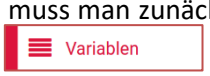
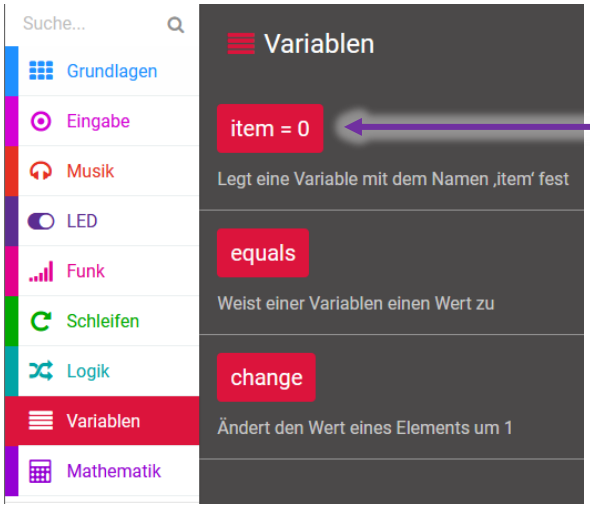
Grundlagen zur Programmierung des micro:bit in Python IV

Werte speichern (aus Variablen)

Ein weiteres wichtiges Programmierkonzept ist die Abspeicherung von Werten und das Arbeiten mit diesen Werten. Man bezeichnet diese als „Variablen“. Eine Variable kann man sich wie ein **leeres Feld oder Kiste** vorstellen, in das man eine Zahl **hineinschreiben** kann. Diese kann nur eine einzige Zahl beinhalten, man kann diese aber ändern (wie ausradieren und neu schreiben), zudem kann man die gespeicherte Zahl immer wieder abrufen (nachsehen, welcher Wert gerade dort gespeichert ist).

Eine solche Variable muss man zunächst erstellen ehe man sie verwenden kann. Dazu klickt man auf die Blockkategorie 

schiebe den Programmierbefehl ‚item=0‘ ins Programmierfeld – ganz zu Beginn. Danach kann man den Namen der Variablen anpassen.



Screenshots zum Erstellen einer Variablen

Man hat weitere Befehle zur Verfügung:

Hier den Namen für die Variable eingeben (Beschriftung).

Block	Beschreibung
equals **item = 0**	Setzt den Wert der Variablen auf den Wert (hier 0), egal was vorher drin war.
change **item += 1**	ändert den bisher gespeicherten Wert um die angegebene Zahl (hier 1), es wird diese Zahl also zum gespeicherten Wert addiert.

Mit diesen neuen Möglichkeiten kann man zum Beispiel einen Zähler programmieren, der zählt wie oft die B-Taste gedrückt worden ist und diese Anzahl mit den LEDs ausgibt.

```
meine_zahl=0

def on_forever():
    basic.show_number(meine_zahl)
basic.forever(on_forever)

def on_button_pressed_b():
        global meine_zahl
        meine_zahl += 1
input.on_button_pressed(Button.B, on_button_pressed_b)
```

WICHTIG: *Wenn man die Variable beim Start angibt, und **diese in einer Methode ändern** möchte, muss man diese Zeile schreiben.* **global meine_zahl** *teilt der Methode mit, dass man die globale Variable (das sind jene, die ganz am Anfang steht) ändern/schreiben möchte. Wenn man die Variable nur anzeigen/lesen möchte, benötigt man diesen Befehl nicht (siehe Methode* **on_forever()**)

Bei obigem Programm wird dauernd der gespeicherte Wert am micro:bit angezeigt. Wird die Taste B gedrückt, so wird der gespeicherte Wert mit der Zahl 1 addiert und ist der neue Variablenwert. Es wird dann also der neue Variablenwert angezeigt (im on_forever()). Man kann mit diesem Befehl (meine_zahl += …) aber auch subtrahieren. Dazu verwendet man eine negative Zahl. Will man z.B. den Wert der Variable um 2 verringern, wenn die Taste A gedrückt wird, macht man das so:

```
meine_zahl=0

def on_forever():
    basic.show_number(meine_zahl)
basic.forever(on_forever)

def on_button_pressed_b():
    global meine_zahl
    meine_zahl += 1
input.on_button_pressed(Button.B, on_button_pressed_b)

def on_button_pressed_a():
    global meine_zahl
    meine_zahl += -2
input.on_button_pressed(Button.A, on_button_pressed_a)
```

Wenn man den Inhalt von Variablen um einen Werkt ändert, kann man das auf zwei Arten machen.

1) **meine_zahl = meine_zahl +1**
2) **meine_zahl +=1**

Probier es aus! Beide Varianten liefern dasselbe Ergebnis!

Man kann auch meine_zahl -=2 schreiben, wird beim Um- und Rückschalten zwischen den Programmiersprachen wieder auf += -2 geändert!

Um einen Startwert anzugeben, setzt man den Variablenwert nur einmal am **Beginn des Programms** auf den gewünschten Startwert.

Wir können aber auch **bei einem zufälligen Wert beginnen**, zum Beispiel zwischen 0 und 9. Überlege kurz, wie das gehen könnte – vielleicht schaffst du das selbst? Die Lösung findet sich sonst auf der nächsten Seite. Eine zweite Erweiterung wäre, eine Fallunterscheidung einzufügen. **Ist der Variablenwert = 10, so soll** anstatt des Variablenwertes **ein Herz angezeigt werden.** Ansonsten (Variablenwert ist nicht 10) soll wie gehabt der gespeicherte Wert ausgegeben werden. Versuche dich selbst an dieser Aufgabe! Die Lösung befindet sich übrigens auch auf der nächsten Seite.

Zur ersten Aufgabe: Wenn man bei einem zufälligen Wert zwischen 0 und 9 beginnen will, nimmt man den Beispielcode für den Startwert von oben (also das ‚Beim Start' mit dem ‚Ändere um') und nimmt anstatt einem fixen Wert (oben war es die Zahl 5) eine Zufallszahl. Der Programmteil ganz oben sieht deshalb wie folgt aus:

```
meine_zahl=randint(0, 9)
```

Man könnte nun das gesamte Programm, mit dem Startwert und dem Zählen, mit einem Zielwert erweitern. Man könnte also, wie in der Aufgabe auf der vorherigen Seite beschrieben, eine Fallunterscheidung einführen, sodass bei einem Wert von 10 ein Herz angezeigt wird (anstatt der Zahl). Um dies zu programmieren, braucht man eine Verzweigung – es ist ja eine Fallunterscheidung. Der Code dafür, sieht dann also gesamt so aus:

```
meine_zahl=randint(0, 9)

def on_forever():
    if meine_zahl==10:
        basic.show_icon(IconNames.HEART)
    else:
        basic.show_number(meine_zahl)
basic.forever(on_forever)

def on_button_pressed_b():
    global meine_zahl
    meine_zahl += 1
input.on_button_pressed(Button.B, on_button_pressed_b)

def on_button_pressed_a():
    global meine_zahl
    meine_zahl += -2
input.on_button_pressed(Button.A, on_button_pressed_a)
```

In diesem Beispiel haben wir alles bisher Gelernte benötigt und angewandt. Zu Beginn wird eine globale Variable auf einen zufälligen Startwert gesetzt. Dieser Variablenwert kann mit den Tasten A bzw. B geändert werden (entweder um +1 oder um -2). Zudem wird währenddessen immer der aktuelle Variablenwert auf dem micro:bit angezeigt. Hat die Variable aber den Wert 10, so wird ein Herz anstatt der Zahl angezeigt.

Du kannst nun neue Arten von Programmen schreiben, schaue dir dazu das zugehörige **Arbeitsblatt** an! Vergiss nicht dein Programm im Anschluss herunterzuladen und auf den micro:bit zu verschieben um es auszuprobieren!