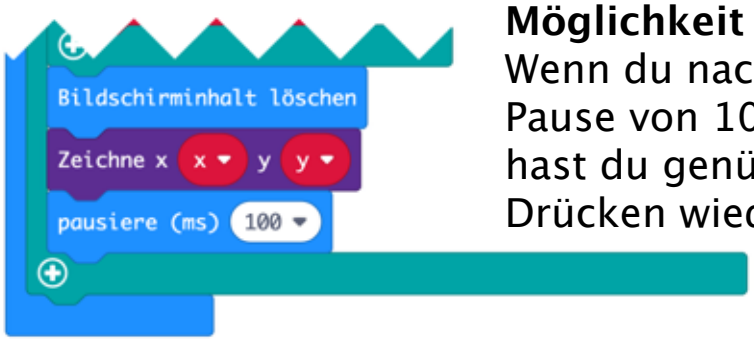
Da beim Betätigen des Joysticks nach rechts, das jeweils bestehende Pixel nicht gelöscht wird, würde bei jedem Druck ein weiteres Pixel hinzugefügt werden.
Um das jeweils vorhergehende Pixel zu löschen, muss vor der Ausgabe des aktuellen Pixels der Bildschirm gelöscht werden.



Löschen des vorhergehenden Pixels

Ausgabe des aktuellen Pixels

Da der Microbit Anweisungen rasend schnell ausführt, würde auch ein kurzer Tastendruck das Pixel sofort um viele Stellen weiter bewegen.
Um dies zu verhindern gibt es 2 Möglichkeiten:



Möglichkeit 1 (Kompromiss):
Wenn du nach der Ausgabe des Pixels eine Pause von 100ms (1 Zehntel Sekunde) einbaust, hast du genügend Zeit den Taster nach dem Drücken wieder loszulassen.

Hinweis

Drückst du den Taster bzw. Joystick zu lange (mehr als 100ms) – bewegt sich das Pixel trotzdem weiter. Machst du die Pause zu lange – kannst du das Pixel nicht mehr schnell genug weiterbewegen.

Zeichnen mit dem Joystick – 4

Möglichkeit 2 (elegante Methode):
Nachdem der Taster gedrückt wurde -> wird mit einer leeren während Schleife gewartet, bis der Taster (Pin1) wieder losgelassen wird.

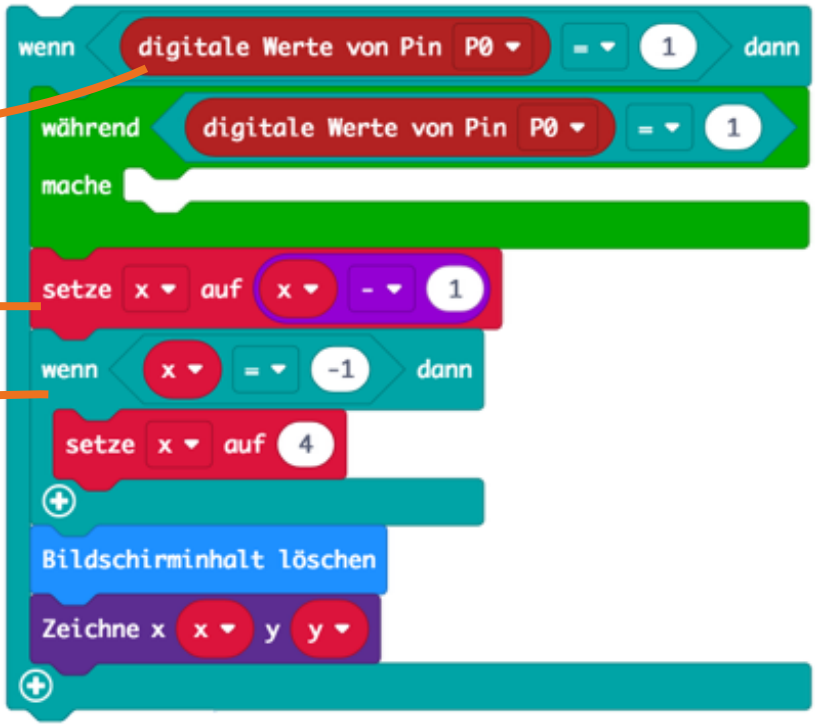
In dieser "Warteschleife" verweilt das Programm solange der Taster gedrückt ist. Sobald der Taster losgelassen wird, wird der Programmcode darunter fortgesetzt. Es ist also egal, wie lange du den Taster drückst – es wird nur als ein Tastendruck interpretiert.

Für die Bewegung des Joysticks nach links musst du folgendes ändern:

Joystick links wird an Pin0 angeschlossen.

Der Pixelwert verringert sich um 1.

Wenn die Spalte ganz links (x=0) um 1 verringert wird (x=-1) – muss diese auf x=4 ganz rechts gesetzt werden.

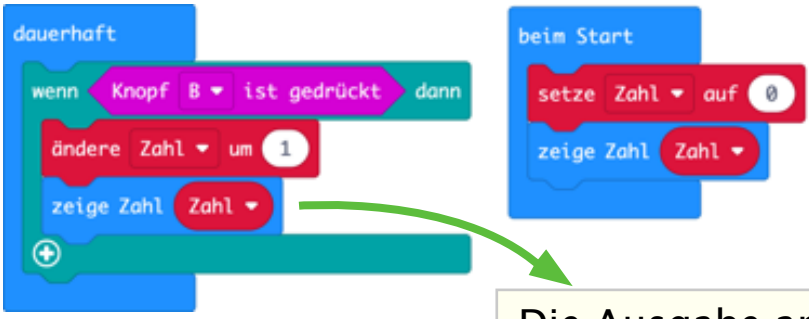


Gesamtsteuerung des Pixels mit dem Joystick:

https://makecode.microbit.org/_0VxWRVh5rhLe

Tastendruck für Fortgeschrittene – Problem

Aufgabe:
Beim Start zeigt das Display die Zahl 0.
Bei jedem Betätigen des Tasters B soll die Ausgabe um 1 erhöht werden.



Die Ausgabe am Display ist standardmäßig auf 0,6 Sekunden (600 ms) eingestellt.

Hinweis
Du kannst die Dauer der Anzeige verändern, indem du in Javascript bzw. Python nach dem Ausgabewert einen Beistrich setzt und die Dauer in Millisekunden (1 Sekunde = 1000ms) angibst.

Problem

Ist die Ausgabedauer zu kurz – springt die Anzeige zwei- oder mehrmals bei einem Tastendruck hoch.
Ist die Ausgabedauer zu hoch – muss man lange warten, bis der nächste Tastendruck erkannt wird bzw. wird ein Tastendruck gar nicht erkannt.

Tastendruck für Fortgeschrittene – Lösung

Problemlösung:
Eine elegante Lösung wäre, den Tastendruck erst nach Loslassen zu werten. Somit ist es dem Anwender überlassen wie kurz bzw. lang er den Taster betätigt – er wird lediglich einmal erfasst.

Umsetzung
Nachdem der Taster gedrückt wurde – wird gewartet bis der Taster wieder losgelassen wurde. Erst danach wird die Anweisung ausgeführt.

Dafür bietet sich eine Schleife an:

die Schleife wird solange wiederholt, solange der Taster gedrückt ist.
Erst beim Loslassen wird das Programm fortgesetzt.
Da während des Wartens nichts passiert, bleibt der "mache"-Block innerhalb der Schleife leer.

https://makecode.microbit.org/_9bpi07HUJasw

Damit auch ein kurzer Tastendruck erkannt wird, reduziere die Ausgabedauer am Display auf ca. 10ms.

```
basic.show_number(Zahl, 10)
```

Hinweis

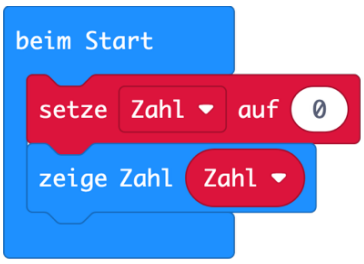
Die Darstellung mehrstelliger Zahlen müsste durch eine separate Abfrage auf eine längere Ausgabedauer am Display angangepasst werden, damit die Ausgabe nicht zu schnell erfolgt.

https://makecode.microbit.org/_Fsr7p915mh7c

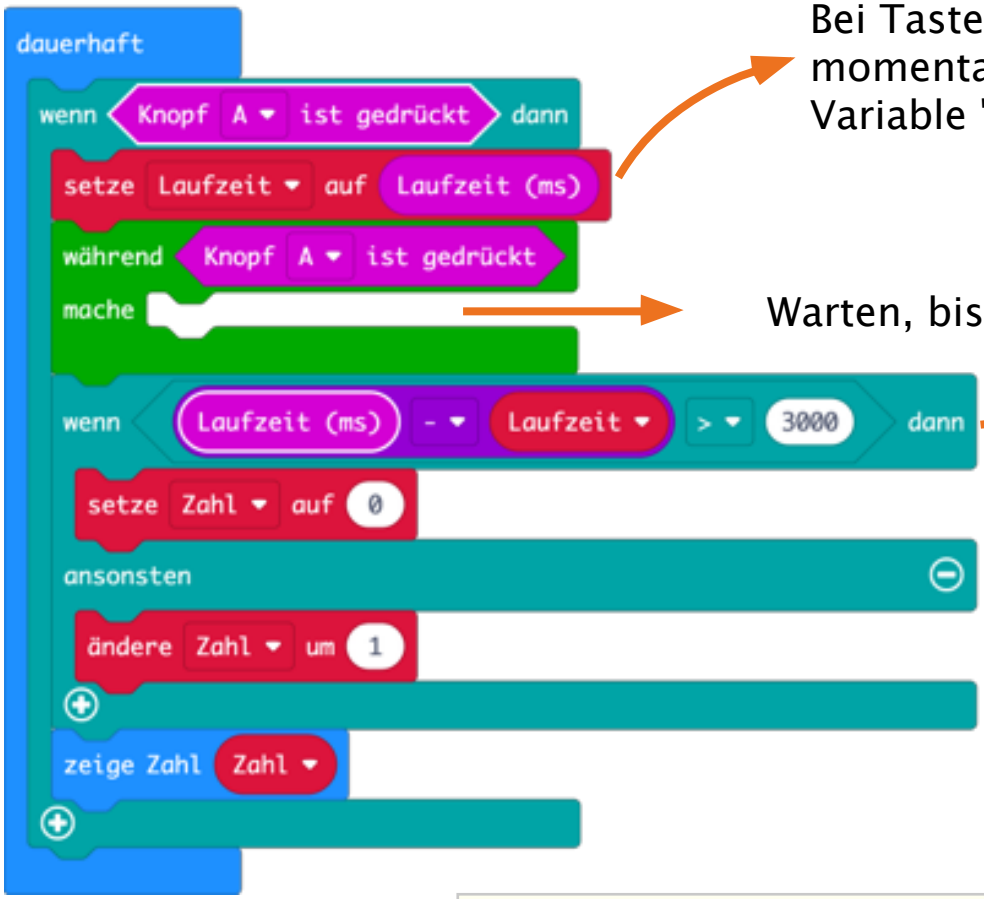
Unterscheidung Tastendruck kurz - lang

Aufgabenstellung

Mit einem einzigen Taster soll bei einem kurzen Tastendruck die jeweils nächste Zahl angezeigt werden. Mit einem langen Tastendruck soll die Anzeige auf 0 zurückgesetzt werden.
Dieses Szenario findet man z.B. bei Hardware, auf der nicht genug Platz für mehrere Taster vorhanden ist. Oder man versucht durch einen langen Tastendruck einen möglichen Reset nicht unabsichtlich durchzuführen.



Um dieses Szenario lösen zu können, benötigt man eine "Art Stoppuhr", die bekannt gibt, ob der Taster lang genug gedrückt ist.
Diese Aufgabe übernimmt ein sogenannter **Timer**. Der Timer startet bei der Stromversorgung des Microbits automatisch und zählt im Microsekundentakt hoch.



Bei Tastendruck wird die momentane Laufzeit in der Variable "Laufzeit" abgespeichert.

Warten, bis Taster losgelassen wird.

War der Tastendruck länger als 3 Sekunden, wird die Zahl auf 0 zurückgesetzt.

https://makecode.microbit.org/_8MWLTCPR8fDx

Verbesserung Tastendruck kurz – lang

Im vorangegangenen Beispiel wurde die Anzeige bei längerem Tastendruck erst nach Loslassen des Tasters zurückgesetzt.
Dieses Zurücksetzen soll nun sofort angezeigt werden, sobald die Zeit überschritten ist.

In dieser Schleife wird nicht nur abgefragt, ob der Taster losgelassen wird, sondern auch ob dieser mehr als 3 Sekunden gedrückt wurde.
Ist dies der Fall -> wird die Anzeige auf 0 zurückgesetzt.

https://makecode.microbit.org/_autMpUT7gJE7

Microbit – Pinbelegung

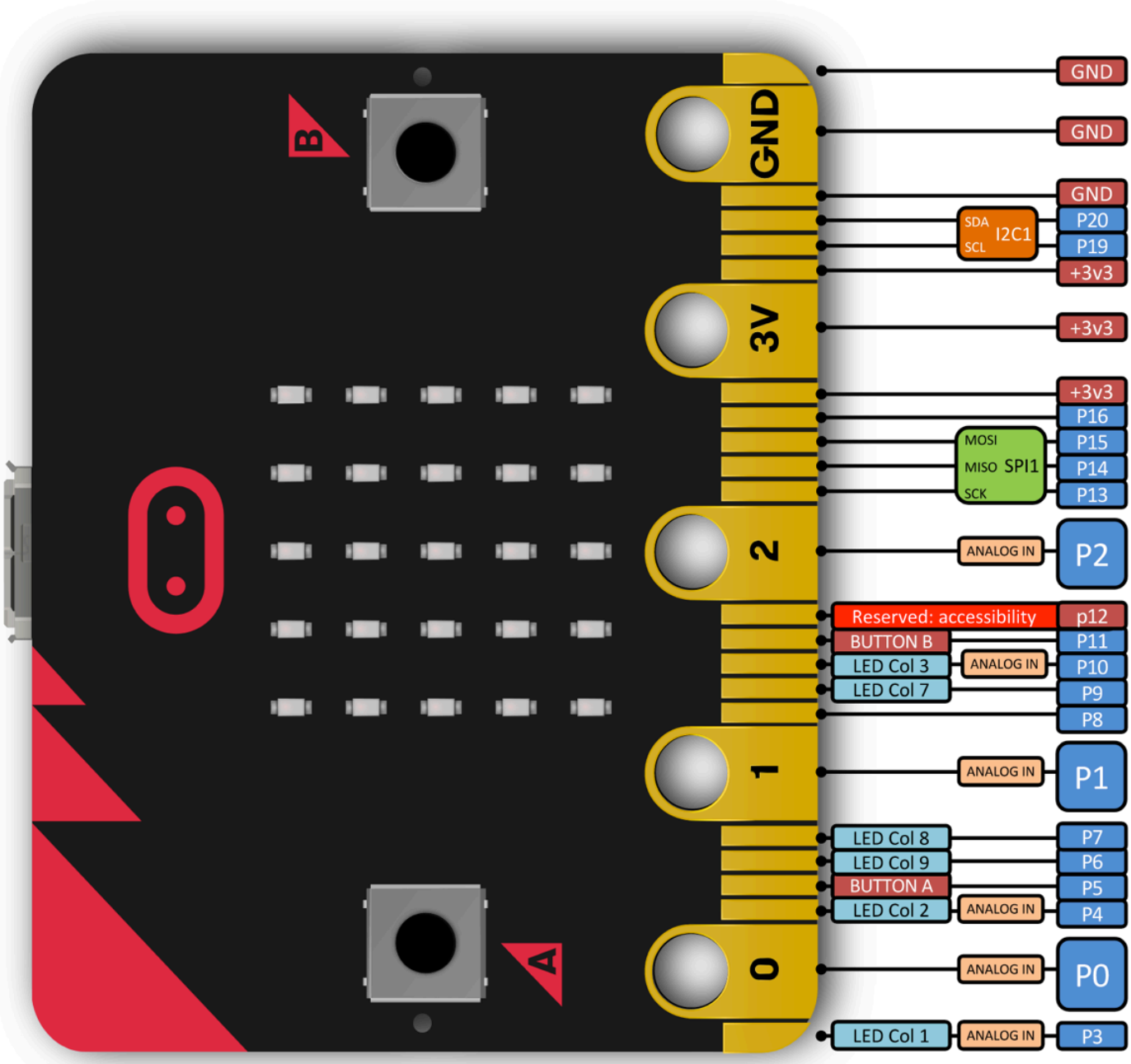


Bild: https://tech.microbit.org/hardware/schematic