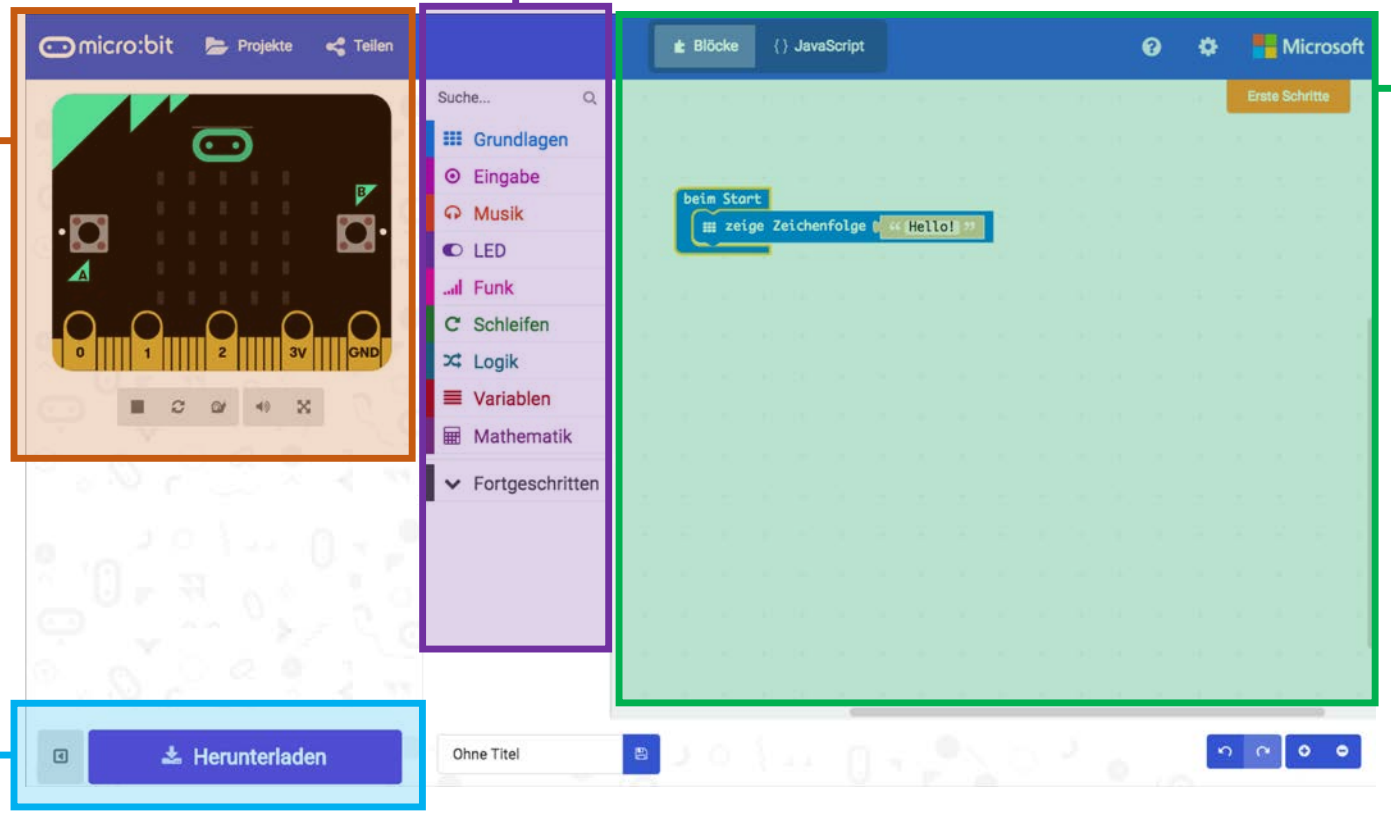
# Die Oberfläche von makecode.microbit.org/beta

**Code-Repertoire**
Kategorien mit den zur Verfügung stehenden Programmblöcke

**micro:bit Simulation**
Programmausgabe wird simuliert. Der Simulator reagiert auf Eingaben (z.B. Knöpfe)




**Programmcode**
Effektiver Programmcode. Die Codeblöcke werden aus dem Code-Repertoire hierhin gezogen

**Download Maschinencode**
Heruntergeladene Datei wird im Explorer/Finder via USB auf die Platine kopiert

# Aufbau: Mini - Aufgabenreihe

## TASKS – Einstieg

Einfache, kurze Tasks
Fokus auf Input / Output, Microbit kennenlernen

1 – 6

## TASKS – Erweiterung

Task, welche eine Kombination von Elementen erfordern
Fokus auf Logik, einfache Algorithmik, Programmkonzepte

7 – 10

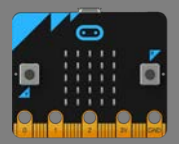
## Problemstellung

Fokus auf Entwicklung eigener Umsetzungsvarianten / Problemlösestrategien

# TASKS: Einstieg (Input / Output)

## TASK I
Der Welt Hallo sagen

**Hardware**

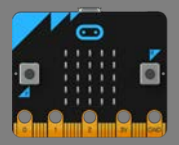


**Programmlogik**

Bedingte Anweisung

## TASK II
Pin-Kontakt

**Hardware**



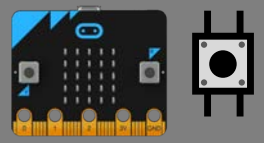
Verbindungskabel

**Programmlogik**

Schleife

## TASK III
Externer Schalter

**Hardware**



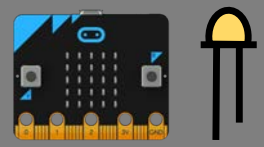
Verbindungskabel

**Programmlogik**

Schleife

## TASK IV
Diode ansteuern

**Hardware**



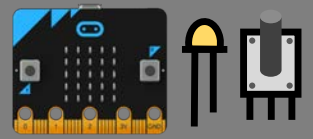
Verbindungskabel

**Programmlogik**

Bedingte Anweisung

## TASK V
Lichtregler

**Hardware**

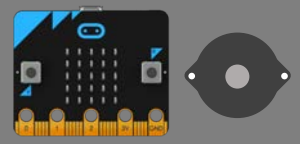


Verbindungskabel

**Programmlogik**

## TASK VI
Buzzer

**Hardware**



Verbindungskabel

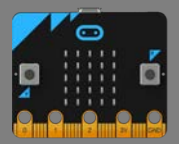
**Programmlogik**

Bedingte Anweisung

# TASKS: Erweiterung (Programmlogik)

## TASK VII
Zähler

**Hardware**

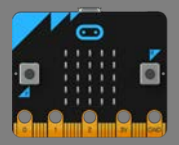


**Programmlogik**

- Bedingte Anweisung
- Arithmetische Operationen
- Variablen

## TASK VIII
Kompass

**Hardware**

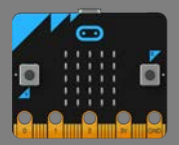


**Programmlogik**

- Bedingte Anweisung
- Logische Operationen
- Variablen

## TASK ZUSATZ
Game Astroids

**Hardware**

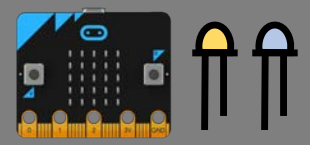


**Programmlogik**

- Bedingte Anweisung
- Schleife
- Arithmetische Operationen
- Variablen

## TASK IX
Sirene

**Hardware**



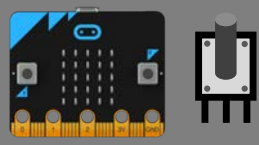
Verbindungskabel

**Programmlogik**

- Bedingte Anweisung
- Schleife
- Variablen

## TASK X
Visueller Regler

**Hardware**



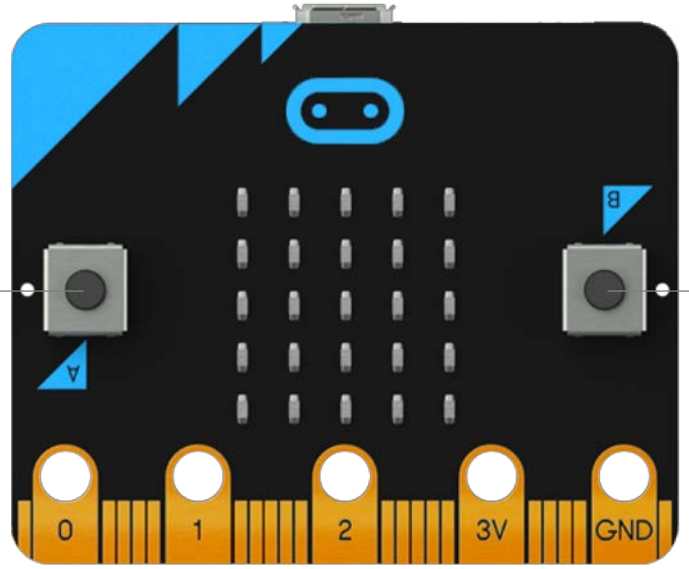
Verbindungskabel

**Programmlogik**

- Bedingte Anweisung
- Schleife
- Arithmetische Operationen
- Variablen

# **TASK I:** Der Welt Hallo Sagen

**Knopf A:**
Textanzeige LED:
**Hallo Welt**




**Knopf B:**
Smiley anzeigen

## TASK

- Nach Knopfdruck Displayanzeige aktivieren / wechseln

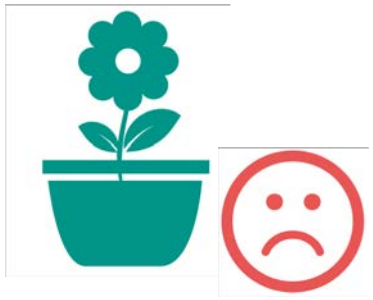
## EINSATZ VON

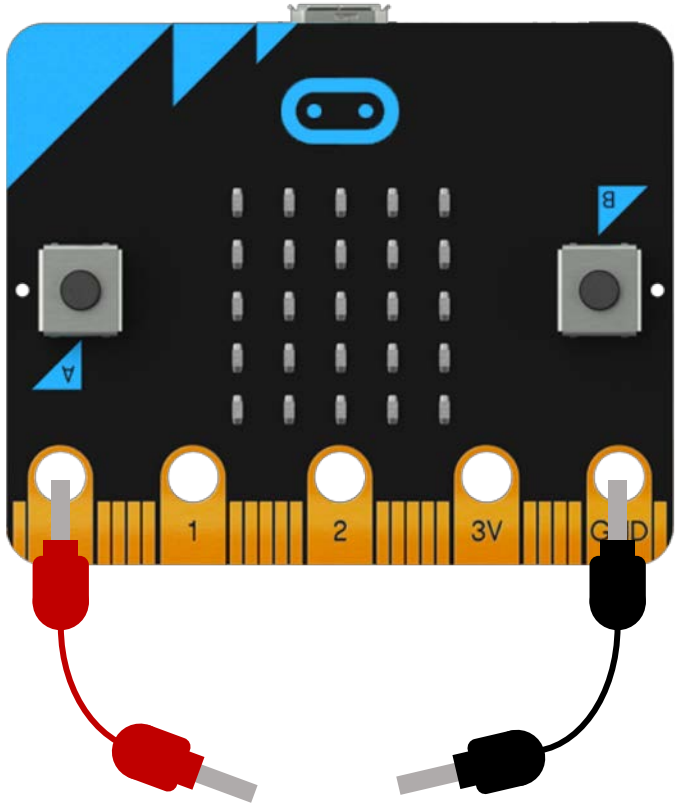
- Integrierte Buttons A & B
- Integrierte LED-Matrix
- Bedingter Anweisung (Knopf)

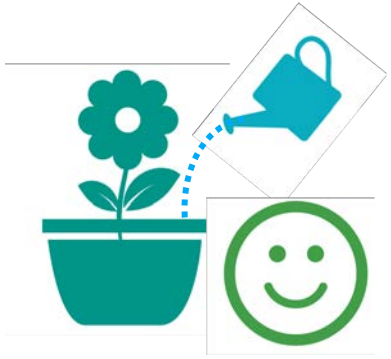
### Benötigte Blöcke

zeige Zeichenfolge [ ]
zeige LEDs
wenn Knopf [ ] gedrückt
halte Animation an

# **TASK II:** Pin-Kontakt (Bewässerungsmelder)





**Kein Kontakt**
Zeige Anti-Smiley

**Kontakt**
Zeige Smiley

## TASK

- Durch den Pinkontakt ändert sich die Displayanzeige von einem Anti-Smiley zu einem Smiley

## EINSATZ VON

- PIN-Kontakt
- Schleife

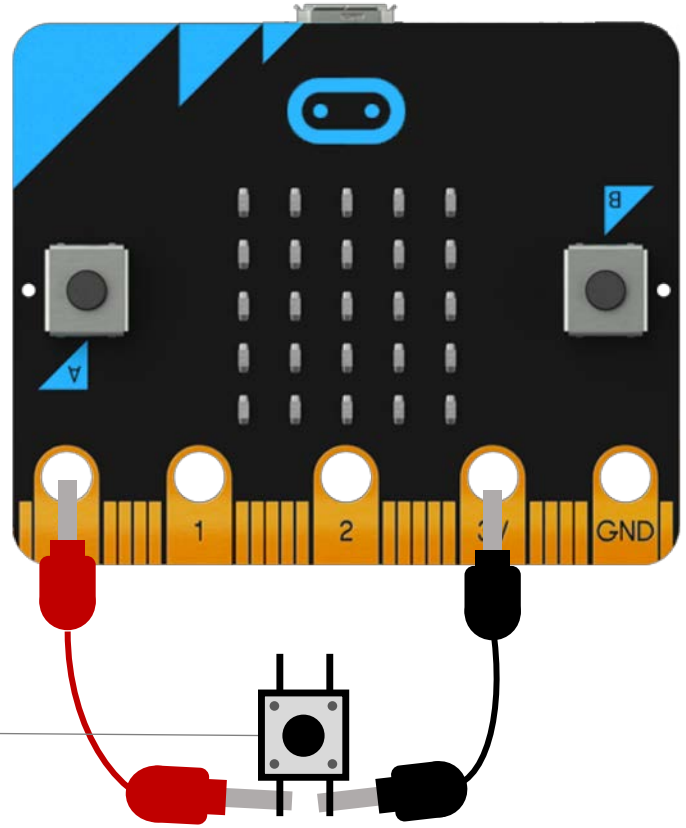
### Benötigte Blöcke

dauerhaft [ ] | zeige LEDs

pin [ ] ist gedrückt

während [ ] mache [ ]

# **TASK III:** Externer Schalter (digitaler Input)




**Externer Schalter:**

```
Alle LEDs leuchten
solange der Schalter
gedrückt wird
```

## TASK

- Per externer Knopfdruck LED-Matrix ein- und ausschalten

## EINSATZ VON

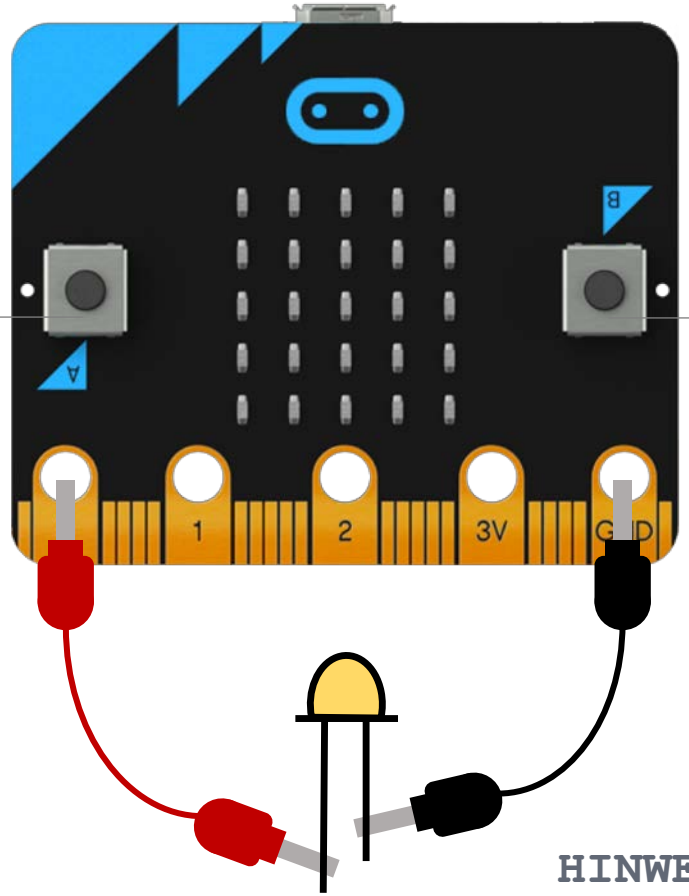
- Integrierte Buttons A & B
- Externer Schalter (digitaler Input)
- Schleife

## Benötigte Blöcke

dauerhaft [ ] | zeige LEDs | Bildschirminhalt löschen | während [ ] mache [ ] | [ ] = [ ] | digitale von Pin [ ]

# **TASK IV:** Diode ansteuern (digitaler Output)

**Knopf A:**
Diode einschalten




**Knopf B:**
Diode ausschalten

**HINWEIS:**
Das kürzere Bein wird an GND angeschlossen

## TASK

- Per Knopfdruck LED ein- und ausschalten

## EINSATZ VON

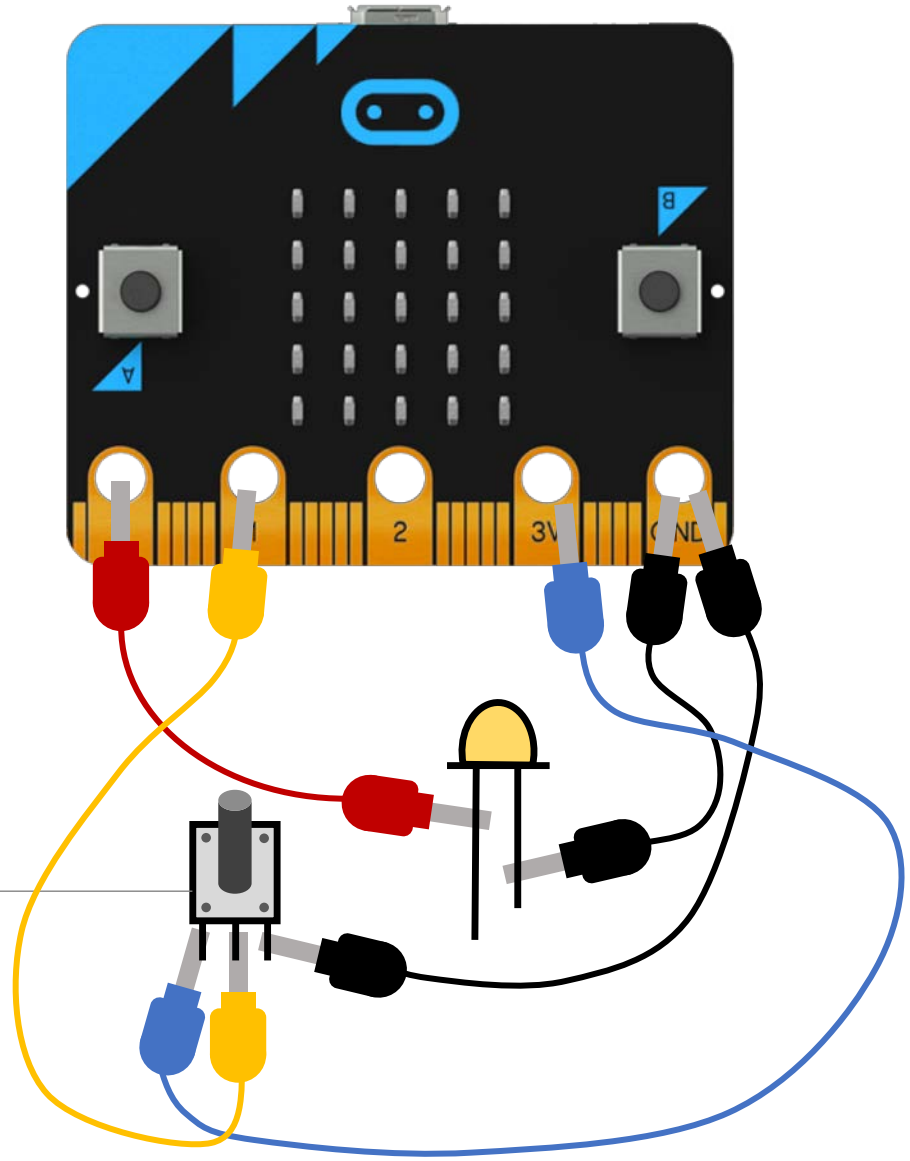
- Integrierte Buttons A & B
- Leuchtdiode (digitaler Output)

## Benötigte Blöcke

wenn Knopf [ ] gedrückt

Schreibe digitalen Wert von Pin [ ] auf [ ]

# **TASK V:** Lichtregler (analoger Input/Output)




**Drehung Regler:**
Steuert die Helligkeit der Leuchtdiode

**HINWEIS:**
Das kürzere Bein der Leuchtdiode wird an GND angeschlossen

## TASK

- Drehung des Reglers steuert die Helligkeit der Leuchtdiode (Outputwert des Reglers = Inputwert der Diode)

## EINSATZ VON

- Potentiometer (analoger Input)
- Leuchtdiode (analoger Output)

### Benötigte Blöcke

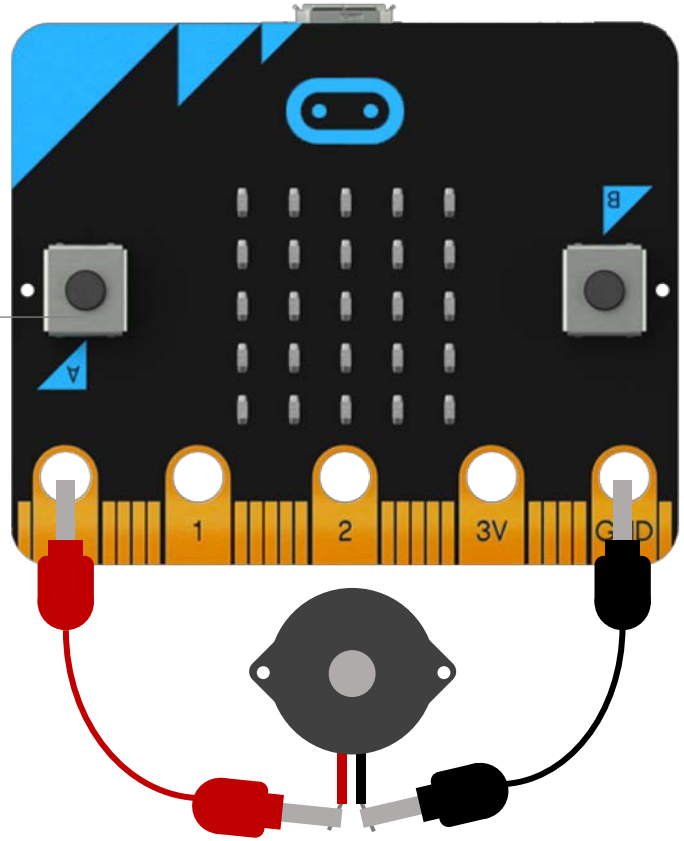
dauerhaft [ ]

schreibe analogen Pin [ ] auf [ ]

analoge Werte von Pin [ ]

# **TASK VI:** Musik

**Knopf A:**
Blues abspielen




## TASK

› Per Knopfdruck Musik abspielen

## EINSATZ VON

› Buzzer

### Benötigte Blöcke

wenn Knopf [ ] gedrückt

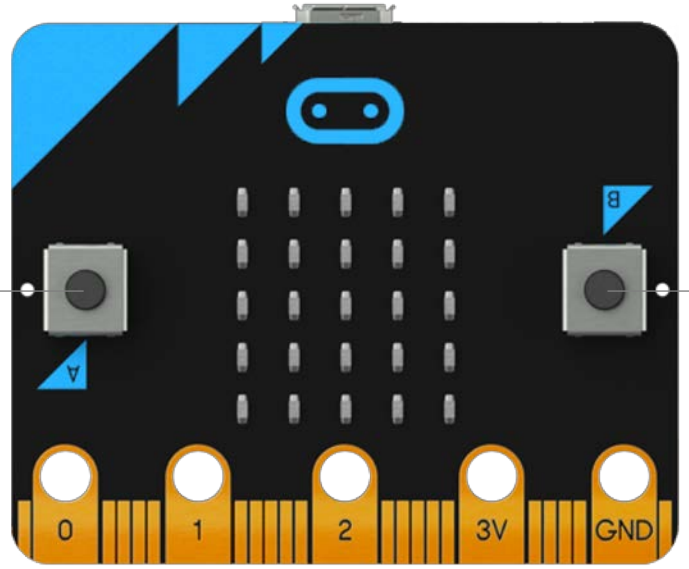
Beginne Melodie [ ] Wiederhole [ ]

# TASK VII: Zähler

**START:**
Zahl in Display **auf 0**

**Knopf A:**
Zahl in Display **-1**

**Zusatz:**
Minimalwert = **0**




**Knopf B:**
Zahl in Display **+1**

**Zusatz:**
Maximalwert = **9**

**SHAKE:**
Zahl in Display = **Zufallszahl**

## TASK

- Eine Variable mit dem Namen «Nummer» wird generiert (Variablen → neue Variable anlegen)
- Je nach Knopfdruck wird der Wert der Variable erhöht oder reduziert
- Ein «Shake» definiert die Variable zufällig neu (1–9)

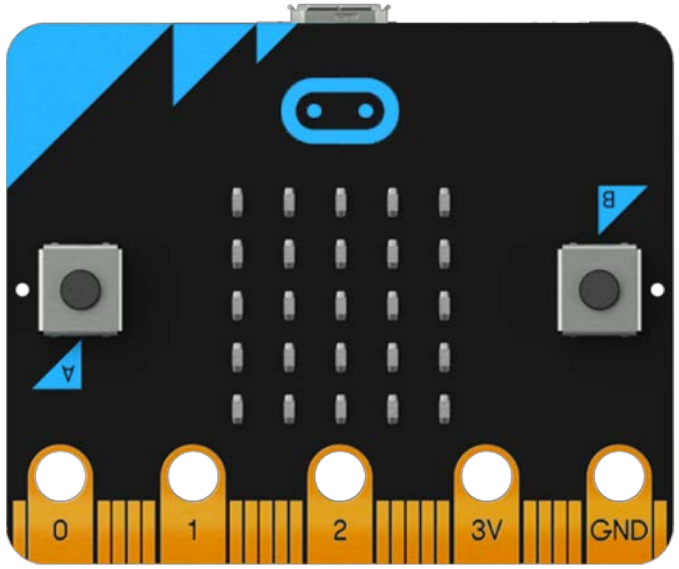
## EINSATZ VON

- Integrierter Beschleunigungssensor
- Variablen
- Bedingte Anweisungen (Wenn – Dann)
- Zufallszahl

## Benötigte Blöcke

# TASK VIII: Kompass

**Drehen der Platine:**
Je nach Richtung **N**, **O**, **S** oder **W** anzeigen




**ACHTUNG:**
Zu Beginn wird der Kompass des micro:bit kalibriert. Dazu drehen sie die Platine bis alle LEDs aufleuchten

## TASK

- Je nach Ausrichtung der Platine wird die Himmelsrichtung angezeigt
- Die Ausrichtung wird in Grad gemessen und wird in einer Variable gespeichert
- 0 Grad entspricht Norden. «N» soll entsprechend zwischen 315° und 360° sowie zwischen 0° und 45° angezeigt werden

## INHALTE

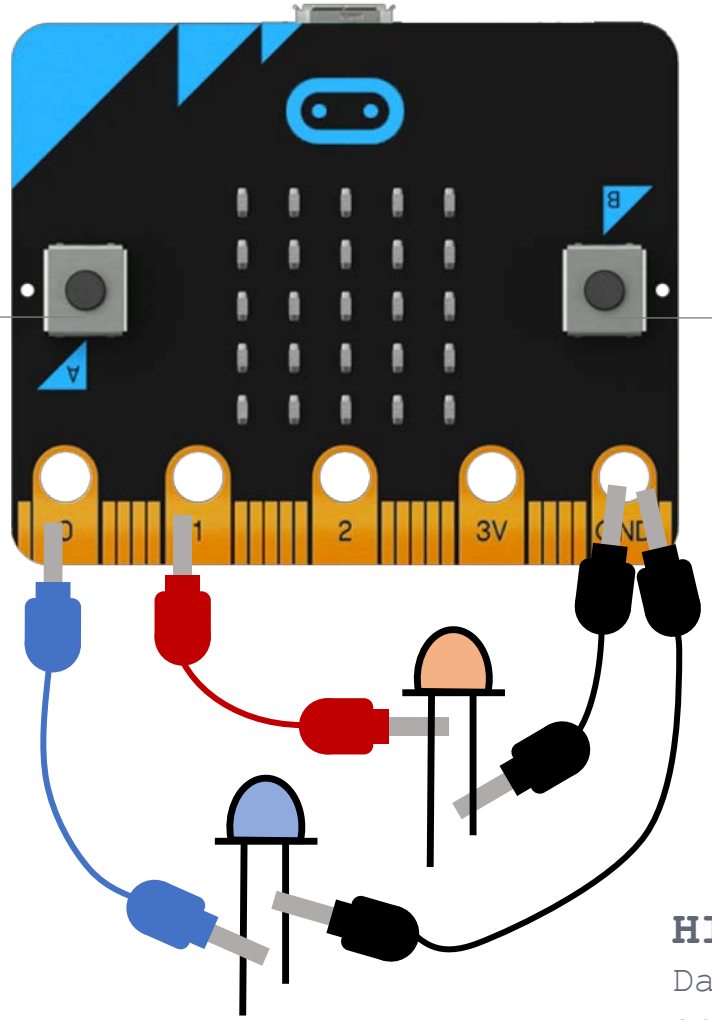
- Integrierter Kompass
- Variablen
- Erweiterte Bedingte Anweisung (Wenn – Dann – Ansonsten)
- Logische Operatoren

## Benötigte Blöcke

dauerhaft [ ]
zeige Zeichenfolge [ ]
Kompassausrichtung [ ]
wenn [ ] dann ansonsten [ ]
[ ] und [ ]
[ ] oder [ ]
ändere [ *Variable* ] auf [ ]

# TASK IX: Sirene

**Knopf A:**
Sirene an




**Knopf B:**
Sirene aus

**HINWEIS:**
Das kürzere Bein wird an GND angeschlossen

## TASK

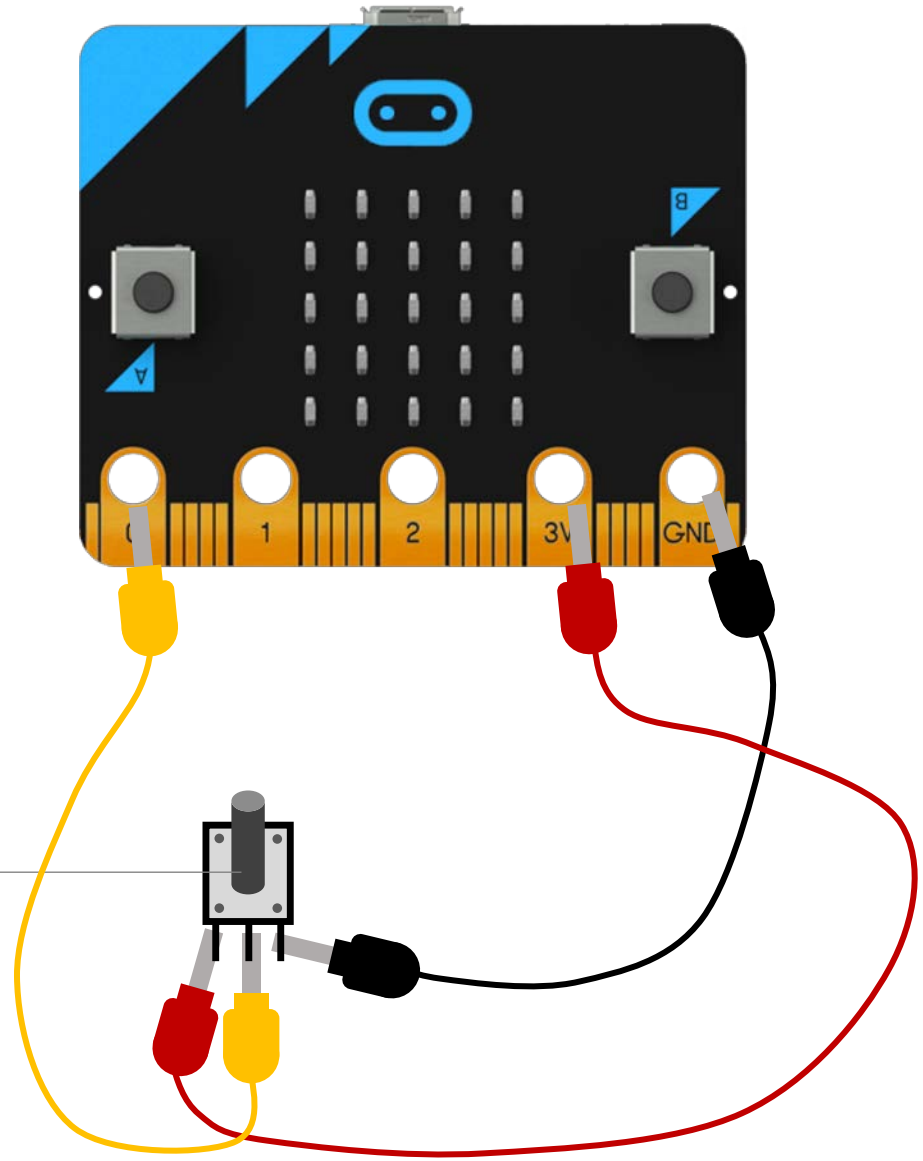
- Per Knopfdruck Sirene ein- und ausschalten
- Sirene bedeutet, dass die Leuchtdioden abwechselnd blinken

## EINSATZ VON

- Integrierte Buttons A & B
- Leuchtdiode (digitaler Output)
- Schleifen
- Variablen

## Benötigte Blöcke

# TASK X: Visueller Regler




**Drehung Regler:**

Steuert LED Matrix
(LED 1 – 25)

## TASK

- Durch Drehung des Reglers wird die LED-Matrix gesteuert
- Regler ganz links: nur 1 LED oben links aktiv
- Regler ganz rechts: Alle LEDs aktiv

## EINSATZ VON

- Potentiometer (analoger Input)
- LED Einzeloutput
- Schleifen
- Bedingten Anweisungen
- Variablen
- Arithmetische Operationen

## Benötigte Blöcke

Start | dauerhaft | [ ] +,-,x,÷ [ ]

[ ] -mal wiederholen | [ ] =,<,> [ ]

wenn [ ] dann ansonsten [ ]

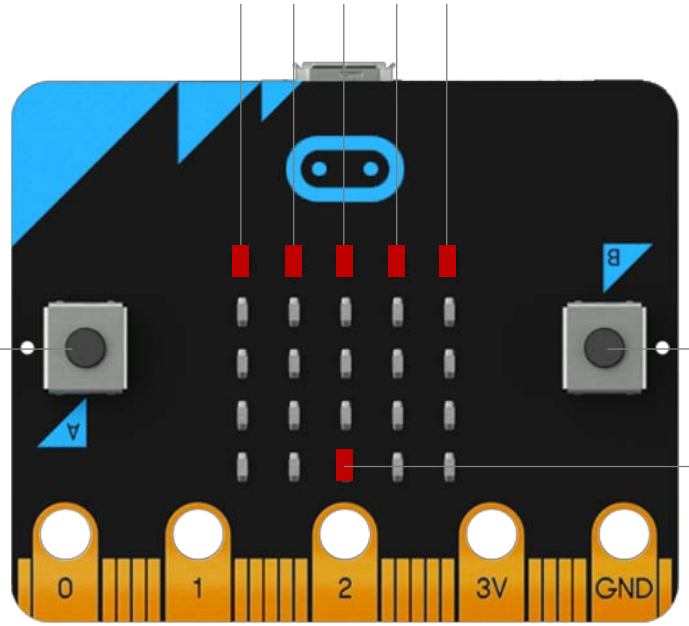
ändere [ *Var* ] auf [ ] | ändere [ *Var* ] um [ ]

analoge Werte von Pin [ ]

zeichne x [ ] y [ ] | Schalte x [ ] y [ ] aus

# TASK Zusatz: Game Astroids

In jeder Spalte fallen nach einer Zufallszeit (0-5sec) Astroiden in der Y-Achse herunter




**Knopf A:**
Spieler nach **Links**

**Knopf B:**
Spieler nach **Rechts**

Spieler, Start Sprite bei [**2**, **4**]

**Erfolgreiches Ausweichmanöver:**
Punkte **+1**

**Kollision mit Astroiden:**
**Game Over**

PH SG

## TASK

- Variablenwerte zu Begin initiieren (z.B. Punkte = 0, spiel_an = wahr)
- Routine für Spielerbewegung entwickeln
- Routine für ersten (Y=0) fallenden Astroiden entwickeln
- Kollisionsereignis entwickeln
- Routine für restliche Astroiden (Y=1 bis Y=4) übertragen

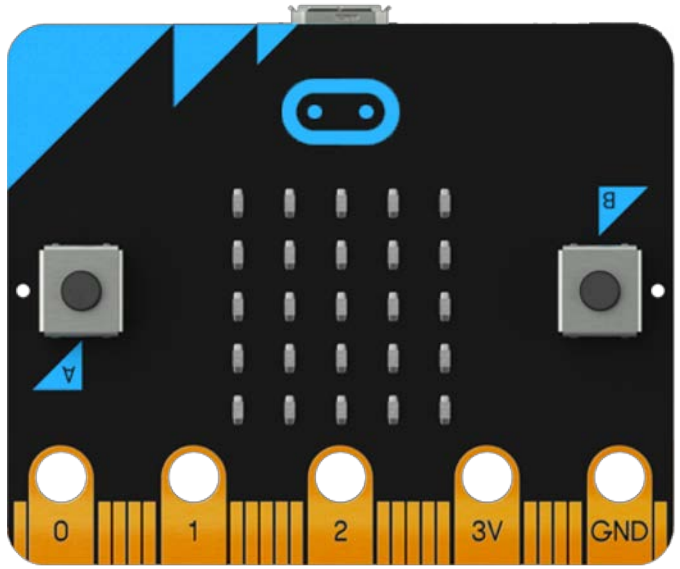
## INHALTE

- Einsatz von Spiel-Blöcken (z.B. Sprite)
- Einsatz von «Wenn – Dann» Bedingungen
- Einsatz von Schleifen

## Benötigte Blöcke (nur Spiel-Block)

- erzeuge Sprite an Position x: [ ] y: [ ]
- [ *Variable* ] y
- [ *Variable* ] stelle [ y ] ein auf [ ]
- [ *Variable* ] ändere [ y ] um 1
- setze Punktestand auf [ ]
- Spiel beendet

# **PROBLEMSTELLUNG:** Mini Spiel




**Variante A:** Mini Spiel **«Zeitgefühl»**

**Variante A:** Mini Spiel **«Reaktionszeit»**

## TASK

- Mit den bisher erarbeiteten Möglichkeiten, ein eigenes Spielkonzept entwickeln und umsetzen

## VORGABEN

- Es ist ein 2-Spieler Spiel
- Das Programm zeigt am Schluss an, ob Spieler A oder B gewonnen hat
- Programmcode ist effizient und frei von Redundanzen

## VARIANTEN

- Variante A – **Zeitgefühl**: Spiel welches das Zeitgefühl (Abschätzen einer Zeitdauer) misst
- Variante B – **Reaktionszeit:** Spiel welches die Reaktionszeit auf ein Ereignis/Impuls misst

Benötigte Blöcke

?