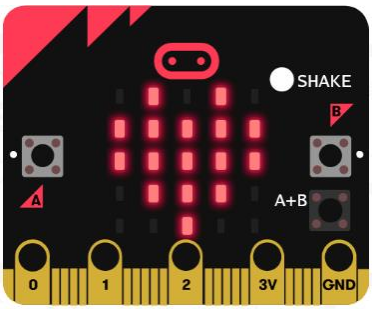
micro:bit mit Python – eine Befehlsübersicht

In Python werden alle Befehle als **Text** geschrieben. Hier findest du eine kleine Übersicht der wichtigsten Befehle. Wenn du einmal nicht weiterkommst, kannst du im Browser einfach auf die blockbasierte Darstellung wechseln. Versuche aber, auch ohne Hilfe auf die Lösung zu kommen!



Beim Start

Jener Code, der *beim Start* ausgeführt werden soll, wird einfach ganz oben angegeben. Auch *Variablen* werden am Beginn des Codes definiert.

Ereignisse und Eingaben sind Methoden

Jeder Ereignis- oder Eingabe-Block, wie der *Dauerhaft, Drücke Button A* oder *Wenn geschüttelt*, wird in Python als **Methode** (oder Unterfunktion) definiert.

```
def dauerhaft():
    # Hier passiert der Inhalt
basic.forever(dauerhaft)
```

entspricht zum Beispiel dem Dauerhaft-Block.

```
def knopfA():
    # Hier passiert der Inhalt!
input.on_button_pressed(Button.A, knopfA)
```

entspricht dem Knopfdruck A

Die letzten Programmzeilen „sagen“ der Methode, wann die Methode aufgerufen werden sollte. Beim zweiten Beispiel heißt das, dass die Methode nur dann ausgeführt werden soll, wenn am *micro:bit* der Knopf A gedrückt wird.

Die Methodennamen (dauerhaft/knopfA) kann man sich selbst aussuchen!

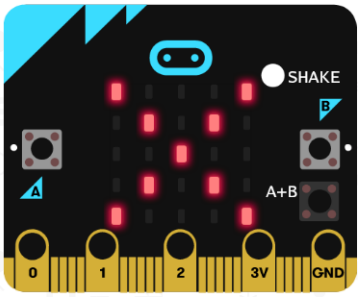
Basic

Alle Befehle die vorne mit *„basic.“* beginnen, betreffen den micro:bit an sich, wie die Ausgabe oder den Bildschirm.

`basic.pause(1000)` pausiert für 1000 mS, also 1 Sekunde.

`basic.clearScreen()` löscht den Bildschirminhalt.

Anzeigen



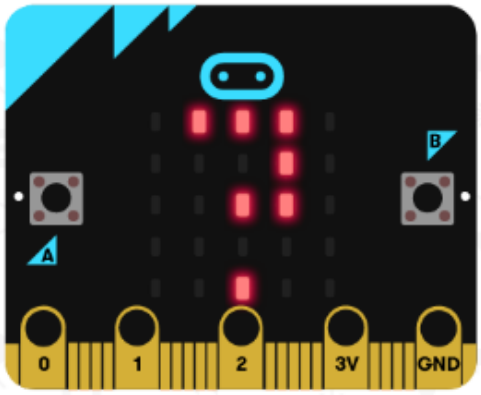
Auf dem Bildschirm möchte man gerne etwas **anzeigen**? Mit `basic.show()` kann man **Symbole** anzeigen.

`basic.show_icon(IconNames.HEART)` zeigt ein Herz an.
`basic.show_icon(IconNames.YES)` zeigt ein Häkchen an.
`basic.show_icon(IconNames.NO)` zeigt ein X an.

Auch Anzeigen von **Text** ist möglich.

```
basic.show_string(“Hallo!”)
```

Auch **selbst gezeichnete Bilder** kann man anzeigen lassen. Um ein selbst kreiertes Bild anzuzeigen, muss man es zuerst "zeichnen". Jede Zahlenfolge repräsentiert eine Reihe des micro:bit-Displays. ‚.‘ heißt hier, das LED an der jeweiligen Position ist aus, bei ‚**#**‘ ist es ein.




```
basic.show_leds("""
        . # # # .
        . . . # .
        . . # # .
        . . . . .
        . . # . .
        """)
```

Verzweigungen und Schleifen

Verzweigungen und Schleifen funktionieren genau gleich, wie in regulären Python-Programmen.

If-Verzweigung

```
if runde == 1:
        basic.show_string("Erste Runde!")
    else:
        basic.show_string("Andere Runde!")
```

Schleife

```
for i in range(4):  #Wichtig: i geht von 0 bis 3!
        basic.show_number(i)
```