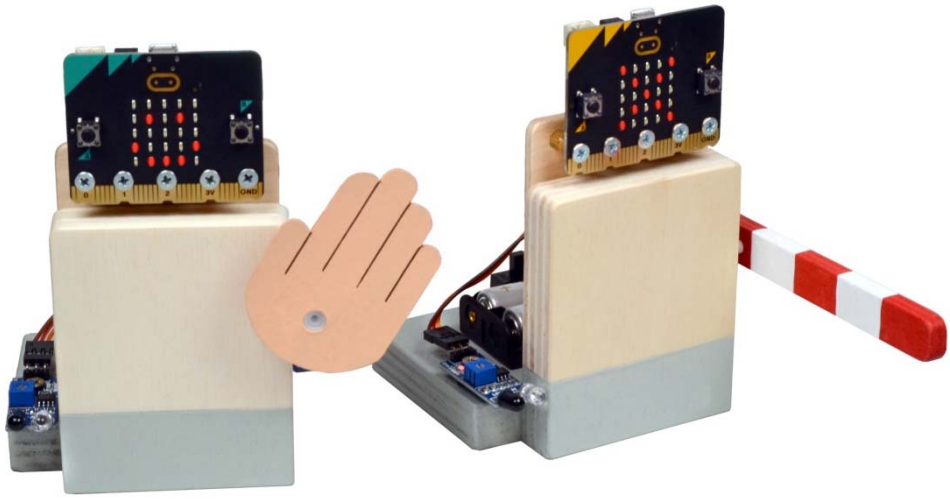
PROGRAMMIERANLEITUNG

Micro:bit SCHRANKE / WINKER mit IR-Sensor

Dieses **Micro:bit-Experimentiermodell** enthält einen **Servo** (180°) und einen **Infrarot-Sensor**. Damit lassen sich in Kombination mit dem Micro:bit zahlreiche Experimente aufbauen und programmieren, wobei die 20 grafischen Programme (MakeCode©) einen Schwierigkeitsgrad von einfach bis mittel aufweisen.



Grundlagen

1. Allgemeine Hinweise:

- Baue die Werkpackung **Schranke/Winker + IR-Sensor** laut beiliegender Anleitung zusammen. Befestige einen Micro:bit mit fünf Senkkopfschrauben M3 x 8 mm an den Messinghülsen und setze zwei neue **Mignon-Batterien** (je 1,5 V) in die Batteriebox (3 V).
  Achtung: Schwache Batterien können Fehlfunktionen des Servos und IR-Sensors verursachen. Das Anlegen höherer Spannungen als 3 V kann zur Zerstörung des Micro:bit führen!
- Aufladbare Batterien (z.B. NiMH, NiCD) haben eine Spannung von 1,2 Volt und sind daher nur bedingt einsetzbar. Ideal und nachhaltig wäre aber die Verwendung von **USB-Powerbanks**.
- Halte den Micro:bit von Feuchtigkeit fern und berühre ihn möglichst nicht an den Kontakten.

2. Anforderungen:

Um den Micro:bit in Betrieb zu nehmen, braucht man:

- ein Laptop oder einen PC mit Windows 10 (8, 7) oder Mac (OSX oder Linux)
- ein Micro-USB-Kabel zum Anschluss des Micro:bit an den Computer
- einen Internet-Zugang (Chrome, Edge, Firefox …) - **Aber**: Für den Betrieb ohne Internet gibt es eine **App**: https://makecode.microbit.org/offline-app

Der Micro:bit kann auch über eine App mit einem **Tablet / iPad oder Smartphone** via Bluetooth programmiert werden. Dazu muss aber der Micro:bit mit diesen Geräten gekoppelt werden. Eine Anleitung dazu im pdf-Format findet man auf der **Winkler-Schulbedarf Homepage** www.winklerschulbedarf.com unter Service - Basteltipps (Download) - Micro:bit.

3. Den Micro:bit vorbereiten:

Schließe den Micro:bit über ein Micro-USB-Kabel an einen freien USB-Anschluss des PCs. Das Kabel dient sowohl zur Stromversorgung des Micro:bit als auch zur Datenübertragung. Der Micro:bit erscheint im Windows Explorer (PC) oder Filemanager (Mac) als Laufwerk mit dem Namen [MICROBIT] und einem Laufwerksbuchstaben, z.B. [E:]. Der Micro:bit kann dann über dieses Laufwerk mit einer Programmdatei (*.hex) versorgt werden.

Bei neuen Micro:bits ist ein **Demo-Programm** vorinstalliert, das die Funktionen des Micro:bit zeigt und zu verschiedenen Aktivitäten aufruft, z.B. Schütteln, Neigen, Taste drücken, usw. - Es wird später einfach durch eigene Programme überschrieben!

Bei Anschluss des Micro:bit an einen Computer muss die Batterieversorgung nicht getrennt werden, denn der Micro:bit schaltet automatisch auf USB-Versorgung um.

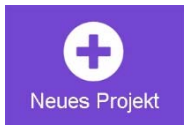
4. Der Makecode-Editor:

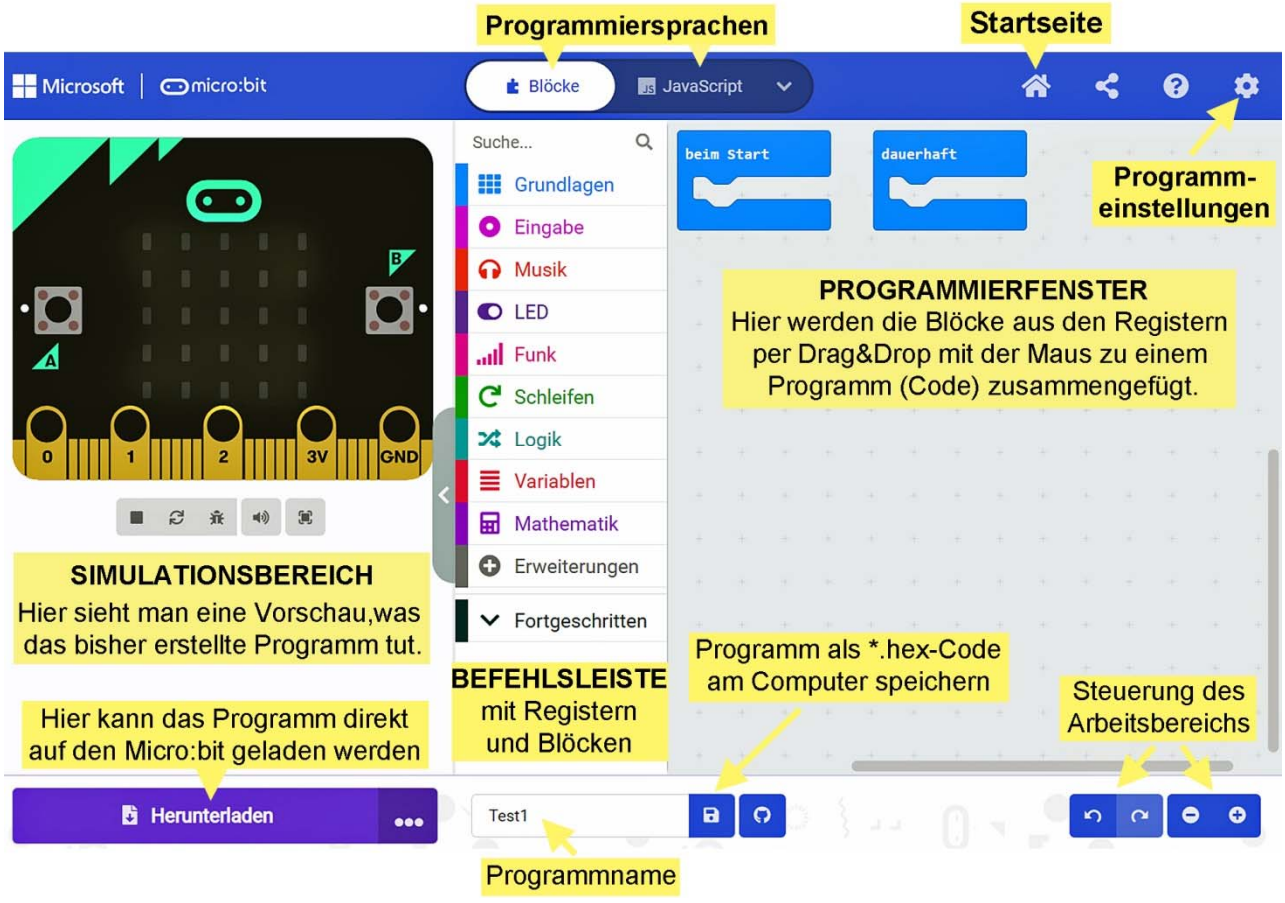
Zum Programmieren verwenden wir die grafische Programmierplattform **Makecode©** von Microsoft: https://makecode.microbit.org/. Eine grafische Programmierung ist ideal für Anfänger, die noch keine Programmiersprache kennen, denn sie ist intuitiv und leicht zu erlernen. **Makecode** läuft im Browser, daher braucht kein eigenes Programm installiert zu werden.

Programmierumgebung

1. Programmstart:

- Schließe den Micro:bit über ein **Micro-USB-Kabel** am Computer an.
- Der Micro:bit wird im Explorer als **Laufwerk** (z.B. MICROBIT [E:]) angezeigt.
- Öffne den Browser (Chrome, Edge, Firefox …) und gib folgende **Programm-URL** ein: https://makecode.microbit.org/
- Wähle die Schaltfläche [Neues Projekt] aus und gib dem Projekt einen Namen (z.B. Test1). Nun erscheint die **Programmieroberfläche**:





2. Programmbeschreibung:

Die Programmieroberfläche von **Makecode©** besteht aus drei Bereichen:
SIMULATIONSBEREICH, BEFEHLSLEISTE, PROGRAMMIERFENSTER

Im Simulationsbereich ist ein Micro:bit abgebildet, der das laufende Programm abspielt.

In der Befehlsleiste befinden sich verschiedenfarbige Register mit Blöcken zum Programmieren. Nach Anklicken der Register erscheinen diverse Blöcke, die man mit der Maus per Drag&Drop ins Programmierfenster ziehen kann. Die Blöcke erscheinen im Programmierfenster zuerst grau und nehmen ihre Originalfarbe erst wieder an, wenn sie richtig im Programm verankert sind. Die Blöcke lassen sich mit einem Klick der rechten Maustaste duplizieren und löschen oder man schiebt sie wieder zurück in die Befehlsleiste. Die Blöcke sind so geformt, dass sie nur

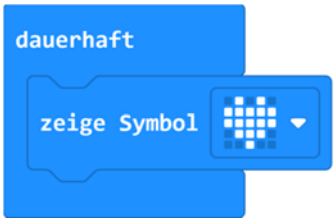
dann ineinander passen, wenn sie logisch zu den Programmbefehlen passen. Dadurch werden Programmierfehler stark reduziert. **Fortgeschrittene** können aber statt der grafischen **Block-Programmierung** auch **JavaScript** oder **Python** verwenden.

Nach einem Klick auf das Zahnradsymbol (rechts oben) können Einstellungen durchgeführt werden: z.B. Sprache, Programme löschen, zusätzliche Blockregister einfügen usw.

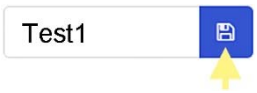
Ein Klick auf das Haus-Symbol (oben) öffnet die Startseite.

3. Ein erstes Testprogramm speichern:

Lösche den **Start-Block** [beim Start] durch Verschieben in den Registerbereich. Ziehe vom Register [Grundlagen] den **Symbol-Block »Herz«** in die **Block-Klammer »dauerhaft«**.



Klicke unten auf das **Diskettensymbol** neben dem **Programm-namen** (Test1). Das Programm wird nun lokal auf dem Computer als microbit-Test1.hex gespeichert.



4. Das Testprogramm auf den Micro:bit übertragen:

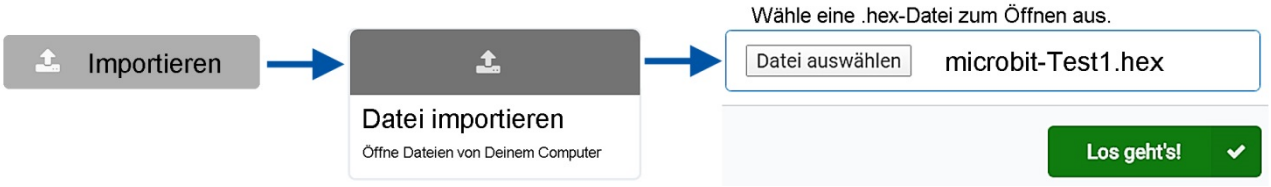
Die **Übertragung des Programms auf den Micro:bit** kann auf **zwei Arten** erfolgen:

- Öffne den Datei-Explorer und ziehe die Datei microbit-Test1.hex mit der Maus auf das Laufwerk [MICROBIT]. Bei der Übertragung blinkt zuerst ein gelbes Licht (Rückseite) und dann startet das Programm.
- Klicke erstmals im **Makecode-Editor** auf die Schaltfläche [Herunterladen], wähle das Laufwerk [MICROBIT] und klicke auf [Speichern]. Ab dem zweiten Mal kann jedes Programm durch einen einfachen Klick auf [Herunterladen] auf den Micro:bit übertragen werden.
- Mit der **Reset-Taste** auf der Rückseite des Micro:bit kann ein Programm neu gestartet werden.

5. Ein Programm (hex-Datei) importieren:

Um den **Programmcode einer hex-Datei** lesen und bearbeiten zu können, muss sie im Programmeditor **Makecode** geöffnet werden. Das kann auf **zwei Arten** erfolgen:

- Ziehe die entsprechende **hex-Datei** direkt **per Drag&Drop** vom Datei-Explorer auf das Programmierfenster von Makecode. Dort kann das Programm dann bearbeitet werden.
- Eine **hex-Datei** kann aber auch von der **Startseite von Makecode** importiert werden:
  Klicke auf die graue Schaltfläche [Importieren], dann auf [Datei importieren].
  Über [Datei auswählen] kann im Datei-Explorer die gewünschte hex-Datei gewählt werden.
  Nach einem Klick auf [Los geht's] wird das Programm auf dem Makecode-Editor geöffnet.

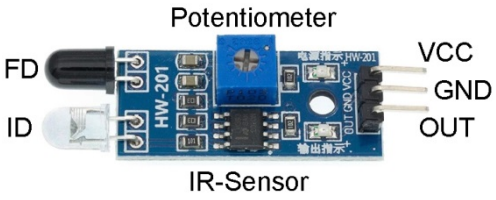


6. Servo (180°):

Ein 180° Servo ist ein elektronisch gesteuerter Getriebemotor mit einer Antriebsachse, die sich nur in einem Bereich von 180° vor- oder zurückdreht. Um diese Bewegung nutzen zu können, muss auf der Achse ein sogenanntes **Ruderhorn** befestigt werden.
Das Anschlusskabel besteht aus drei verschiedenfarbigen Drähten: **BRAUN** kommt an GND (-), **ROT** an 3 V (+) und **ORANGE** an **Pin 2** des Micro:bit. Über die orange **Signalleitung** an Pin 1 kann der Micro:bit den Servo winkelgenau von 0° - 180° steuern.
Obwohl Servos in der Regel mit 4.5 V bis 6 V betrieben werden, funktioniert der beiliegende Servo mit leicht verminderter Leistung schon ab 3 V.

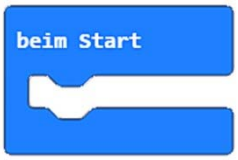
7. Infrarot-Sensor (IR):

Am **IR-Sensor** befindet sich eine **Infrarotdiode (ID)**, die unsichtbares Infrarotlicht abstrahlt. Nähert sie sich einem Gegenstand, dann wird das IR-Licht daran zurückreflektiert.
Eine dunkle **Fotodiode (FD)** reagiert auf dieses Licht und veranlasst, dass der Sensor ein Spannungssignal (0 - 3 V) an den Micro:bit sendet, der es in einen **Wert von 0 - 1023** umwandelt. Die Stärke des Signals ist abhängig von der Entfernung, Form und Oberfläche des Gegenstandes. Die **Lichtempfindlichkeit** des IR-Sensors kann am **Potentiometer** mit einem kleinen Schraubendreher eingestellt werden.
Beachte den richtigen **Anschluss** des Sensors:
VCC (+), **GND (-)** und **OUT** (Signal an **PIN 1**)
Leider reagiert die schwarze **Fotodiode (FD)** durch seitlichen Lichteinfall teilweise auch auf helles Tageslicht. Abhilfe schafft das Aufschieben eines dunklen Schlauchs (z.B. Schrumpfschlauch) oder ein Isolierband.



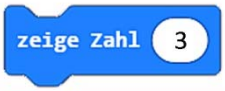
8. Die wichtigsten Blöcke für den Anfang:

- Aus dem **Register [Grundlagen]**:



Alle Blöcke (= Programme) innerhalb der **Start-Klammer** werden nur einmal beim **Start** ausgeführt.

Blöcke innerhalb dieser Klammer werden vom Micro:bit so lange als **Endlosschleife** ausgeführt, bis man den Strom abschaltet.



Dieser Block zeigt die eingefügte **Zahl** (hier 3) auf der **LED-Matrix** an.

Dieser Block zeigt den eingefügten **Text** (hier „Hallo“) am Micro:bit als **LED-Laufschrift** an.

Dieser Block zeigt das gewählte **Symbol** (hier „Herz“) als **LED-Symbol**. Mit der **Pfeilauswahl** erscheint eine Auswahl von 40 Symbolen.

Diese Funktion stellt die **Micro:bit LED-Anzeige** mit 25 LEDs dar.
Durch Anklicken der dunkelblauen Felder kann man die einzelnen LEDs an- und ausschalten und damit eigene Symbole erstellen.

Durch das Einfügen eines **Pause-Blocks** wird der Programmablauf für eine bestimmte Zeit (hier 100 ms) verzögert.
Die Angabe ist in **Millisekunden** (ms) → **1 Sekunde = 1000 ms**

- Aus dem **Register [Eingabe]**:

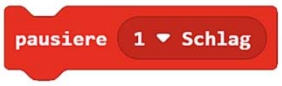
Bei einem Klick auf **Knopf A** des Micro:bit wird der Programmblock innerhalb der Klammer ausgeführt.
Mit der Pfeilauswahl kann man weitere Knöpfe aktivieren: **B** und **A+B**

- Aus dem **Register [Schleifen]**:

Alle Befehle (Blöcke) innerhalb des **Wiederhol-Blocks** werden in der gewählten Anzahl (hier 4-mal) wiederholt.

- Aus dem **Register [Musik]**:

Dieser Block erzeugt einen wählbaren **Ton** (hier Mittleres C) für eine **bestimmte Zeit** (hier 1 Schlag).



Dieser Block schaltet **Töne** für eine bestimmte Zeit (hier 1 Schlag) **aus**. An der **Pfeilauswahl** kann die „Schlagzeit“ verändert werden.

Micro:bit-Programme für den Winker

- Befestige den **Servo** für die **„Winker-Programme“** laut Aufbauanleitung mit zwei Spanplatten-schrauben 3 x 25 mm und dem Lochstab (4 L) an den **seitlichen Bohrungen** von Brett (B). Befestige die **„Hand“** vorerst noch nicht an der Servoachse.
- Programme mit dem Zusatz **„nur V2“** laufen nur an einem **Micro:bit V2**. Mit einem kleinen, **hochohmigen Lautsprecher** (Buzzer) an **GND** und **PIN 0** des Micro:bit können aber Sound-Programme auch am **Micro:bit V1** abgespielt werden.
- Die vorgeschlagenen **Namen** der **„hex-Dateien“** können natürlich geändert werden.

Programm 1: Begrüßung

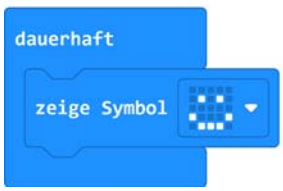
Öffne den **Makecode-Editor** (https://makecode.microbit.org/), klicke auf [Neues Projekt] und gib ihm den Namen „Hallo1“.

Vorgabe: Nach dem Einschalten soll der Micro:bit einmal den Lauftext **„Hallo!“** anzeigen und dann **dauerhaft** einen **»freundlichen Smiley«**

Programm-Code: (microbit-Hallo1.hex)

Im linken **Simulationsbereich** des **Makecode-Editors** sieht man bereits eine Vorschau, was das Programm tut.
Speichere, wie auf Seite 3 beschrieben, das fertige Programm auf dem Computer.
Verbinde den Micro:bit über ein Micro-USB-Kabel mit dem Computer und übertrage das Programm (microbit-Hallo1.hex) auf den Micro:bit.

Der Lauftext **„Hallo“** wird nur einmal angezeigt.



Solange der Micro:bit Strom hat, leuchtet der Smiley.

Weitere Aufgabe: Ändere den Text auf: **„Ich bin ein Microbit“** und das Symbol auf ein **„Herz“.**

Programm 2: Herzklopfen

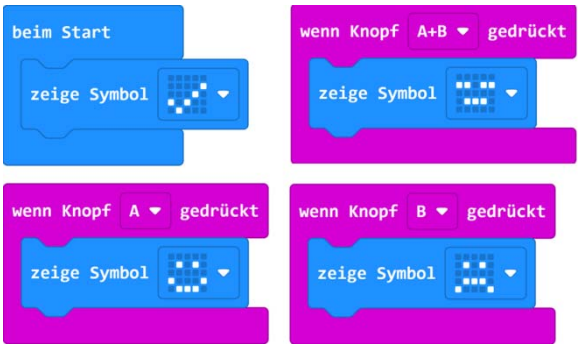
Vorgabe: Ein großes und ein kleines **»Herz-Symbol«** sollen jeweils **200 ms** lang abwechselnd aufleuchten.

Programm-Code: (microbit-Herzklopfen1.hex)

Der **Pause-Block** lässt das „Herz“ 200 ms lang aufleuchten.

Weitere Aufgabe: Ändere die **Herzfrequenz** durch längere Pause-Zeiten (z.B. 500 ms)

Programm 3: Knopf A und B



Vorgabe: Nach Drücken der **Knöpfe A, B und A+B** sollen verschiedene »**Smileys**« aufleuchten.

Programm-Code: (microbit-KnopfAB-1.hex)

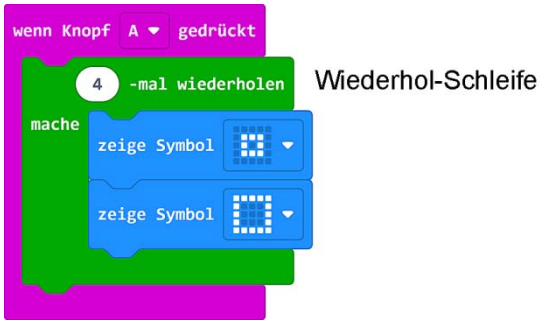
Weitere Aufgabe: Nach Drücken der **Knöpfe A, B** und **A+B** sollen die **Buchstaben „A“**, **„B“** und **„C“** angezeigt werden.

Programm 4: Wiederholung

Durch den Einsatz einer **»Wiederhol-Schleife«** aus dem **Register [Schleifen]** kann man die Anzahl der Wiederholungen von eingefügten Programmteilen genau festlegen.

Vorgabe: Ein kleines und ein großes **»Quadrat-Symbol«** sollen nach Betätigung von **Knopf A 4-mal blinken**.

Programm-Code: (microbit-Wiederholung1.hex)



Wiederhol-Schleife

Weitere Aufgaben:

- Ändere die **Anzahl der Wiederholungen**
- Setze an Stelle der Symbole **2 Zahlen** (z.B. 0 / 1)
- Ändere die Blinkfrequenz durch **»Pause-Blöcke«**

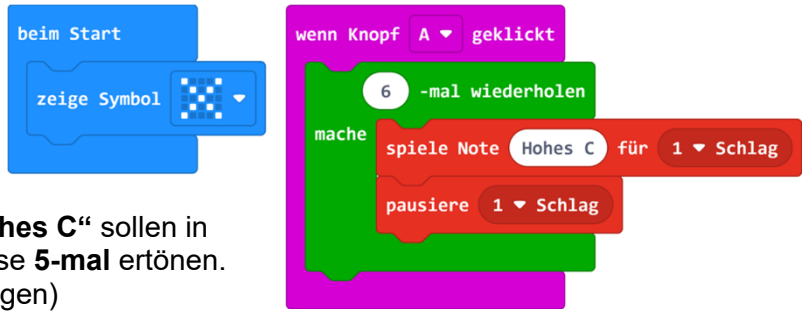
Programm 5: Pulston (nur V2)

Um **Töne** mit **Pausen** erzeugen zu können, benötigt man die **Blöcke »spiele Note … für …«** und **»pausiere …«** aus dem **Register [Musik]**.

Vorgabe: Beim Start soll ein **»X-Symbol«** aufleuchten. Nach Klicken von **Knopf A** soll die **Note „Hohes C“** (1 Schlag) mit **Pause** (1 Schlag) in einer **Wiederholschleife** (6-mal) erklingen.

Programm-Code:

(microbit-Pulston1.hex)



Weitere Aufgaben:

Die Noten **„Mittleres G“** und **„Hohes C“** sollen in der **Wiederholschleife** ohne Pause **5-mal** ertönen. (Folgetonhorn von Einsatzfahrzeugen)

Programm 6: Hand befestigen

Wähle in der Befehlsleiste das **Register [⊕ Erweiterungen]** und klicke auf das Bild **„Servos“**. In der Leiste erscheint jetzt das **Register [Servos]** mit dem **Block »setze Winkel von Servo an P0 auf 0°«**

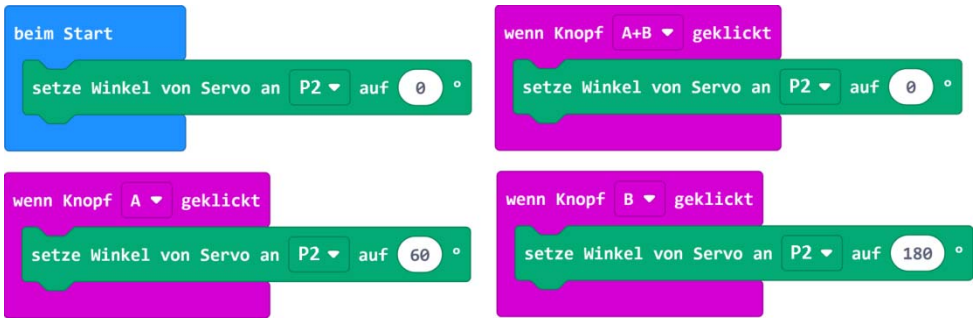
Vorgabe: Setze den **Winkel des Servos (an P2)** beim **Start** auf **30°**. Schiebe nun das **Ruderhorn** der **Hand** so auf die Servoachse, dass die Finger gerade nach oben zeigen. Mit der beiliegenden Schraube kann die Hand schließlich an der Achse befestigt werden.

Programm-Code: (microbit-Hand1.hex)

Programm 7: Servo-Übungen

Vorgabe: Setze den **Winkel des Servos (an P2)** beim **Start** auf „**0°**“. Durch Drücken der **Knöpfe A** und **B** soll der Servo auf **bestimmte Winkel** gesetzt werden:
A → 60°, B → 180°, A+B → 0°

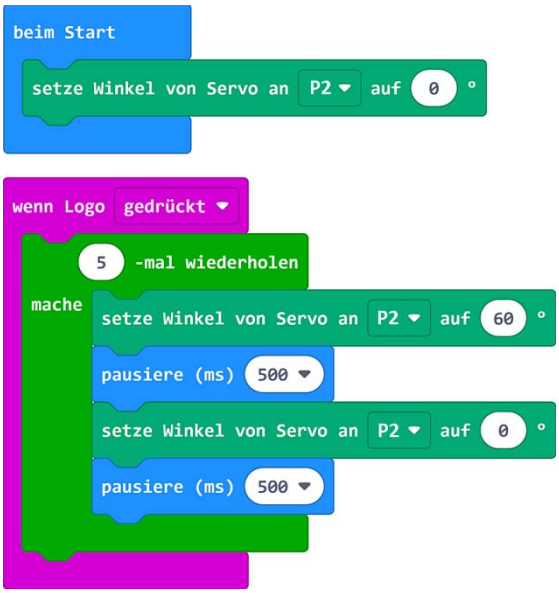
Programm-Code: (microbit-Servo1.hex)



Programm 8: Winken mit Logo

Vorgabe: Setze den **Winkel des Servos (an P2)** beim **Start** auf „**0°**“. Nach **Berühren** des **Logos** oberhalb der LEDs soll die **Hand 5-mal** winken.

Programm-Code: (microbit-Logo-Winken1.hex)



Weitere Aufgabe:

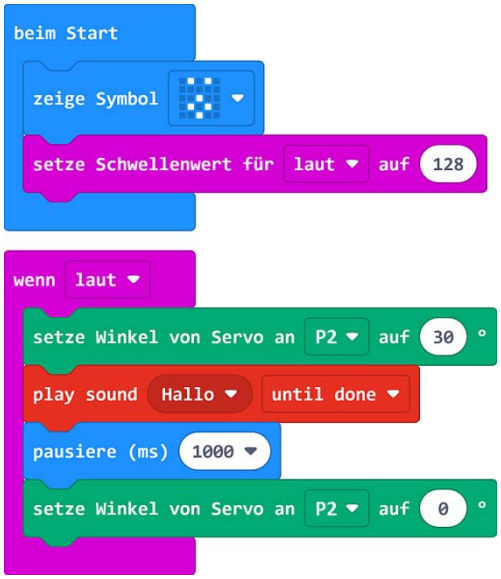
Die Hand soll nach einem Klick auf **Knopf A 8-mal** winken.

Programm 9: Klatsch-Schalter (nur V2)

Der **Micro:bit V2** ist mit einem **Mikrofon** ausgestattet, das zur Steuerung von Programmen genutzt werden kann. Die **Ansprechempfindlichkeit** (Schwellenwert) reicht von **0 - 255**.

Vorgabe: Setze beim **Start** einen **»erstaunten Smiley«** und stelle den **Schwellenwert** auf **128**.
Der **Block »wenn laut«** vom **Register [Eingabe]** soll auf ein **Klatsch-Geräusch** reagieren. Die **Hand** soll dann „**30°**“ bewegt und nach dem **»Sound „Hallo“«** aus dem **Register [Musik]** wieder zurückgesetzt werden.

Programm-Code: (microbit-Klatschen1.hex)



Weitere Aufgaben:

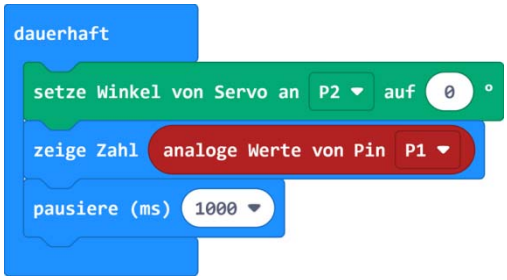
Ändere den **Winkel** des **Servos** und lass die **Melodie „Entertainer“** aus dem **Register [Musik]** erklingen.

Programm 10: Analoge Werte des Infrarot-Sensors anzeigen

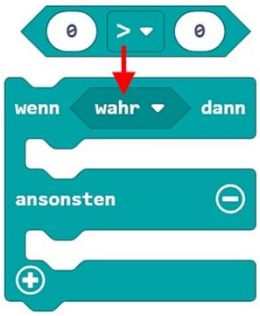
Sensoren erzeugen an den Micro:bit-Eingängen **analoge Werte** zwischen **0** und **1023**.
Bei **digitalen Sensoren** gibt es nur zwei Werte: **1 = wahr** und **0 = falsch**
Mit dem **Potentiometer** am **IR-Sensor** kann man die Empfindlichkeit des Sensors einstellen.

Vorgabe: Der Micro:bit soll den **Servo** auf „**0°**“ stellen und dauerhaft den **analogen Wert** des **IR-Sensors** an **P1** anzeigen.

Programm-Code: (microbit-IR-Wert1.hex)



Programm 11: Näherungssensor

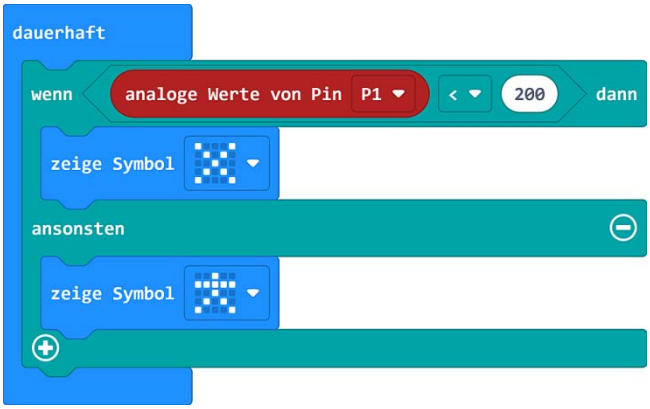


Für das Programm benötigen wir einen »**Wenn-Block« (wenn/dann)** mit Verzweigung und einen sechseckigen »**Vergleichs-Block (0 > 0)** aus dem **Register** **[Logik]**.
Wenn die **Bedingung wahr** ist (**analoger Wert von P1 < 200**), dann wird der obere Programmteil ausgeführt, **ansonsten** der untere.
Am ⊕ kann der »**Wenn-Block«** erweitert, am ⊖ reduziert werden.

Der **Block »analoge Werte von P1«** befindet sich in den **Registern** **[Fortgeschritten]** **+** **[Pins]**

Vorgabe: Wenn sich jemand dem **IR-Sensor** nähert, soll ein **»X-Symbol«** aufleuchten - ansonsten ein **»Figur-Symbol«**.

Programm-Code: (microbit-Naeherung1.hex)

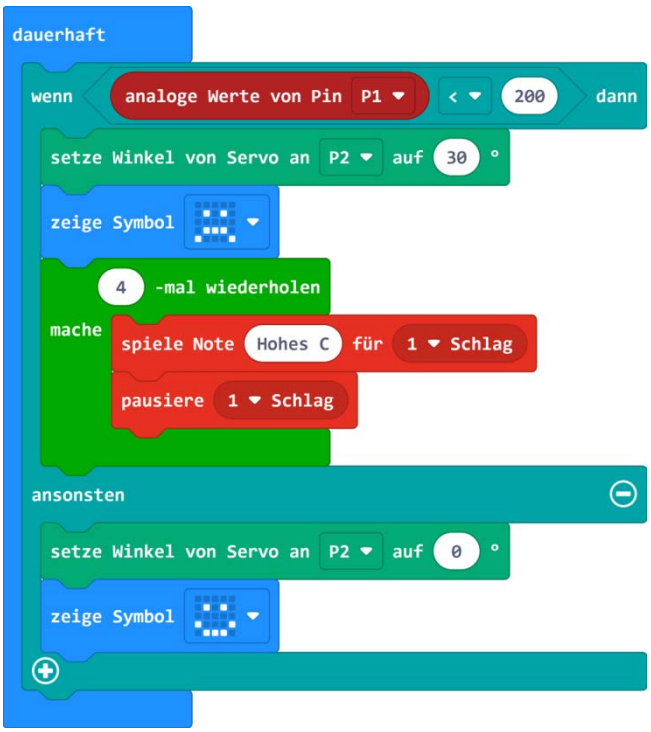


Weitere Aufgabe: Anstelle des **»X-Symbols«** soll der **Text „Stopp!“** angezeigt werden.

Programm 12: IR-Alarmanlage (nur V2)

Vorgabe: Nähert sich jemand dem IR-Sensor, soll die **Hand nach oben** zeigen, ein **»böser Smiley«** aufleuchten und ein **Warnton 4-mal** erklingen. Ansonsten soll die **Hand auf „0°“** gesetzt und ein **»fröhlicher Smiley«** gezeigt werden.

Programm-Code: (microbit-IR-Alarm1.hex)



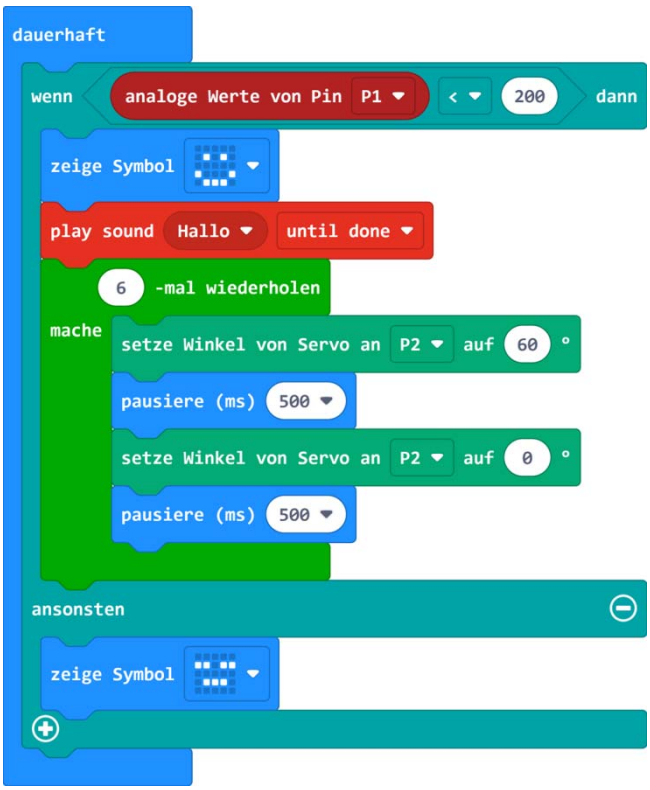
Weitere Aufgaben: Die Hand soll statt „30°“ **auf „120°“** gestellt werden.
Die Noten **„Mittleres G“** und **„Hohes C“** sollen in der **Wiederholschleife** ohne Pause **5-mal** ertönen.

Programm 13: Infrarot-Winker (nur V2)

Bei diesem Programm freut sich der Winker, wenn jemand näher kommt.

Vorgabe: Wenn sich jemand dem **IR-Sensor** nähert, soll ein **»froher Smiley«** aufleuchten, ein **„Hallo“** erklingen und die **Hand 6-mal** winken. Ansonsten soll ein **»gelangweilter Smiley«** sichtbar sein.

Programm-Code: (microbit-IR-Winker1.hex)



Weitere Aufgaben: Ändere den **Sound** und die **Anzahl** der Winkbewegungen.

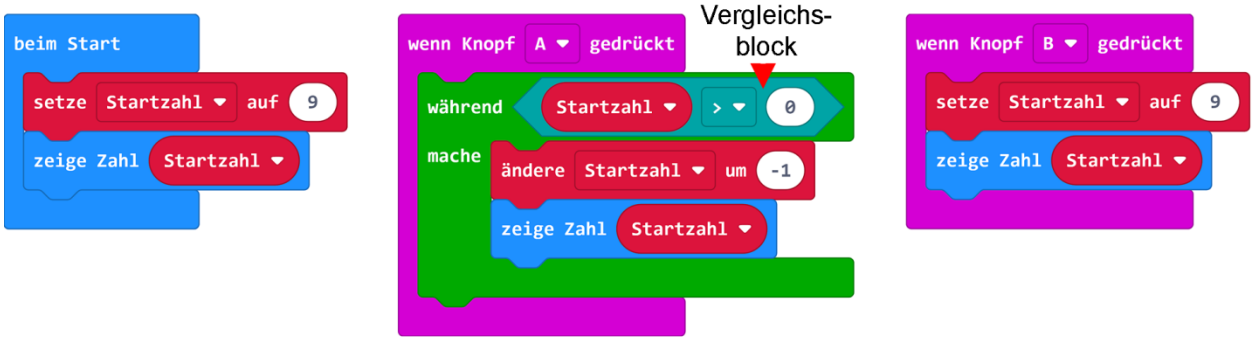
Programm 14: Countdown

Für weitere Programme benötigen wir **Variablen. Variablen** sind quasi **„Behälter“**, die Zahlen und Werte für ein laufendes Programm zwischenspeichern können.
Öffne das **Register [Variablen]**, klicke auf **»Erstelle eine Variable«**, gib der **Variablen** den Namen **„Startzahl“** und bestätige mit **»OK«**. Es erscheinen dann drei neue, **rote Blöcke**.

Für die **Bedingung** in der **»Während-Schleife«** benötigt man für das folgende Programm einen sechseckigen **»Vergleichs-Block« (0 > 0)** aus dem **Register [Logik]**.

Vorgabe: Erstelle eine **Variable** mit dem **Namen »Startzahl«** und setze sie beim **Start** auf **„9“**. Während die **Startzahl** größer (>) als „0“ ist, soll nach Klicken von **Knopf A** die Startzahl jeweils um **„1“ vermindert** und **angezeigt** werden. Mit **Knopf B** wird die **Startzahl** wieder auf **„9“** gesetzt.

Programm-Code: (microbit-Countdown1.hex)



Weitere Aufgaben:

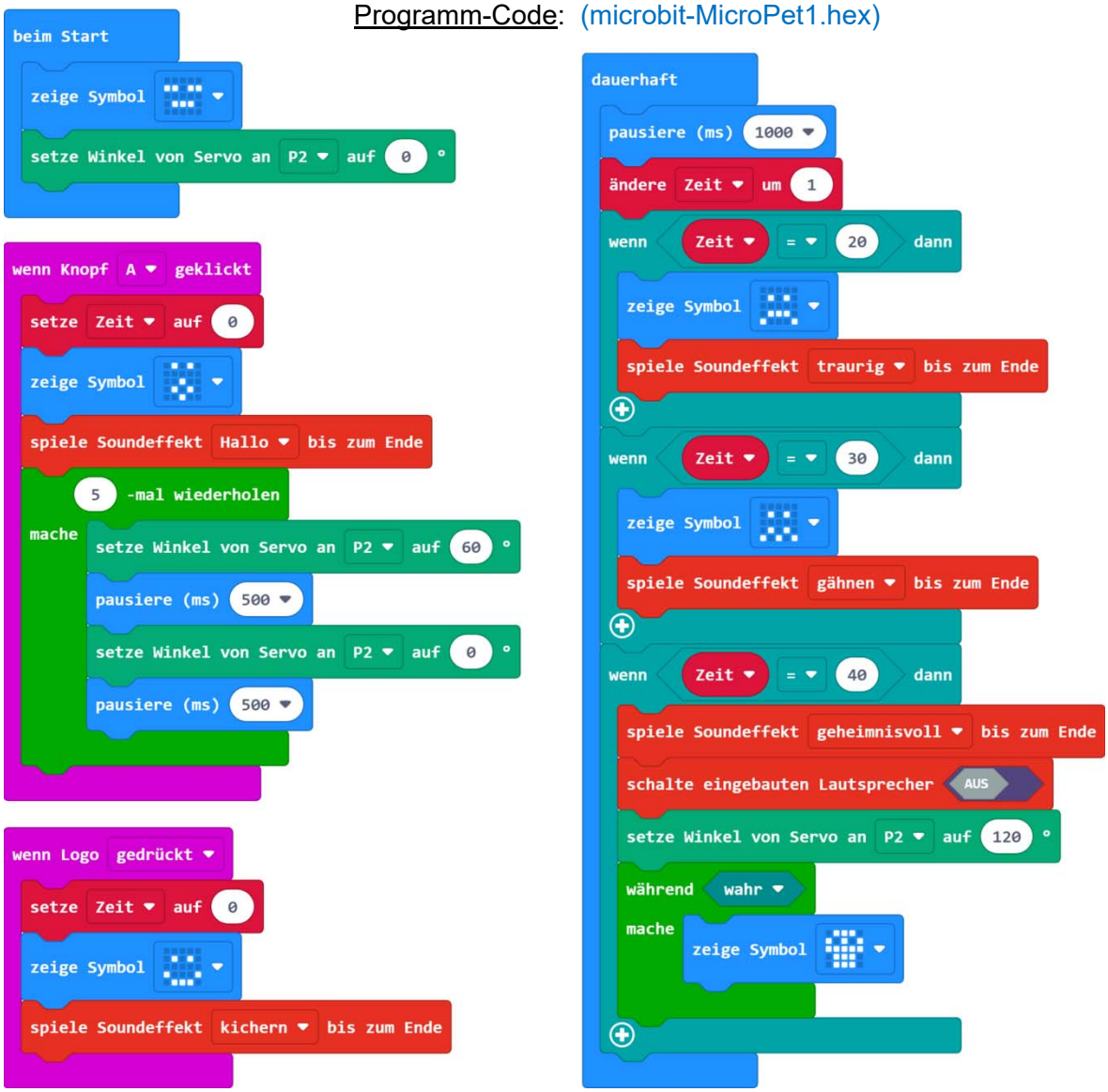
Setze die Variable **»Startzahl«** beim **Start** auf **„0“**. Während die **Startzahl kleiner (<) als „9“** ist, soll nach Klicken von **Knopf A** die Startzahl jeweils um **„1“ erhöht** und **angezeigt** werden. Mit **Knopf B** soll die **Startzahl** wieder auf **„0“** gesetzt werden.

Programm 15: MicroPet (nur V2)

„**Tamagotchis**“ waren in den 1990er-Jahren ein sehr beliebtes, eiförmiges Spielzeug. Es war eine Art elektronisches Haustier, um das man sich kümmern musste. Wenn man nicht alle paar Stunden gewisse Tasten drückte, dann verstarb es.
Unser **MicroPet** soll ähnlich funktionieren: Je länger das MicroPet keine **Zuneigung** bekommt (Logo berühren oder Knopf A drücken), desto trauriger wird es und nach 40 Sekunden ohne Zuneigung verabschiedet es sich. (Hinweis: Am **»Pause-Block«** kann man die Zeit erhöhen!)

Vorgabe: Erstelle eine **Variable** mit dem Namen **„Zeit“**. Setze beim **Start** den **Servo** auf **„0°“** und zeige einen **»gelangweilten Smiley«**.
Mit einem Druck auf **Knopf A** soll die **Variable »Zeit«** auf **„0“** gestellt, das **Symbol »erstaunter Smiley«** gezeigt und der **Sound „Hallo“** abgespielt werden. Weiters soll das MicroPet dann **5-mal winken**.
Beim Berühren des **Logos** soll die **Variable »Zeit«** ebenfalls auf **„0“** gestellt, das **Symbol »froher Smiley«** gezeigt und der **Sound „kichern“** abgespielt werden.
Das eigentliche Programm in der **Schleife »dauerhaft«** soll mit einer **Pause** (1000 ms) beginnen und die **Variable »Zeit«** bei jedem Programmdurchlauf um **„1“** erhöhen.
In drei **»Wenn-Blöcken«** soll festgelegt werden, wie das MicroPet nach einer **»Zeit«** von **„20“, „30“ und „40“** reagieren soll.
Wenn bis **„40“ keine Zuneigung** (Knopf A oder Logo) erfolgt, soll der **Sound »geheimnisvoll«** gespielt, der **Lautsprecher** ausgeschaltet und der **Servo auf „120°“** gesetzt werden. In einer **»während-Schleife« soll** zuletzt ein **»Totenkopf-Symbol«** erscheinen.
Nur der **„Reset-Knopf“** am Micro:bit kann es jetzt wieder zu neuem Leben erwecken.

Programm-Code: (microbit-MicroPet1.hex)



Micro:bit-Programme für die Schranke

- Befestige den **Servo** für die **„Schranken-Programme“** laut Aufbauanleitung mit zwei Spanplattenschrauben 3 x 25 mm und dem Lochstab (4 L) an den **rückseitigen Bohrungen** von Brett (B). Befestige die **„Schranke“** vorerst noch nicht an der Servoachse.

Programm 16: Schranke befestigen

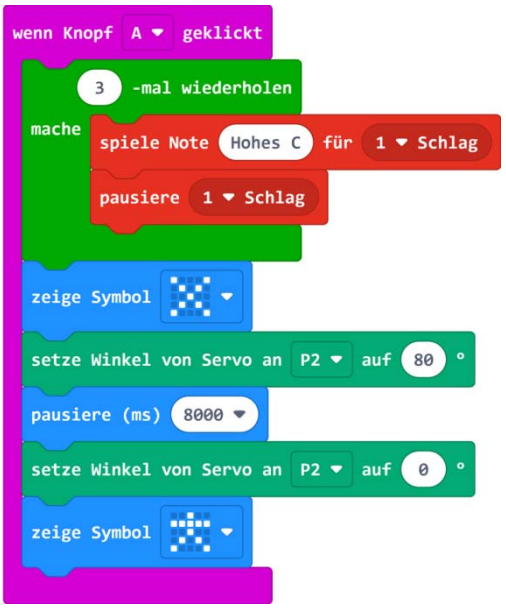
Vorgabe: Setze den **Winkel des Servos (an P2)** mit **Knopf A** auf „**0°**“ und **Knopf B** auf ca. „80**°**“ Befestige dann die **Schranke** bei „**80°**“ mittels Ruderhorn **waagrecht** am Servo.

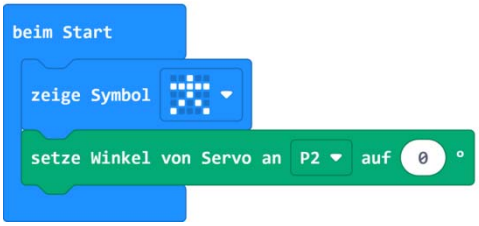
Programm-Code:
(microbit-Schranke1.hex)

Programm 17: Schranke schließt mit Warnton (nur V2)

Vorgabe: Beim **Start** soll ein **»Fußgänger-Symbol«** leuchten und die Schranke auf **„0°“** gesetzt werden. Nach Klicken von **Knopf A** sollen drei **Warntöne** erklingen, ein **»X-Symbol«** erscheinen und die Schranke schließen (80°). Nach **8000 ms** soll die Schranke wieder hochgehen (0°) und das **»Fußgänger-Symbol«** wieder leuchten.

Programm-Code: (microbit-Schranke-Ton1.hex)



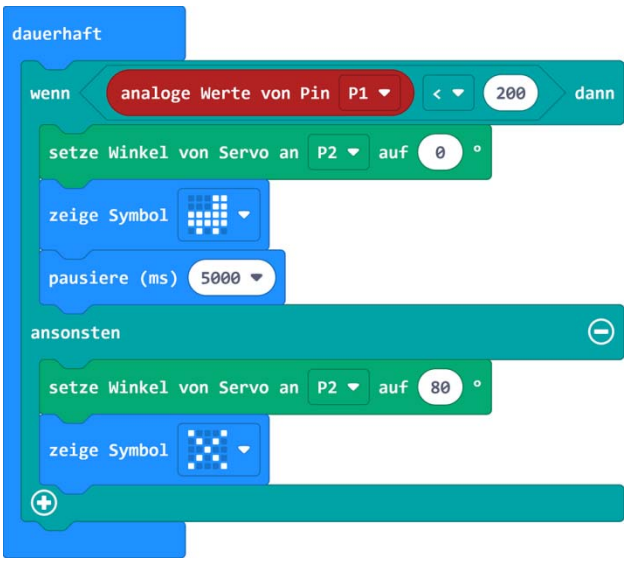


Weitere Aufgaben: Beim **Start** soll die Schranke geschlossen werden (80°). Nach Berühren des **Logos** soll die Schranke hochgehen (0°). Nach **10 Sekunden** soll ein **Ton** (2 Schlag) erklingen und die Schranke wieder geschlossen werden (80°).

Programm 18: Infrarot-Schranke

Vorgabe: Wenn der **analoge Wert** des **Infrarot-Sensors** kleiner (<) als „**200**“ ist, soll die **Schranke hochgehen** (0°) und ein **»Fahrzeug-Symbol«** leuchten. Ansonsten soll die Schranke auf **„80°“** gesetzt und ein **»X-Symbol«** angezeigt werden.

Programm-Code: (microbit-Schranke+IR1.hex)



Weitere Aufgabe: Wenn der **analoge Wert** des **Infrarot-Sensors** kleiner (<) als „**200**“ ist, soll die **Schranke** niedergehen (80°) - ansonsten aber hochgehen (0°).

Programm 19: Index-Schleife

»**Index-Schleifen**« aus dem **Register** [Schleifen] funktionieren wie »**Wiederhol-Schleifen**« mit dem Vorteil, dass sie die **aktuelle Wiederholzahl** anzeigen und in der abrufbaren **Variablen »Index«** speichern können.

Vorgabe: Nach Betätigung von **Knopf A** sollen mittels **»Index-Schleife«** die **Zahlen von 0 - 12** angezeigt werden.

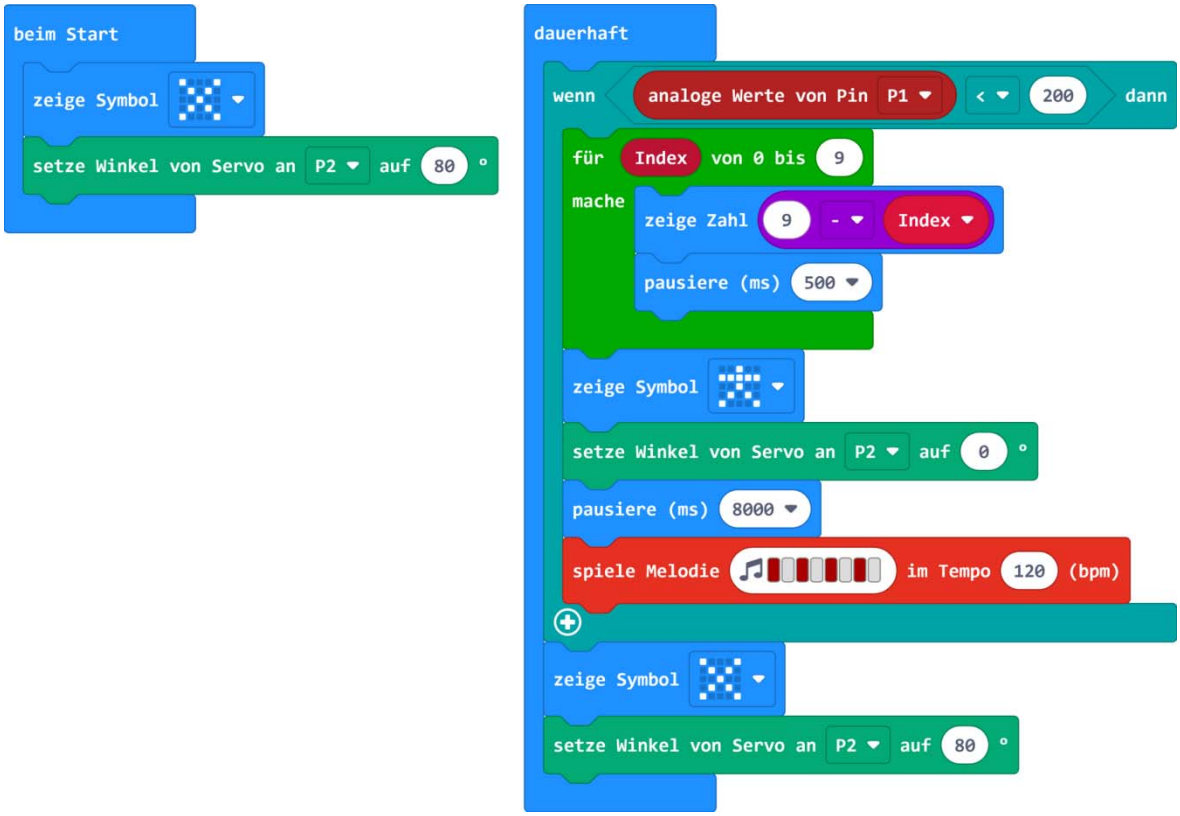
Programm-Code: (microbit-Index-Schleife1.hex)

Programm 20: IR-Schranke öffnet nach Countdown (nur V2)

Für dieses Programm benötigt man eine **»Index-Schleife«** aus dem **Register** [Schleifen] und einen **»Rechenblock „0 - 0“«** aus dem **Register** [Mathematik].

Vorgabe: Beim **Start** soll ein **»X-Symbol«** leuchten und die Schranke auf „**80°**“ gesetzt werden. Wenn der **analoge Wert** des **Infrarot-Sensors** kleiner (<) als „**200**“ ist, soll eine **»Index-Schleife«** von **„9“** bis **„0“** zählen, das **»Fußgänger-Symbol«** soll aufleuchten und die Schranke hochgehen (**0°**). Nach **8 Sekunden** soll eine **Melodie** (Pulston) ertönen, ein **»X-Symbol«** erscheinen und die Schranke soll geschlossen werden (**80°**).

Programm-Code: (microbit-Schranke+IR2.hex)



Nachwort:

Mit dieser Programmieranleitung haben wir versucht, die wichtigsten Grundfunktionen dieses faszinierenden Minicomputers mit einfachen bis leicht fortgeschrittenen Programmideen an einem selbst gebauten Modell mit Servo und Infrarot-Sensor zu veranschaulichen und zu festigen. Die gezeigten Programme sollen als Basis für weiterführende eigene Programmideen dienen.