01_01_Auspacken-Einschalten

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

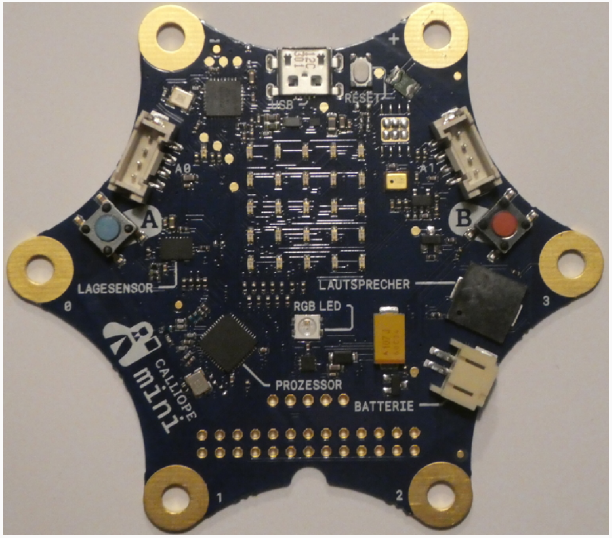
Auspacken des Calliope und Erkunden

Auspacken des Calliope

Das Paket enthaelt

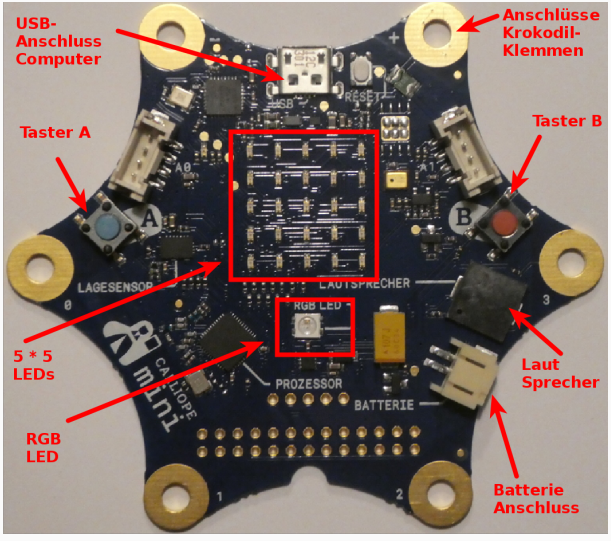
- Den Calliope Mini selbst
- Ein Micro-USB-Kabel
- Ein Batterie-Fach
- 2 Batterien
- ~~2 Krokodil-Klemmen~~
- ~~2 LEDs~~ (war früher mit dabei...)
- Eine Anleitung

Der Calliope Mini selbst



- Der Calliope-Mini, ausgepackt

Calliope Mini : Was ist dran

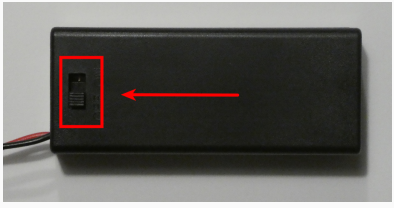


- Der Calliope Mini, ausgepackt

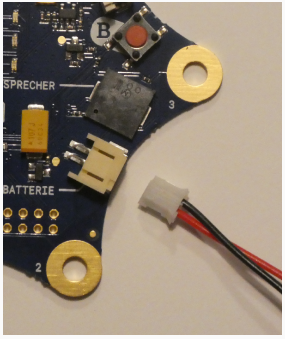
Batteriefach

- Batterie einlegen

- und Batteriefach zumachen

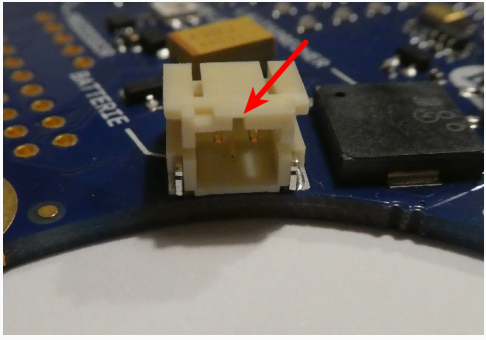


Batterie anschliessen



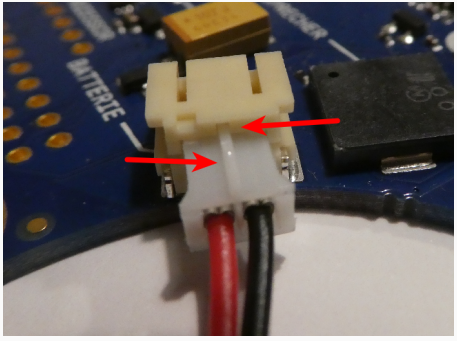
- Das Kabel der Batterie anschliessen

Anschluss - Nut



- Auf die Anschluss-Nut achten.

Kabel-Nase



- Auf die Anschluss-Nut achten, die Nase des Kabels muss nach oben zeigen.

Einschalten und Spielen

- Calliope mit Schalter am Batteriefach einschalten und etwas warten
- Vier verschiedene Spiele stehen zur Auswahl
  - Taste B : Spiel-Nr erhoehen
  - Taste A : Spiel-Nr verringern
  - Calliope schütteln : Spiel auswählen
  - Im Spiel : Beide Tasten drücken => Zurück zum Menu
- Spiele-Auswahl

1. Mini-Orakel
2. Schere Stein Papier
3. Funkt's
4. Snake

- Beschreibung im Beilag-Heftchen

Das Start-Programm

Wenn Ihr im Verlauf des Programmier-Kurses das ursprüngliche Programm mit Euerem eigenen überschrieben habt (was ich hoffe), dann gibt es hier das ursprüngliche eingebaute Programm zum

Download: Original-Start-Programm hier klicken

Wir lernen später, was wir damit anfangen, können.

Merkt Euch einfach an dieser Stelle, dass es hier das Programm zum Herunterladen gibt.

Lizenz/Copyright-Info

01_02_Start_Simulator

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Start Simulator und Bearbeiten

Starten des Editors im Web-Browser

Aufruf von https://makecode.calliope.cc

Der Arbeits-Bereich

Links Simulator-Bereich

Simuliert den Calliope

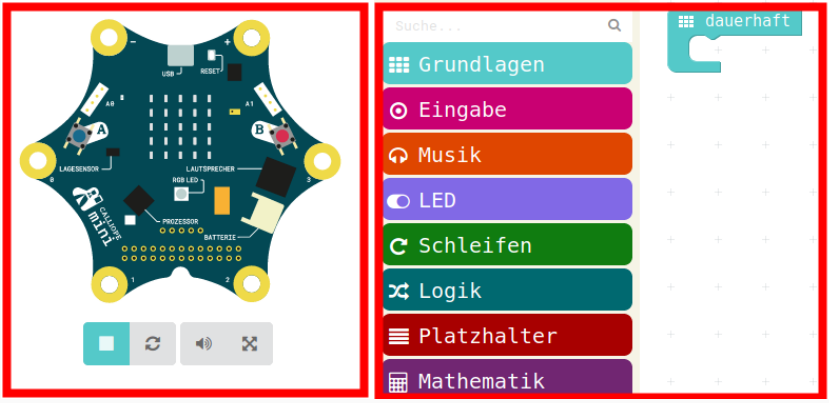


Figure 1: Arbeits-Bereich

Rechts der Arbeits-Bereich zum Bearbeiten/Programmieren

Bearbeiten von Programmen

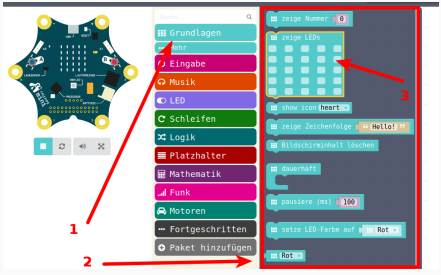


Figure 2: Grundlagen öffnet ein Menu

- Ein Klick auf Grundlagen
- Öffnet ein Menu mit grundlegenden Befehlen
- Hier klickt man z.B. mit der Maus auf “zeige LEDs” und schiebt das ICON in den Arbeitsbereich,
- Dazu mit der linken Maustaste auf das ICON gehen, die Maustaste NICHT loslassen und dann das ICON nach rechts in den Arbeitsbereich schieben

Arbeiten im Arbeitsbereich

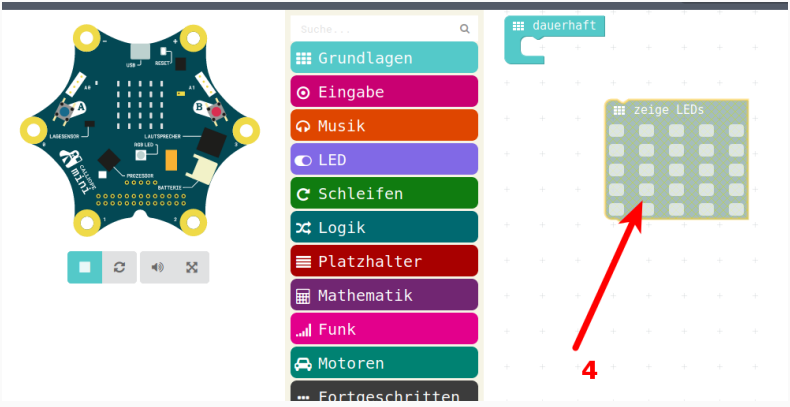


Figure 3: Zeige leds im Arbeitsbereich

- Im Arbeitsbereich landet das Icon “zeige LEDs”

Einhängen der Symbole in die Arbeits-Schleife

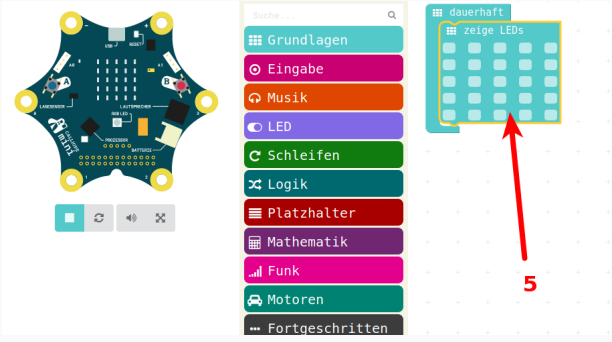


Figure 4: Einklicken in die Schleife

- Dieses kann man nun in die vorhandene Schleife “dauerhaft” einklicken
- Auch dazu das ICON mit der linken Maustaste anklicken und die Maustaste gedrückt halten
- Mit gedrückter Maustaste in die Schleife schieben

Bearbeiten des Programm-Stücks

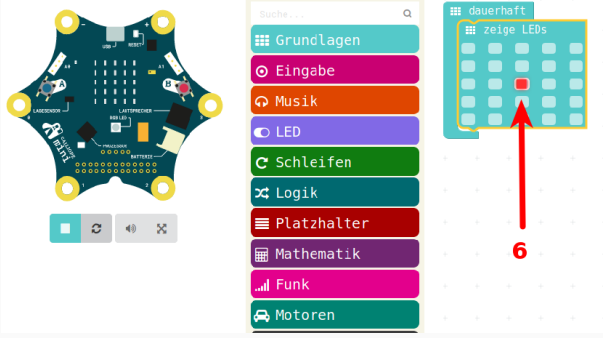


Figure 5: Leds ändern

- Nun kann man einzelne Leds im Arbeitsbereich an und ausschalten
- Einfach mit der linken Maustaste die LED anklicken zum Ein und
- nocheinmal Anklicken um die LED wieder auszuschalten

Übersetzen in Computer-Sprache

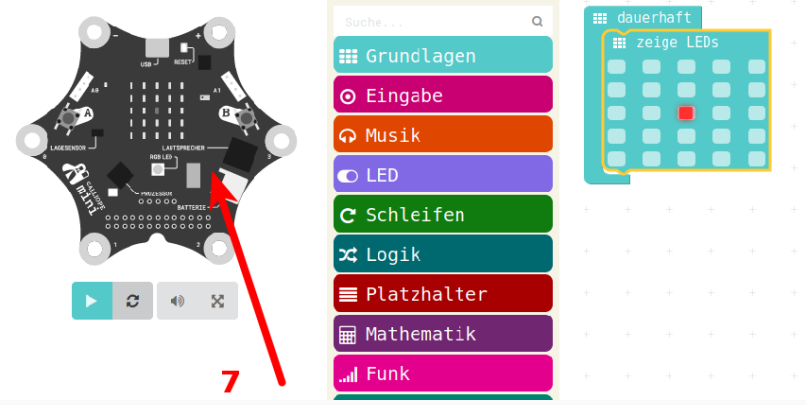


Figure 6: Simulator arbeitet

- Im Hintergrund wird das Programm neu "übersetzt" und in den Simulator geladen
- Der Simulator ist grau, kann nicht genutzt werden

Das Programm läuft im Simulator

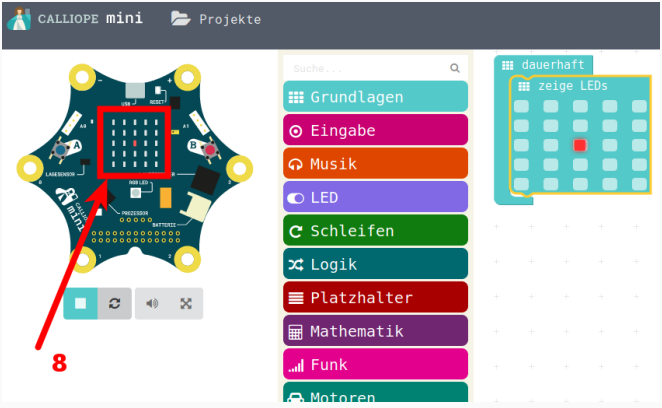


Figure 7: Programm im Simulator

- Das Programm ist in Calliope-Computer-Sprache übersetzt und in den Simulator geladen

Lizenz/Copyright-Info

01_03_LED_Anzeigen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

LED anzeigen

Auswahl aus Menu

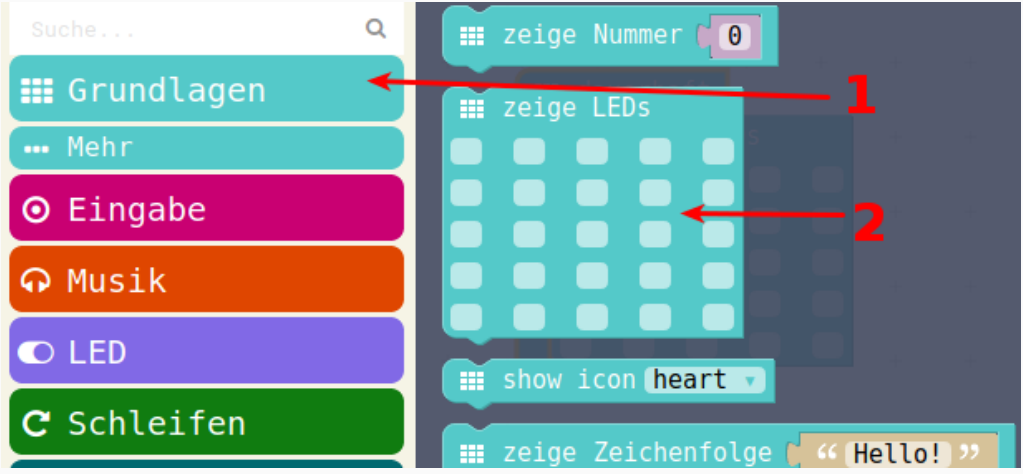


Figure 1: LED anzeigen Menu

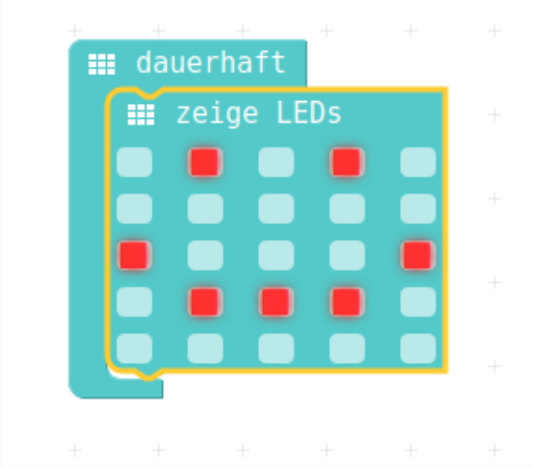


Figure 2: LED anzeigen

JavaScript-Code / Hex-Code

JavaScript-Code
Java-Script-Code

```
basic.forever(() => {
    basic.showLeds(`
        . # . # .
        . . . . .
        # . . . #
        . # # # .
        . . . . .
        `)
})
```

Download Hex-Code
Hex-code

Erweiterung mit Warten

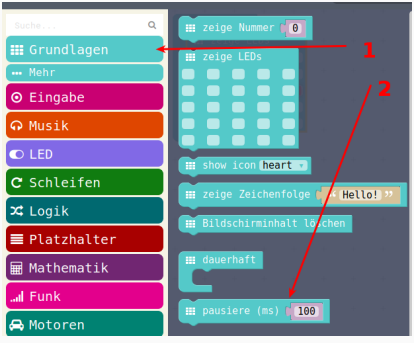


Figure 3: Pause einfuegen Menu

Einheit:

- ms ist MilliSekunden, das ist eine tausendstel Sekunde
- 1000 Millisekunden sind 1 Sekunde
- Wenn man mehrere verschiedene LED-Anzeigen haben will, ca 1 Sekunde = 1000 ms dazwischen

PXT-Code

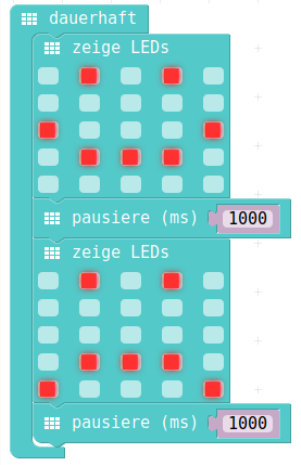


Figure 4: LED anzeigen mit Pause

JavaScript-Code / Hex-Code

Java-Script-Code

```
basic.forever(() => {
    basic.showLeds(`
        . # . # .
        . . . . .
        # . . . #
        . # # # .
        . . . . .
        `)
    basic.pause(1000)
    basic.showLeds(`
        . # . # .
        . . . . .
        . . . . .
        . # # # .
        # . . . #
        `)
    basic.pause(1000)
})
```

Lizenz/Copyright-Info

Für alle Texte und Bilder auf diesen Folien:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

01_04_Programm_Auf_Calliope_Laden

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Programm auf den Calliope laden

Simulator

Bislang haben alles “nur” im Browser gesehen, Code und Simulator. Nun wollen wir aber unseren ersten Code auf dem Calliope laufen lassen.

Namen vergeben

Dazu müssen wir als erstes unserem ersten Programm einen sinnvollen Namen geben:

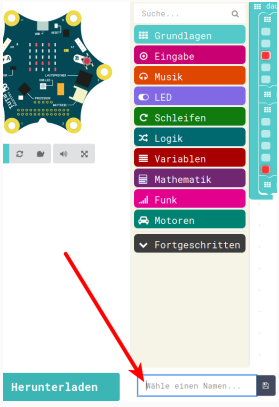


Figure 1: Namen wählen

Speichern (1)

Nach der Eingabe eines sinnvollen Namens und klick auf das Disketten-Symbol (wer kennt noch Disketten?)

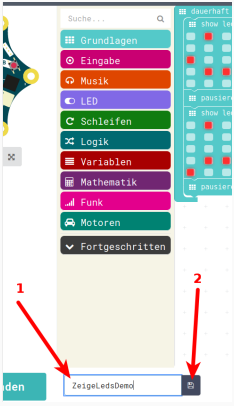


Figure 2: Eingeben und Speichern

Speichern (2)

öffnet sich je nach Betriebs-System ein Speichern-Dialog, der es ermöglicht, das Programm als HEX-Datei auf der Festplatte abzulegen. Die meisten Browser sind so konfiguriert, dass die Dateien in einem Ordner namens “Downloads” abgelegt werden

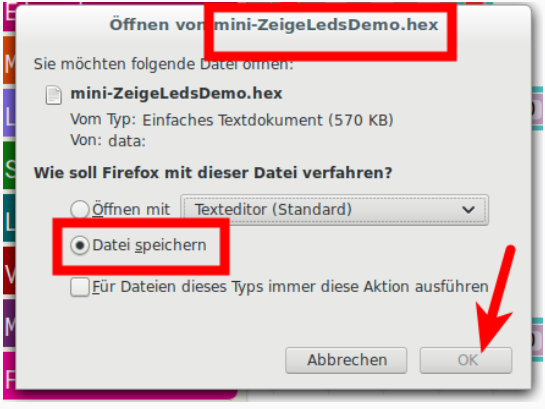


Figure 3: Speicher Dialog

Speicher-Ort finden (1)

Da wir das aber nicht sicher wissen, kann man auch den Browser anweisen, einen zum Ort der heruntergeladenen Dateiein zu führen.

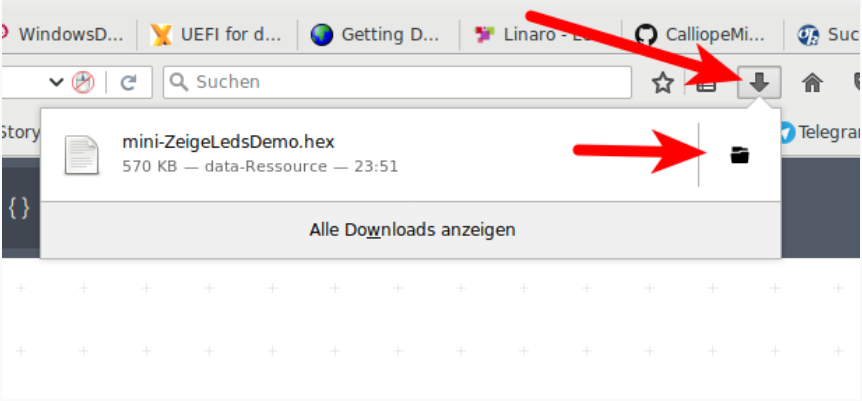


Figure 4: Download-Button

Speicher-Ort finden (2)

In diesem Beispiel wurde die HEX-Datei im Verzeichnis “Downloads” des Benutzers “Jogi” abgelegt.

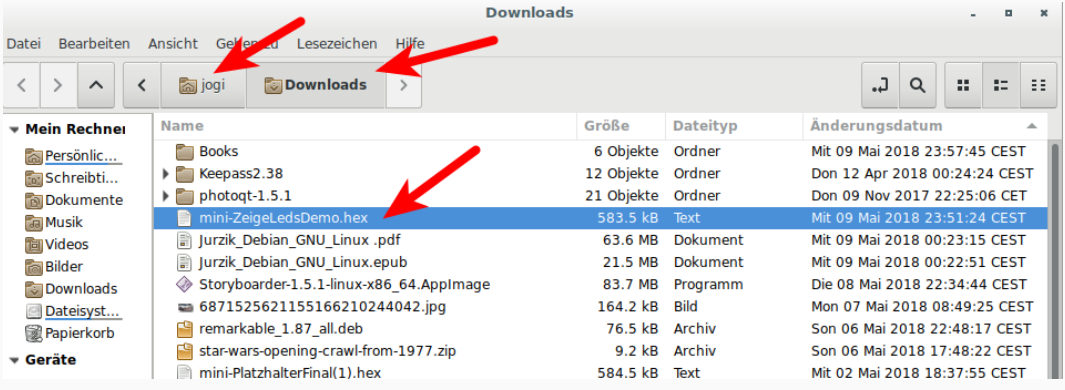


Figure 5: Download-Ordner finden

Calliope Mini anstecken (1)

ACHTUNG

Vor Anstecken des Calliopes an einen USB-Port des Computers die Batterie-Spannung abschalten. Besser noch die Batterien abstecken. Der Calliope darf nicht gleichzeitig von den Batterien gespeist werden und am USB-Port des Computers per Kabel hängen!

Calliope Mini anstecken (2)

Nach Anschluss des USB-Kabels an den Mini

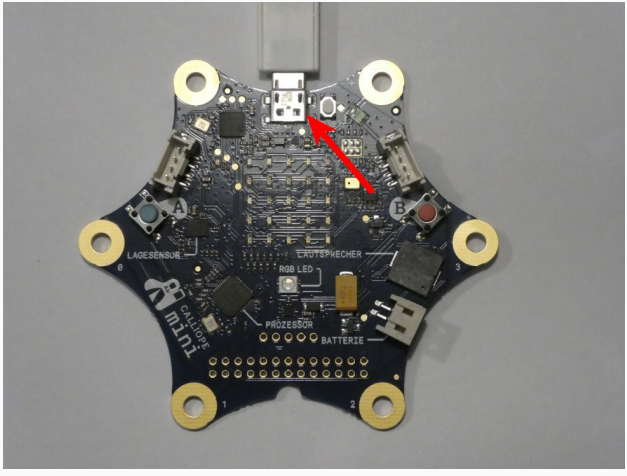


Figure 6: Mini Anstecken

Calliope Mini anstecken (3)

und an einen USB-Port des Computers erscheint der Calliope Mini als USB-Speicher am Computer.

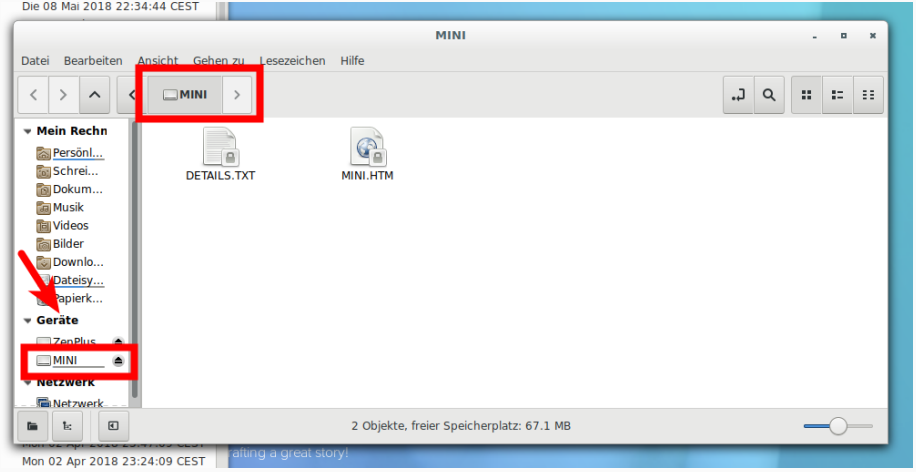


Figure 7: Mini erscheint als USB-Laufwerk

HEX-Datei auf den Mini laden (1)

Nun kann man die Datei entweder mit der Maus per Drag and Drop auf den Mini ziehen:

- Datei mit der linken Maustaste anklicken,
- Maustaste festhalten
- Datei auf das Ziel, also in unserem Fall den MINI
- ziehen und jetzt die Maustaste loslassen

HEX-Datei auf den Mini laden (2)

oder aber mit Kopieren/Einfügen (Maus)

- Datei einmal mit der linken Maustaste anklicken, die Datei ist angewählt
- rechte Maustaste auf die Datei klicken
- Kopieren anwählen
- in einen freien Bereich im Ziel-Laufwerk mit der linken Maustaste klicken
- rechte Maustaste klicken, es erscheint ein Menu
- Einfügen wählen


HEX-Datei auf den Mini laden (3)

oder aber mit Kopieren/Einfügen (Tastatur)

- Datei einmal mit der linken Maustaste anklicken, die Datei ist angewählt
- STRG -Taste festhalten, C drücken (C wie Copy)
- in einen freien Bereich im Ziel-Laufwerk mit der rechten Maustaste klicken
- STRG-Taste festhalten, V drücken (V weil es direkt neben C ist)
- Das fügt die kopierte HEX-Datei auf das Laufwerk

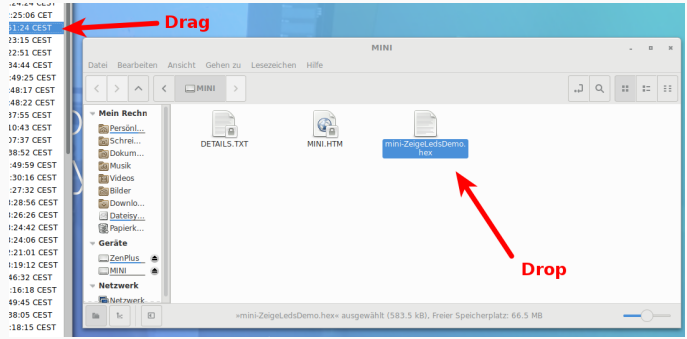


Figure 8: Datei ziehen

Das Programm wird “geflasht”

Vom USB Speicher-Bereich des Calliopes wird es automatisch in den internen Programm-Bereich geflasht. Das erkennt man an der blinkenden gelben LED

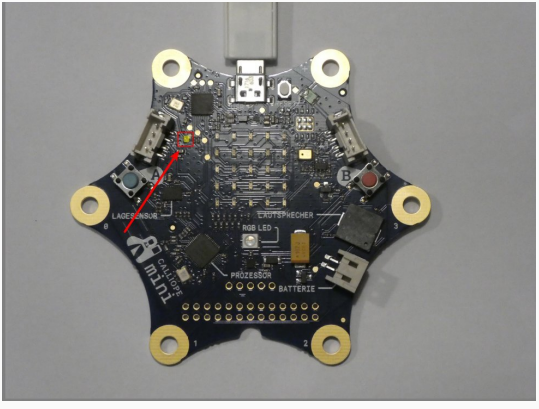
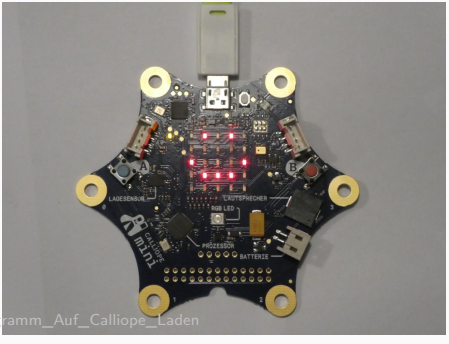


Figure 9: Calliope wird programmiert

Das Programm läuft

Wenn das Programm fertig geflasht ist,

- dann resetet sich der Calliope automatisch,
- er verschwindet kurz als USB-Laufwerk vom Computer
- er erscheint wieder als USB-Laufwerk am Computer, aber das Programm ist verschwunden
- dafür wurde es in den internen Programm-Speicher übertragen und
- wird nun ausgeführt



Lizenz/Copyright-Info


01_05_Texte_Anzeigen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Texte anzeigen

Zeichenketten

In der Computer-Sprache nennt man Texte auch “Zeichenketten”, eine ‘Verkettung’ von einzelnen Zeichen.
Man trifft auch sehr oft auf die englische Bezeichnung “String”.

Darum, wenn man einen Text anzeigen will : => Zeichenketten!


Auswahl aus Menu

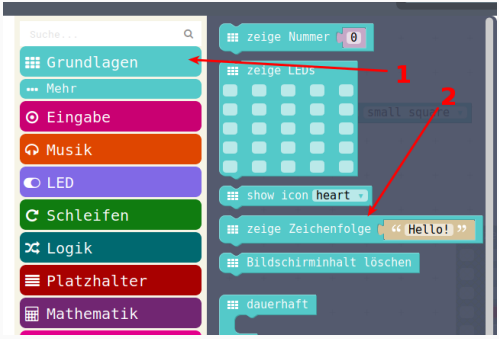


Figure 1: Menu-Auswahl

PXT-Code

In der Computer-Welt, wenn man eine neue Programmier-Sprache lernt, ist üblicherweise das erste Programm ein “Hallo Welt.” Diese Programm gibt genau diese Zeichenfolge auf dem Bildschirm aus.
Unser “Bildschirm” ist die LED-Anzeige. Wenn der Text nicht auf den Bildschirm (die LED-Anzeige) passt, dann wird ein Lauftext erstellt.

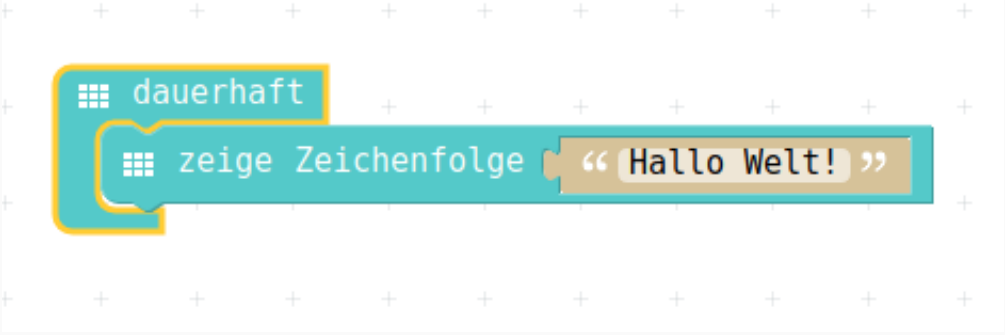


Figure 2: Hello World in Calliope

JavaScript-Code / Download HEX

Java-Script-Code

```
basic.forever(() => {
    basic.showString("Hallo Welt!")
})
```

Download Hex-Code

Hex-code

Lizenz/Copyright-Info

Für alle Texte und Bilder auf diesen Folien gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

01_06_Zahlen_Anzeigen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Zahlen/Nummern anzeigen

Zahlen / Nummern

In der Computer-Sprache unterscheidet man die Zeichenketten/Strings (im vorherigen Kapitel) von den Nummern/Zahlen.
Zeichenketten kann man anzeigen (und auch noch anders verändern) aber Nummern/Zahlen eignen sich zum Rechnen.
Darum wird unterschieden zwischen Texten und Nummern.

- Wenn man reine Texte hat: => Zeichenkette
- Wenn man Nummern (mit denen man rechnet) anzeigen will: => Nummern


Auswahl aus Menu

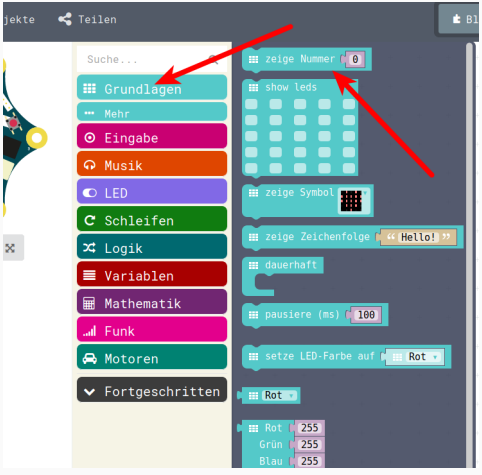


Figure 1: Menu-Auswahl

PXT-Code

Im ersten Schritt wollen wir nur eine einzelne Nummer anzeigen. In weiteren Schritten zeigen wir dann Zahlen an, die nicht am Stück auf das Display passen und schauen uns an, wie das angezeigt wird.

- Kleiner Tipp : Mit einem gelöschten Bildschirm (zeige LEDs) und evt noch einen Pause (pausiere ms) lässt sich das dann besser identifizieren, was angezeigt wird.

Einzelne Ziffer

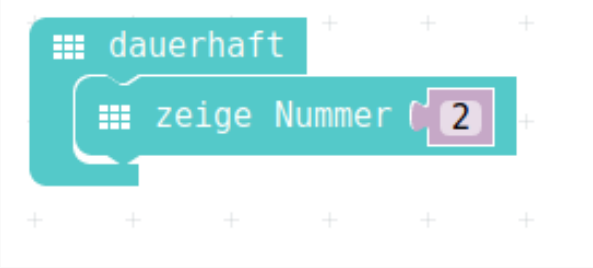


Figure 2: Einzelne Ziffer

Grosse Zahlen

Figure 3: Grosse Zahlen

Grosse Zahlen mit Löschen

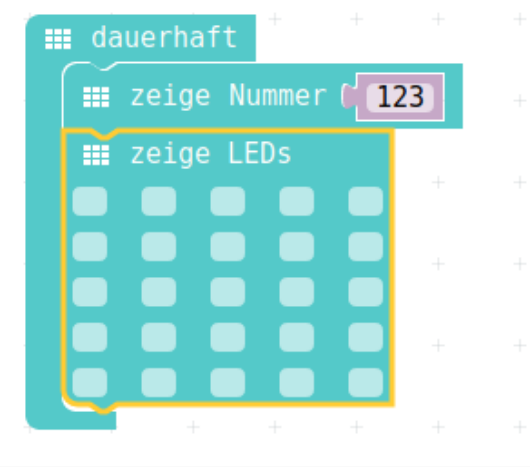


Figure 4: Grosse Zahlen mit Löschen

Grosse Zahlen mit Löschen

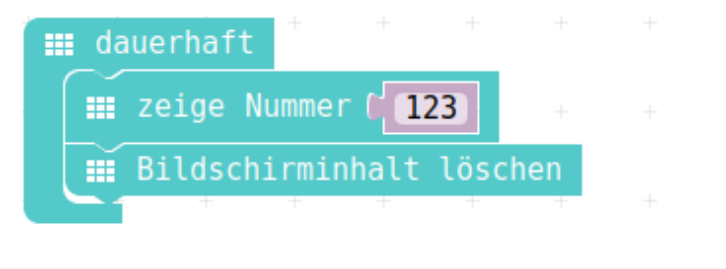


Figure 5: Grosse Zahlen mit anders Löschen

- Dieses "**Bildschirminhalt löschen**" bewirkt das Gleiche wie vorher die LED-Matrix mit nicht angeklickten LEDs
- Es wird aber schneller durchgeführt
- Und es braucht weniger Platz auf unserem Arbeitsbereich
- Man findet es unter **"Grundlagen->Mehr"** ganz unten

Grosse Zahlen mit Löschen und Warten

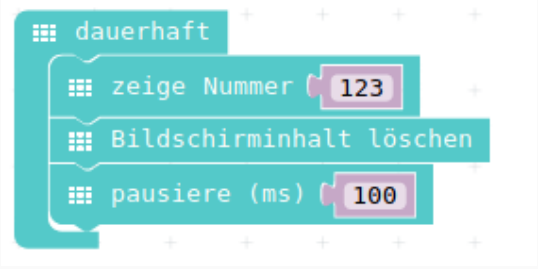


Figure 6: Grosse Zahlen mit Löschen und Warten

- Wenn man noch eine Pause einführt, dann sieht man besser was passiert
- Pausen heissen : **pausiere (ms)** und finden sich auch im Menu: **Grundlagen**

Additions-Ergebnis einfügen

Vorher hatte ich gesagt, dass sich der Befehl **zeige Nummer** eignet, um Zahlen anzuzeigen, mit denen man auch richtig rechnen kann.

Das wollen wir ausprobieren, wir wollen nun direkt das Ergebnis einer Plus-Rechnung, einer Additon sehen

- Im Menu **Mathematik** findet man eine Additions-Puzzleteil, dieses ziehen wir nun in unser **zeige Nummer** rein.

Additions-Ergebnis anzeigen

- Wir ziehen die Additon über die Zahl **123** im Befehl **zeige Nummer** drüber, die Zahl wird dann einfach entfernt.
- Für die Addition ersetzen wir noch von Hand die beiden 0 durch Zahlen (hier 2 und 5)
- Unser Calliope (oder unser Simulator) berechnet das Ergebnis und zeigt es an..

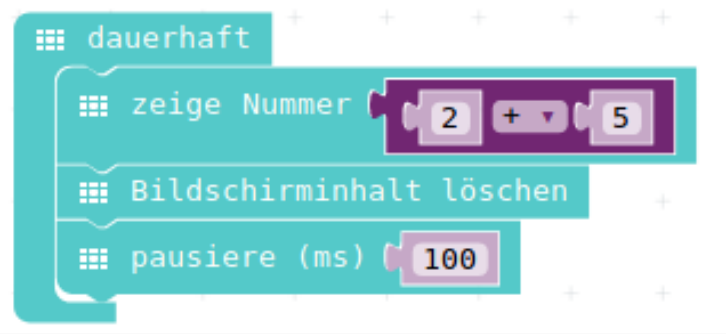


Figure 7: Zahlen Additionsergebnis zeigen

Additions-Ergebnis schöner anzeigen

- Nun können wir nochmal eine Zeichenkette vorher eintragen, um unseren kleinen “Taschenrechner” zu verschönern.
- Wir holen uns wieder aus dem Menu Grundlagen den Befehl **zeige Zeichenfolge**
- Klicken ihn über der Berechnung in unsere **dauerhaft** - Schleife ganz oben ein.
- Und schreiben unsere Berechnung als Text hin

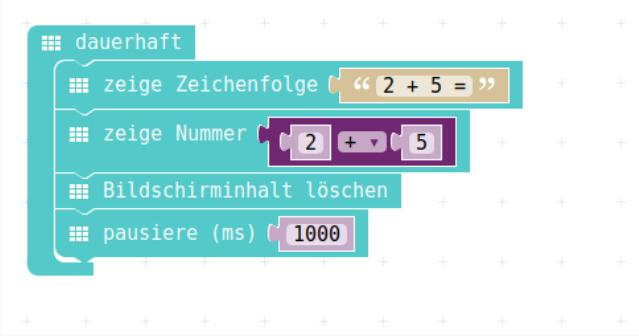


Figure 8: Zeichen und Ergebnis zeigen

JavaScript-Code

Java-Script-Code

```
basic.forever(() => {
    basic.showString("2 + 5=")
    basic.showNumber(2 + 5)
    basic.clearScreen()
    basic.pause(1000)
})
```


Lizenz/Copyright-Info


01_07_Platzhalter

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Platzhalter / Variablen

Variablen

Bis jetzt haben wir nur Dinge angezeigt, die wir genau so eingegeben haben. Nun wollen wir aber “Bewegung” und “Veränderung” in die Dinge bringen, die wir anzeigen wollen. Dazu müssen wir ein wichtiges Element beim Programmieren kennen lernen:

Platzhalter :

Anstatt den Programm-Code immer abzuändern um unterschiedliche Zahlen anzuzeigen brauchen wir sogenannte Platzhalter.
Platzhalter nennt man beim Programmieren auch **Variablen**, weil die Platzhalter **unterschiedliche Werte**, variable Werte aufnehmen kann.
Das kann man sich z.B. beim einfachen Zählen mit zwei Händen vorstellen:

Hände als Variablen

- Jede Hand ist ein Platzhalter.
- Jede Hand kann in diesem Fall Werte bis 5 “aufnehmen” (Wertebereich des Platzhalters von 0-5)
- Die beiden Hände können unterschiedliche Werte “aufnehmen”
- Nachdem beiden Händen Werte “zugewiesen” wurden, kann man mit den Platzhaltern rechnen.
- Linke_Hand + Rechte_Hand = Gesuchte_Summe
- In der Programmierung wird das ganze dann umgedreht :
- Gesuchte_Summe = Linke_Hand + Rechte_Hand


Die Hände

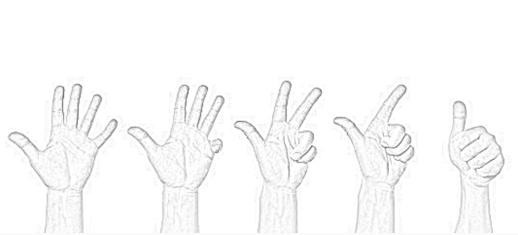


Figure 1: Linke Hand

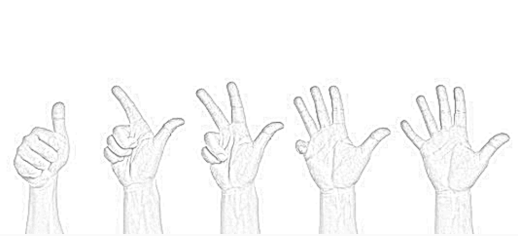


Figure 2: Rechte Hand

Zuweisung

Damit Platzhalter Werte aufnehme können, werden ihnen Werte zugewiesen.
Das geschieht in der Programmierung mit dem Gleichheitszeichen.
Dieses Gleichheitszeichen ist nicht zu verwechseln mit dem Gleichheitszeichen in der Mathematik.
Das Gleichheits-Zeichen beim Programmieren bedeutet, dass dem Platzhalter auf der linken Seite der Wert auf der rechten Seite des Gleichheits-Zeichens zu gewiesen wird.

Beispiel 1:

Linke_Hand = 3

Figure 3: Linke Hand

heisst : Ab jetzt hat der Platzhalter **Linke_Hand** den Wert 3.

Beispiel 2:

Rechte_Hand = 5

Figure 4: Rechte Hand

heisst : Ab jetzt hat der Platzhalter **Rechte_Hand** den Wert 5.

Addition mit Platzhaltern (1)

Anstatt $3 + 5$ heisst unsere Rechnung nun:

Linke_Hand + Rechte_Hand = Gesuchte_Summe

Um daraus einen Programmiervorschrift zu machen, dreht man die beiden Teile um das Gleichheits-Zeichen herum, also:


Addition mit Platzhaltern (2)

Gesuchte_Summe = Linke_Hand + Rechte_Hand

Diese eine **Berechnungs-Anweisung / Formel** ist unser **“Programm”** und ist für völlig verschiedene Werte von Rechte_Hand und Linke_Hand durchführbar.
Man kann die Zuweisungen irgendwann beim Programm-Start machen und erst viel später (wenn man als Mensch schon lange die Werte vergessen hat) die beide Platzhalter addieren.


Eine Variable anlegen

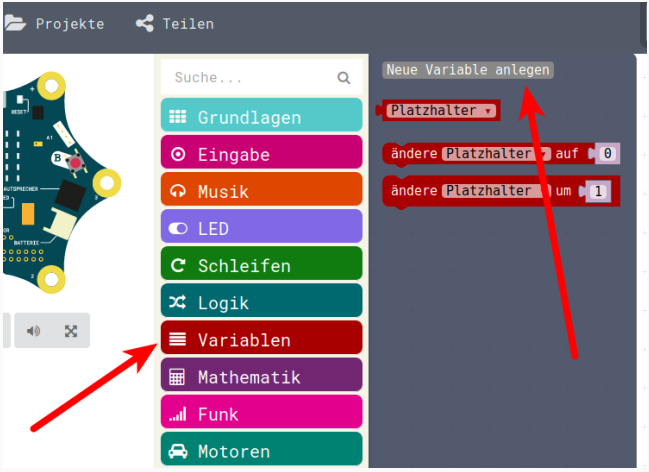


Figure 5: Menu-Anlegen

Variable benennen

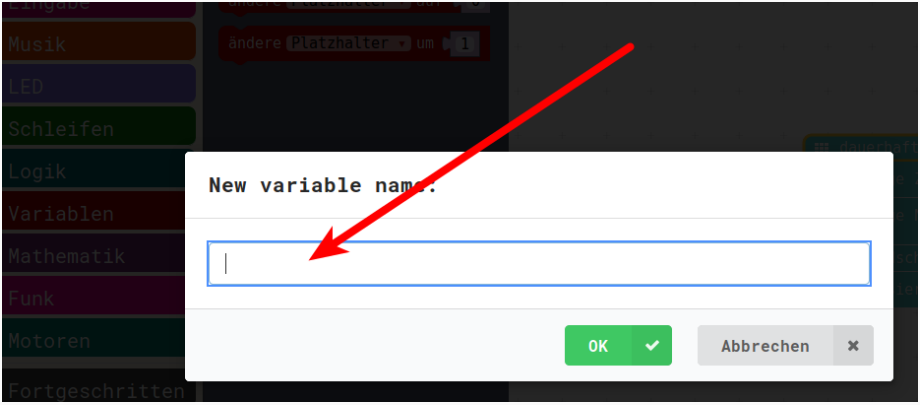


Figure 6: Menu-Benennen

Wichtig : Variablen-Namen

Beim Benennenn der Variablen sollte man ein paar Dinge beachten:

- Variablen dürfen **keine Leerzeichen** enthalten
- Variablen dürfen Zahlen enthalten, aber **nicht mit Zahlen anfangen**.
- Variablen sollten **keine Sonderzeichen** enthalten, dazu zählen auch ä,ö und ü (ersetzen durch ae,oe und ue)

Darum wird aus unserer Rechnung oben:

- Gesuchte Summe = Linke Hand + Rechte Hand
- GesuchteSumme = LinkeHand + RechteHand oder
- Gesuchte_Summe = Linke_Hand + Rechte_Hand oder
- gesuchteSumme = linkeHand + rechteHand usw ...

Variablen-Namen eingeben

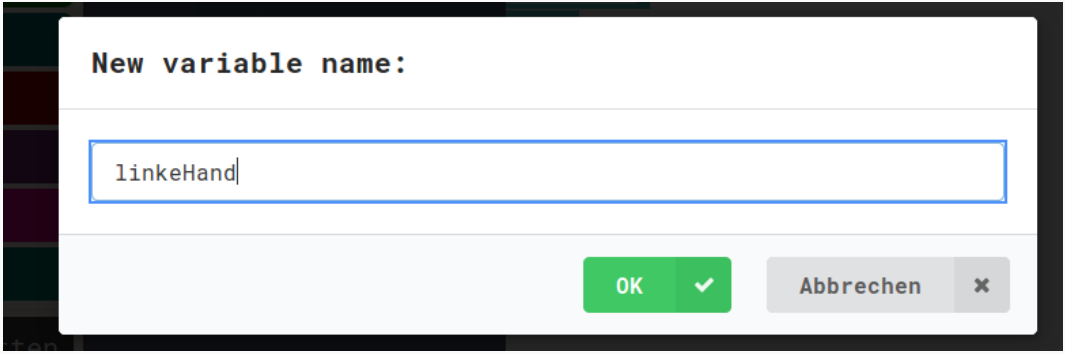


Figure 7: Menu-Benennen

Zuweisung und Benutzung (1)

Wir haben oben in Texten die Namen der Variablen hingeschrieben:

- **linkeHand** = 3
- rechteHand = 5
- gesuchteSumme = **linkeHand** + rechteHand

Wir haben hier zweimal die Variable **linkeHand** stehen, einmal auf der linken Seite des = - Zeichens einmal auf der rechten Seite.

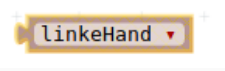
Im einen Fall belegen wird die Variable **linkeHand** mit einem Wert (3), das andere Mal benutzen wir den Wert, wir "fragen" die Variable, welchen Wert sie denn enthält.


Zuweisung und Benutzung (2)

Unsere Calliope-Programmiersprache unterscheidet/muss unterscheiden, ob man einer Variablen einen bestimmten Wert zuweist, oder ob man die Variable/den Platzhalter benutzen will.

Wenn man der Variable einen Wert **zuweisen** will, dann muss man diesen ganzen Block verwenden:

Wenn man die Variable abfragen will, man will sie **benutzen** , dann kann man folgenden Block verwenden, das Puzzleteilchen:



(Siehe dazu auch den Refresh in Tag3 zum Thema Variablen)

Zuweisung und Benutzung (3)

Hat man nun mit obigem Befehl eine neue Variable angelegt, dann ist sie im **Menu Variablen** als Puzzle-Teilchen vorhanden, sprich man kann sie **benutzen/abfragen**.

Aber eine Zuweisung wie bei der Variable namens **Platzhalter** ist nicht vohanden:

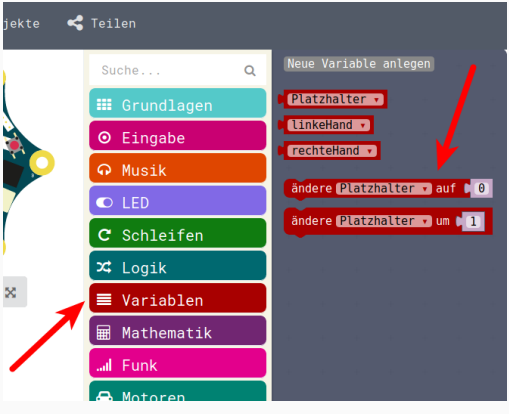


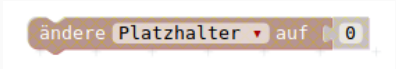
Figure 8: Menu-Zuweisung

Neue Variable belegen

Lösung des Problems:

Das kleine Dreieck bei Platzhalter bietet ein Auswahlmenu aller angelegten Variablen.

Dazu ziehen wir den Baustein auf die Arbeitsfläche:



und klicken anschliessend auf das kleine Dreieckchen. Dann öffnet sich ein Auswahlmenu, in welchem wir die Variable auswählen können:

Lizenz/Copyright-Info

Für alle Texte und Bilder auf diesen Folien gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

01_08_BeimStart

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

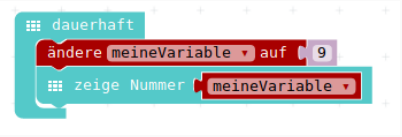

Variablen nutzen / BeimStarten

Variablen nutzen

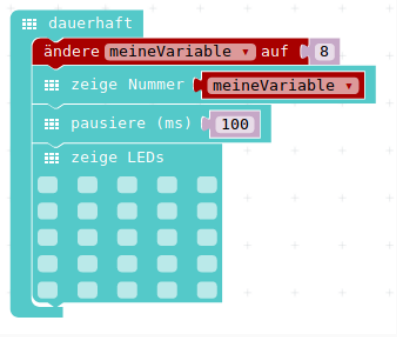
Nun wollen wir unser neues Wissen über Variablen in einem einfachen Programm ausprobieren.

- In der **dauerhaft**-Schleife:
- Wir belegen eine Variable **meineVariable** mit einem beliebigen Zahlenwert
- Wir lassen uns diese Variable **meineVariable** als Zahl anzeigen

Erstes Programm mit Variablen



oder damit sich wenigstens ein bisschen was bewegt: (**pausiere** und **zeige LEDs** sind beide im Grundlagen-Menu)



Beim Start (1)

Bislang haben wir alle unsere Programm-Teile in die Schleife mit dem Namen **dauerhaft** reingezogen. Diese Schleife läuft - wie Ihr Name vermuten lässt - die ganze Zeit “im Kreis”.

Es werden also alle Befehle immer wieder ausgeführt.

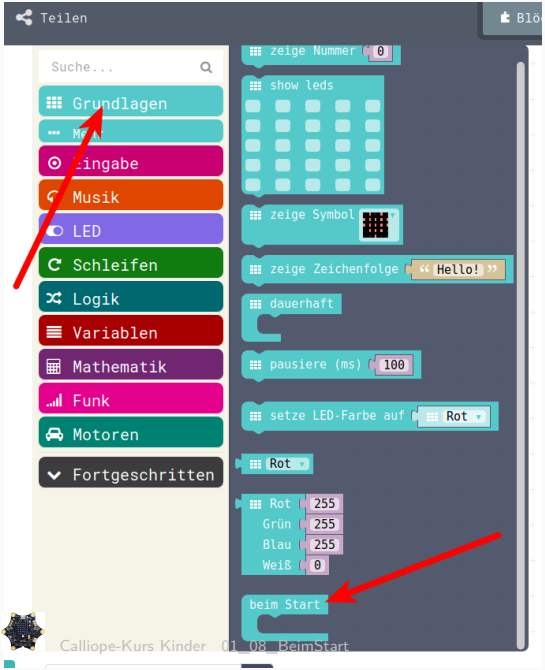
Eine paar andere Möglichkeit, Programmteile zu starten, werden wir gleich sehen.

Eine ist eine ähnlich aussehender Block, die sich **beim Start** nennt und auch im **Grundlagen**-Menu zu finden ist.

Dieser Block wird - wie zu erwarten - nur ein einziges Mal, beim Start des Calliope bzw beim Start des Simulators durchgeführt.

Das wollen wir nun mal ausprobieren, wir holen uns den **beim Start** Block zusätzlich auf die Arbeitsfläche.

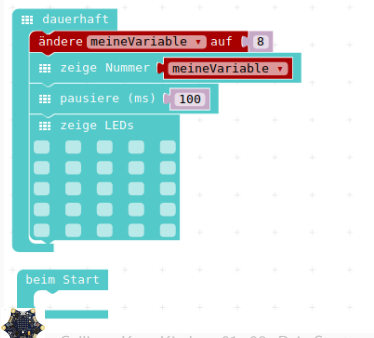
Beim Start (2)


Teilen
Blö
Suche...
Grundlagen
Eingabe
Musik
LED
Schleifen
Logik
Variablen
Mathematik
Funk
Motoren
Fortgeschritten
zeige Nummer 0
show leds
zeige Symbol
zeige Zeichenfolge “ Hello! ”
dauerhaft
pausiere (ms) 100
setze LED-Farbe auf Rot
Rot
Rot 255
Grün 255
Blau 255
Weiß 0
beim Start

Beim Start (3)

Unterschied : Schleife vs Einmal Ausführen

Unser neues Programm:



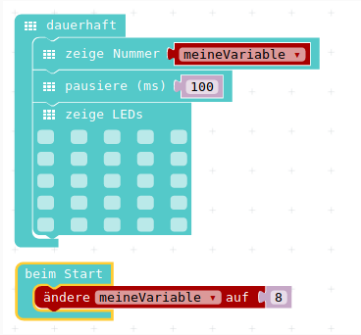
Beim Start mit Funktion

Das einmalige Belegen der Variable **meine Variable** können wir nun aus der **dauerhaft**-Schleife rausziehen und in den **beim Start** - Block einfügen.

Hinweis : Man kann immer nur ganze Programm-Teile nach **unten** wegziehen, d.h.

- den gesamten Programm-Block aus der **dauerhaft**-Schleife rausziehen
- alles unterhalb **ändere meineVariable** wieder in die Schleife reinziehen
- **ändere meineVariable** nun in den **beim Start**-Block einklicken

Beim Start mit Funktion



Vorbereitung

- Wenn wir unser Programm nun im Simulator oder im Calliope anschauen, hat sich eigentlich nichts geändert.
- Aber wir haben eine neue Möglichkeit kennengelernt, um Befehle ausführen zu lassen, wenigstens einmalig beim Start
- Und wir haben unser Programm vorbereitet, fü die nächsten Schritte...

Lizenz/Copyright-Info

01_09_TastenEingabe

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Eingabe mit Tasten

Starten Programm-Teile via Tastendruck

Nun gibt es auch die Möglichkeit, Programm-Teile dann ausführen zu lassen, wenn ein Taste gedrückt wird. Dazu holen wir uns die entsprechende “Klammer” um unser Programm herum aus dem Bereich “Eingabe”

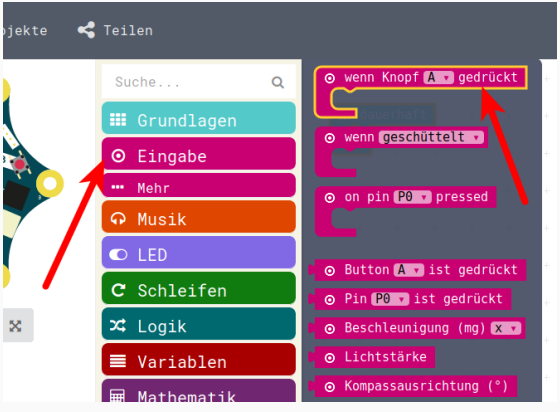


Figure 1: InputButton

Bearbeiten

Den Variablen-Anzeige-Block ziehen wir aus der **dauerhaft**-Schleife auf den Arbeitsbereich.

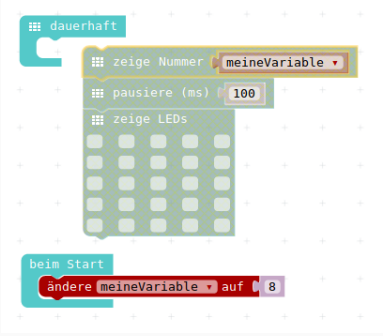


Figure 2: Ausgegraut

Wenn ein Block nicht ausgeführt werden kann (dieser "hängt in der Luft, ihm fehlt eine Ausführungs-Möglichkeit), dann wird er ausgegraut. Man sieht dann schon an der wachen Farbe (ausgegraut), dass nichts mehr passiert!

Einfache Tastendruck-Demo

Dazu holen wir uns also den **wenn Knopf A gedrückt** in den Arbeitsbereich

Nun noch eine Reaktion auf den Tastendruck, z.B. ein LED-Gesicht:

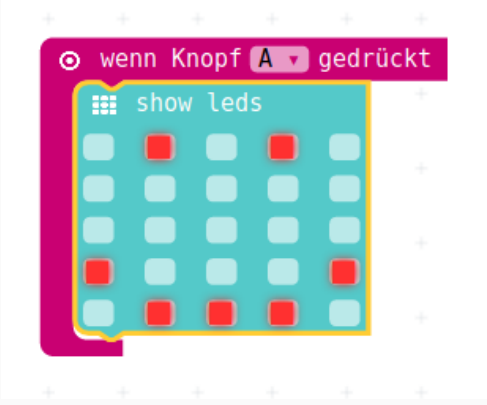


Figure 3: Reaktion auf Tastendruck

Damit haben wir ein erstes Programm, das auf Eingabe reagiert.

Reaktion auf zweite Taste

Möchte man nun noch eine zweite Reaktion auf eine andere Taste programmieren, dann holt man sich wieder aus dem Bereich **Eingabe** die “Klammer” :

Wenn Knopf A gedrückt

Sobald diese Klammer auf der Programm-Oberfläche liegt, verliert sie Ihre Farbe, wird “ausgegraut”.

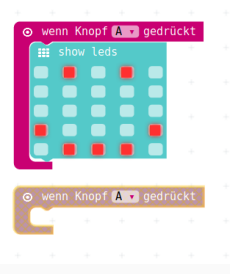


Figure 4: Augegraute Eingabe

Ausgegraute KLammern etc

Wir haben nun ein zweite “Klammer” angelegt, die ausgeführt werden soll, wenn Knopf A gedrückt wird. Das ist nicht möglich. Es kann immer nur eine Klammer als Reaktion auf einen Tastendruck im Programm-Bereich geben.

Sobald wir nun den Knopf auf Knopf B wechseln:

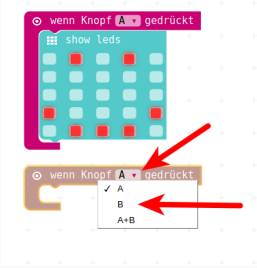


Figure 5: Auswahl anderer Knopf

bekommt die Klammer Ihre ursprüngliche Farbe wieder und kann sinnvoll verwendet den.

Verschiedene Start-Möglichkeiten

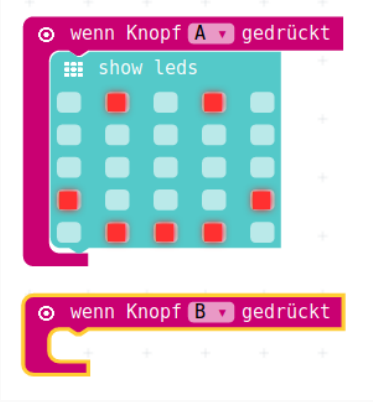


Figure 6: Auswahl anderer Knopf

Anderes Gesicht anzeigen

z.B. durch Anzeige eines anderen Gesichts

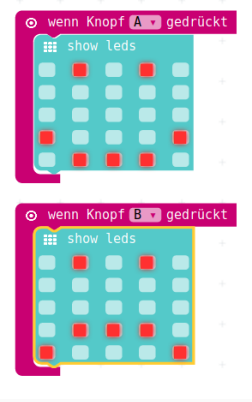


Figure 7: Auswahl anderer Knopf

Unterschied Start und Dauerhaft

Mit diesem Grundgerüst kann man nun auch sehr einfach ausprobieren, was der Unterschied im Programm-Verhalten ist, wenn man “dauerhaft” oder “Beim Start” anwählt. Dazu muss man aber kleine Pausen mit einbauen, sonst wird die Ausgabe schwer verständlich.

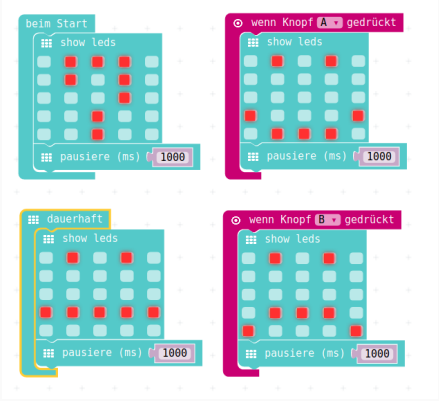


Figure 8: Unterschied Loop und Start

JavaScript-Code

Finales Eingabe-Kontroll-Programm.
Zeigt anhand von ICONs/Gesichtern das Verhalten des Programms bei Eingaben, bei Start und Dauerhaft.

Java-Script-Code

```
input.onButtonPressed(Button.A, () => {
    basic.showLeds(`
        . # . # .
        . . . . .
        . . . . .
        # . . . #
        . # # # .
        `)
    basic.pause(1000)
})
input.onButtonPressed(Button.B, () => {
    basic.showLeds(`
        . # . # .
        . . . . .
```


Download Hex-Code

Download HEX-code:

Hex-code

Lizenz/Copyright-Info

01_10_Zaehler

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Der Zähler

Bisher gelernt (1)

Bis jetzt haben wir gelernt:

- Zeichenketten (Texte) auf den LEDs als Laufschrift anzeigen
- Zahlen auf den LEDs anzeigen
- Erster Umgang mit Platzhaltern/Variablen
- Einfache Berechnung (+)
- Ausführen von Programm-Teilen einmalig beim Start
- Ausführen von Programm-Teilen immer
- Reaktion auf Tasten-Drücke : **Knopf A** und **Knopf B**


Bisher gelernt (2)

Damit sollten wir eigentlich nun in der Lage sein, einen sehr einfaches Zählprogramm zu bauen. Es soll:

- Beim Start: einen Platzhalter (z.B. mit Namen **Zaehler**) mit 0 belegen
- Beim Drücken der linken Taste (**Knopf A**): vom **Zaehler** eine 1 abziehen
- Beim Drücken der rechten Taste (**Knopf B**): den **Zaehler** um 1 erhöhen
- Dauerhaft: Den Wert der Variablen **Zaehler** anzeigen


Erweitern des Programms

Wir können nun unsere “Gesichter” nach links in den Mülleimer ziehen.

Dann aktivieren wir wieder die ausgegrauten Teile von davor, indem wir sie wieder in Ihre ursprünglichen Starter einziehen:

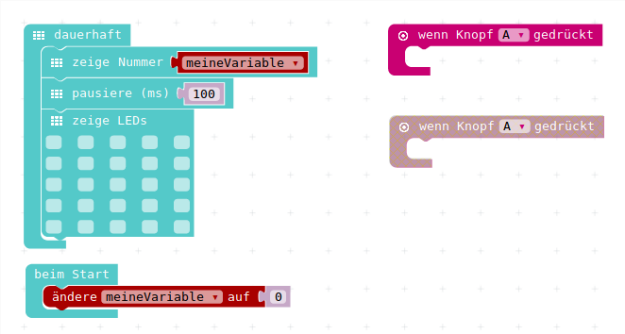


Figure 1: GrundGerüstZaehler

Die restliche Programm-Logik ist oben beschrieben und wurde schon vorher gezeigt, darum an dieser Stelle nur noch ein/zwei Hinweise und dann eine “Musterlösung”.

Platzhalter/Variablen , Achtung

Hinweis 1: Bei den Variablen befinden sich zwei sehr ähnlich klingende Befehle:

- Der eine ist die Zuweisung an eine Variable, wie wir es schon gesehen haben. Er belegt die Variable mit einem konkreten Wert, er initialisiert die Variable **AUF** den angegebenen Wert.
- Der andere Befehl nimmt den Wert in der Variable und ändert ihn **UM** den angegebenen Wert, er führt also eine Berechnung mit dem Wert durch. Diesen Befehl können wir zum Beispiel beim Drücken der rechten Taste benutzen.

Platzhalter/Variablen minus 1

Hinweis 2: Man kann einen Platzhalter um eins verringern durch folgende Rechnung: Platzhalter = Platzhalter - 1

Das ist eine “normale” mathematische Berechnung und findet sich in der Mathematik.

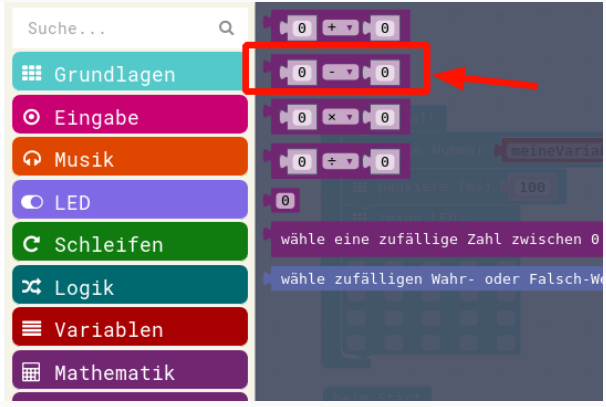


Figure 2: MinusRechnung

“Musterlösung”

Eine mögliche “Musterlösung” mit Benutzung von 2 verschiedenen Möglichkeiten für die linke und die rechte Taste ist hier

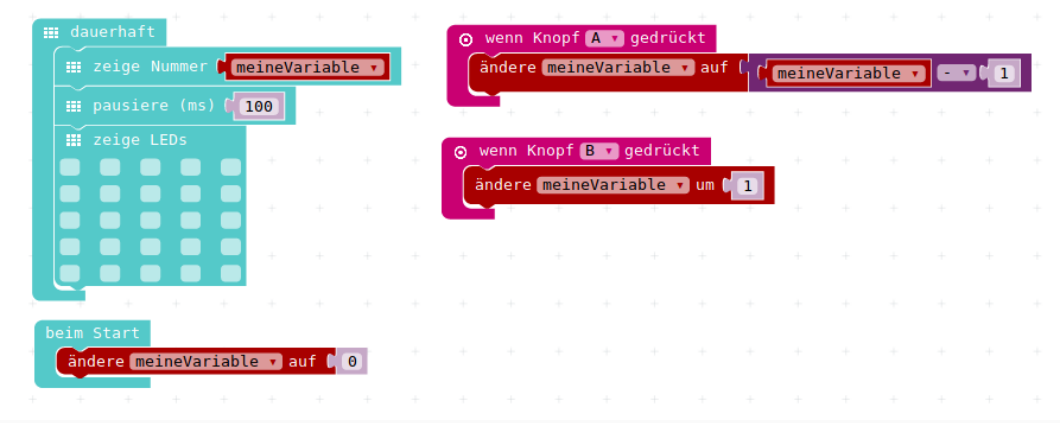


Figure 3: ZaehlerFinal

Lizenz/Copyright-Info


01_11_HexFiles_Simulator

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


HEX-Files in Simulator

- Aus dem Simulator haben wir HEX-Dateien heruntergeladen.
- Diese enthalten unser Programm
- Bislang haben wir diese HEX-Datei in den Calliope kopiert
- Damit wurde unser Programm in unseren Calliope geladen
- Anstatt dessen kann man die HEX-Dateien aber auch wieder in den Simulator laden
- Damit kann man alte Programme wieder laden und weiterbearbeiten
- Das wollen wir nun mal ausprobieren.


Projekte

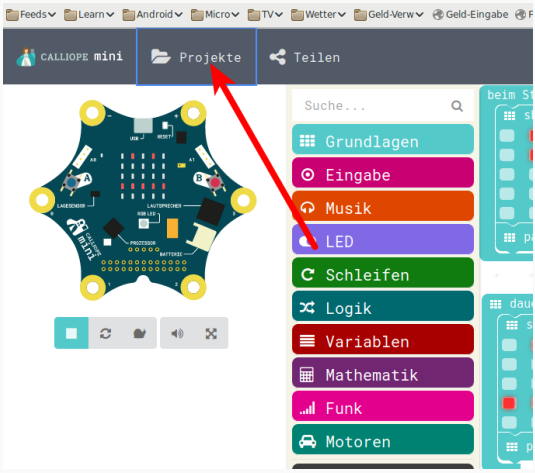


Figure 1: Projekte

Importieren

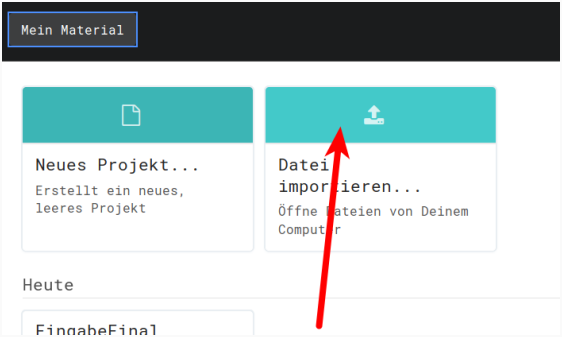


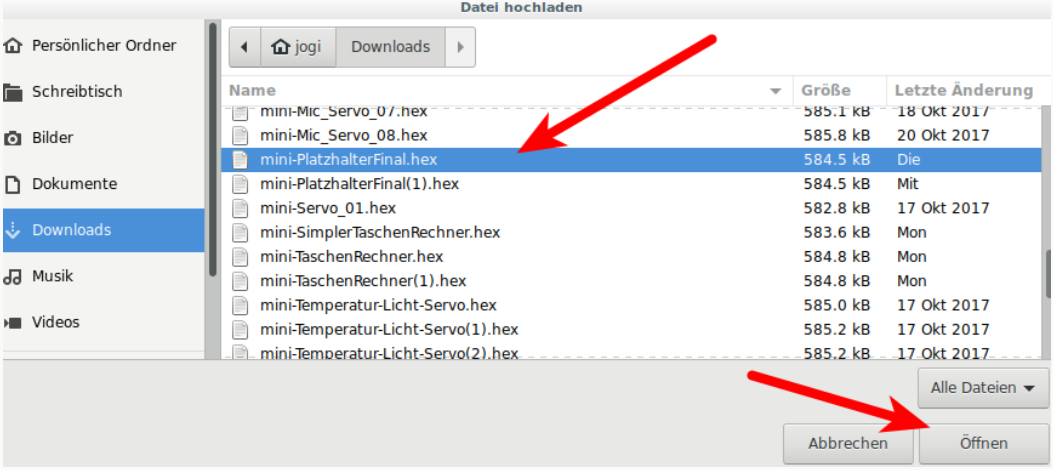
Figure 2: Importieren

Durchsuchen

Figure 3: Durchsuchen

Durchsuchen

Das alte Programm (HEX-Datei) suchen und anklicken.



Laden

Figure 4: Oeffnen Anklicken

Altes Programm ist geladen

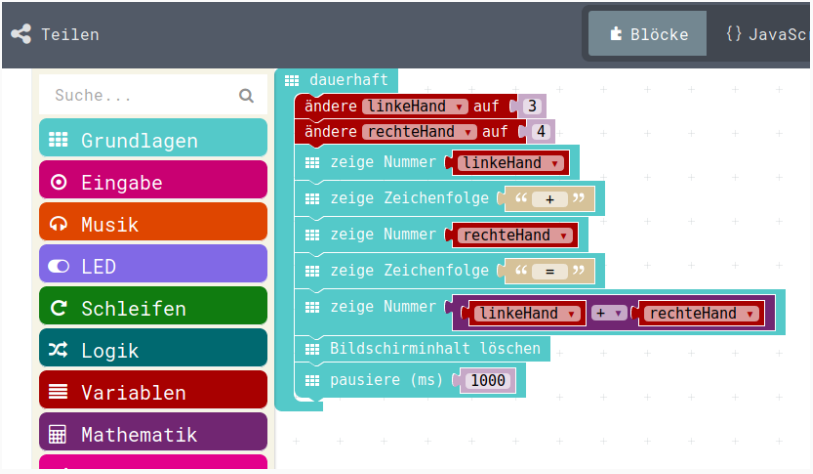


Figure 5: Altes Programm geladen

(Dieses Programm haben wir noch nicht programmiert, aber es geht hier ja ums Prinzip)

Lizenz/Copyright-Info

01_12_Taschenrechner_Komplett

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Download Hex-Code

Hex-code

Ein Mini-Taschenrechner

Ein einfacher Taschenrechner

Nun kann man die Ausgabe von Texten und von Zahlen kombinieren und einen sehr sehr sehr einfachen Taschenrechner programmieren.

Die Addition im Menu finden

Die Addition befindet sich - wie bei den Schulfächern - bei der Mathematik

Zahlen addieren im Menu Mathematik


Das vollständige Rechenprogramm

Nun ergibt eine Kombination der gerade gezeigten Zahlen-Ausgabe mit der Addition und der vorher gezeigten Zeichenketten-Ausgabe einen kleinen “Taschenrechner”.

Zahlen addieren


JavaScript-Code / Hex-Code

JavaScript-Code
Java-Script-Code

```
basic.forever(() => {
    basic.showString("2 + 5 =")
    basic.showNumber(2 + 5)
    basic.clearScreen()
    basic.pause(1000)
})
})
```

Download Hex-Code
Hex-code

Einschränkungen

Naja, das Rechnen ist doch etwas umständlich, der Calliope kann so nur genau eine Rechnung durchführen.
Aber immerhin.
Weiter geht es dann mit einem “echten” kleinen Rechner.


Erklärung (1)

Wenn man den Taschenrechner etwas universeller haben will, dann muss man mehr Flexibilität haben.
D.h man muss mit verschiedenen, unterschiedlichen Werten arbeiten können.
Anstatt den Programm-Code immer abzuändern und die Berechnung jedesmal neu zu übersetzen, brauchen wir sogenannte Platzhalter. Platzhalter nennt man beim Programmieren auch Variablen, weil die Platzhalter unterschiedliche Werte, variable Werte aufnehmen kann.
Das kann man sich z.B. beim einfachen Zählen mit zwei Händen vorstellen:

Hier ist jetzt der Variable-Einschub schon gemacht

Verwendung des Platzhalters

Figure 1: Menu-Verwenden

Verwendung der Variablen Ausgaben und Berechnungen anstatt fester Werte.

Benutzung Platzhalter

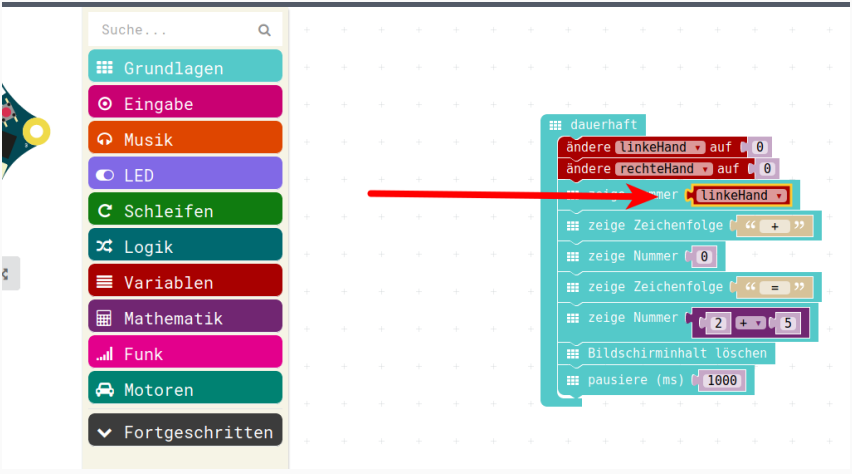


Figure 2: Menu-Verwenden-2

Dazu zieht man die Variable/den Platzhalter genau an die Stelle an der vorher feste Werte benutzt wurden.

PXT-Code

Finales simples Taschenrechner-Programm.
Immernoch muss Programm-Code geändert werden um eine neue Berechnung durchzuführen, aber die Werte für die Berechnung müssen nur an einer zentralen Stelle geändert werden. Dank Verwendung von Platzhaltern/Variablen passt sich die Ausgabe jeweils entsprechend an.

JavaScript-Code / HEX-Code

JavaScript-Code
Java-Script-Code

```
let rechteHand = 0
let linkeHand = 0
basic.forever(() => {
    linkeHand = 3
    rechteHand = 4
    basic.showNumber(linkeHand)
    basic.showString(" + ")
    basic.showNumber(rechteHand)
    basic.showString(" = ")
    basic.showNumber(linkeHand + rechteHand)
    basic.clearScreen()
    basic.pause(1000)
})
```

Download Hex-Code
Hex-code



02_01_Auffrischen

Calliope-Kurs Kinder

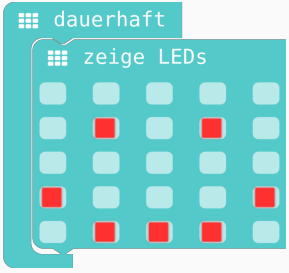
Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Wiederholung / Auffrischen

Die Endlosschleife

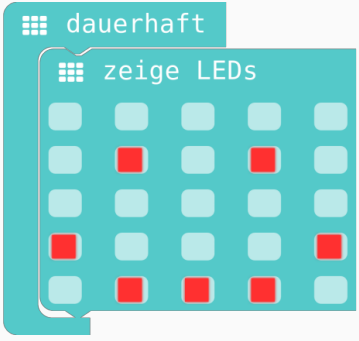
Die Endlos-Schleife, wird wie Ihr Name auch schon sagt, dauerhaft oder endlos ausgeführt.

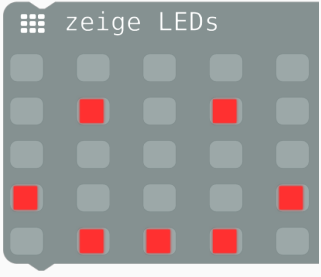


Sie ist normalerweise das Programm-Teil, in den man das Programm komplett einhängt. **Aber Achtung:** Wenn am Anfang der Endlosschleife z.B. Zahlen festgelegt werden und diese Zahlen sich weiter unten im Programm ändern, dann werden diese Zahlen beim nächsten Ausführen der Schleife wieder überschrieben.

“Ausgegraut”

- Blöcke die nicht funktionieren, weil sie in keiner “Ausführ-Klammer” eingeklickt sind, sind “ausgegraut”.
- Sie werden nicht ausgeführt.
- Sobald ein Block in einer “Ausführklammer” eingeklickt ist, erhält er seine normale Farbe und wird dann auch ausgeführt.





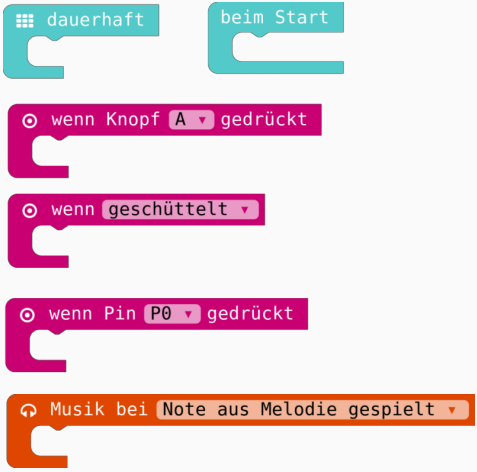
Andere Startmöglichkeiten (1)

Neben der Endlos/Dauerhaft-Schleife gibt es noch einige andere Möglichkeiten, Programm-Blöcke zu starten/ausgeführen.

Gemeinsamkeit :

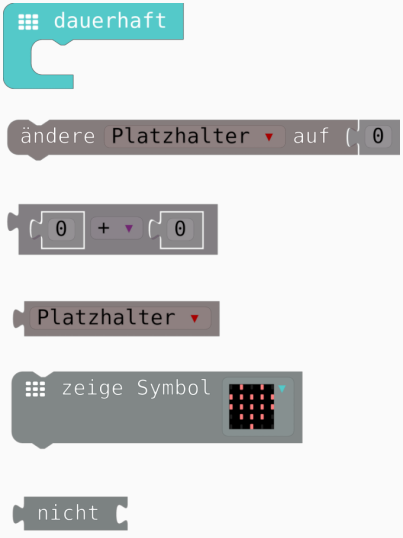
- Alle sind als "Klammern" dargestellt, sie umklammern die Programm-Blöcke die ausgeführt werden.
- Aber Achtung: Nur bei der Dauerhaft-Endlos-Schleifen-Klammer, werden die Programm-Teile dauerhaft ausgeführt.
- Bei allen anderen Start-Möglichkeiten, werden die Programmblöcke nur einmal ausgeführt.


Andere Startmöglichkeiten (2)



Puzzleteile: Nicht alles passt

- Wenn man sich die Programm-Blöcke etwas genauer anschaut,
- dann sehen die ein bisschen aus wie Puzzleteile.
- Das ist Absicht und soll zeigen,
- dass nicht alle Programm-Blöcke an allen Stellen “eingeklickt” werden können.



Arbeitsbereich : Menu

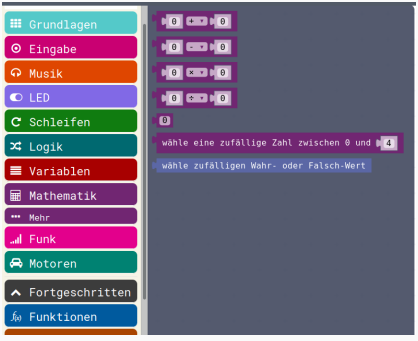
- In der Mitte des Bildschirms ist das Menu.
- Von dort schiebt man sich die Blöcke nach rechts auf den Arbeitsbereich.
- Im Menu gibt es unterschiedliche Kategorien, diese sind unterschiedlich gekennzeichnet.
- Die Programm-Blöcke in diesen Menus haben auch immer die Farbe der jeweiligen Kategorie.

Zeichenfolgen <=> Zahlen

- Beim Ausgeben auf unserem Mini-Bildschirm können wir unterscheiden zwischen Zeichenfolgen und Zahlen.
- Mit Zeichenfolgen kann man Buchstaben, Wörter oder auch ganze Sätze ausgeben.
- Man kann auch einzelne Zahlen ausgeben
- Grosser Unterschied:
  - Mit den Zahlen kann man rechnen
  - Mit den Zeichenfolgen kann man nicht rechnen

Zahlen = Mathematik

Die Rechnungen, die wir benutzen (Addition und Subtraktion) befinden sich alle unter : **Mathematik**



Die Art der Rechnung kann man im Nachhinein auch immernoch durch Klick in das Dreieck ändern!

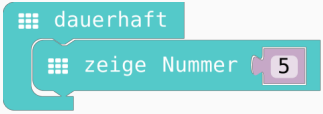
Platzhalter (1)

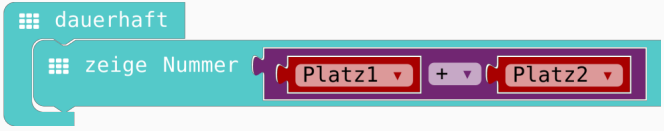
- Erste Rechnungen (aus der Mathematik) mit festen Werten.
- Die verwendeten Zahlen sind dabei festgelegt: **fest** , **konstant**.
- d.h. Wenn das Programm etwas anderes berechnen soll, dann muss ein neues Programm geschrieben werden.
- da das Übersetzen “nebendran” erfolgt, fällt das gar nicht so auf.
- Wenn das Programm aber jeweils auf den Calliope übertragen werden soll, merkt man das
- Man kann den Calliope dann auch nur mit **EINEM** Programm, mit einer Berechnung mitnehmen.
- Vergleich : Taschenrechner, der nur eine Berechnung kann, für jede andere Berechnung muss man einen neuen kaufen. . .


Platzhalter (2)

- **Platzhalter** lösen dieses Problem.
- Sie können unterschiedliche Werte annehmen
- Sie sind damit **veränderbar**
- Veränderbar = **variabel**
- => Platzhalter = **Variable**


Platzhalter (3)

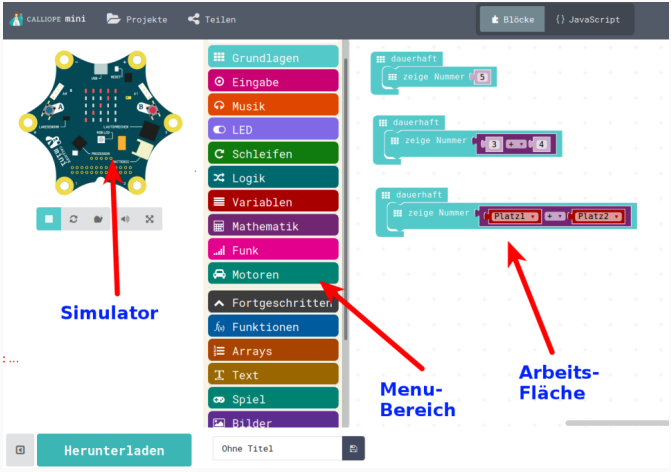




Die Entwicklungs-Umgebung

- Unterscheidung **Simulation** und echter Calliope
- Der simulierte Calliope ist links in der Entwicklungs-Umgebung
- Bei Download des Programms (als Hex-Datei) wird diese - wie eine Datei aus dem Internet - im **Downloads-Ordner** gespeichert
- Der Calliope ist wie ein USB-Stick
- Beim Anstecken wird ein Datei-Explorer geöffnet
- Dann kann man die Datei, das HEX-Programm, vom Download-Ordner in den Calliope kopieren.

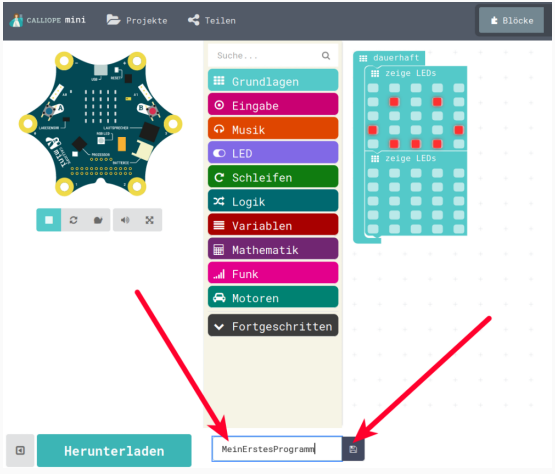
Der Arbeits-Bereich



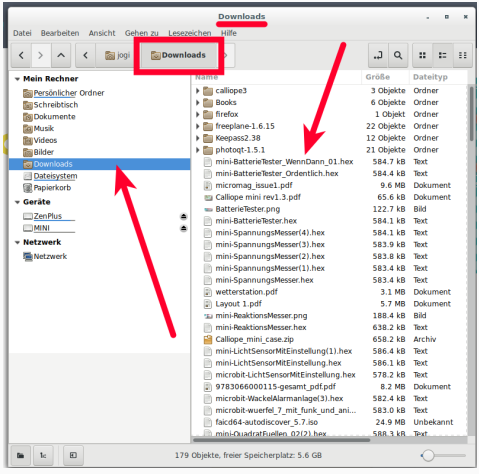
Speichern und Übertragen auf Calliope (1)

- Das erstellte Programm läuft automatisch im Simulator.
- Soll es im echten Calliope laufen, muss es zuerst heruntergeladen werden
- So wie Filme/Bilder/Texte etc die im Browser aus dem Internet herunter geladen werden, landet auch die **HEX**-Datei im Verzeichnis **Downloads**
- Beim Anstecken des Calliope wird dieser vom Computer wie ein USB-Stick behandelt, er taucht wie ein USB-Stick im Datei-Explorer auf
- Um das erstellte Programm auf den Calliope zu bringen, um dort ausgeführt zu werden, muss die HEX-Datei vom Download-Verzeichnis auf dem Computer auf den "USB-Stick" namens **MINI** kopiert werden.

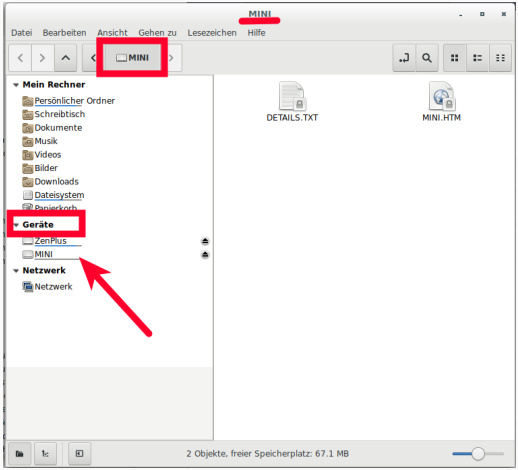
Speichern und Übertragen auf Calliope (2)



Speichern und Übertragen auf Calliope (3)



Speichern und Übertragen auf Calliope (4)



Lizenz/Copyright-Info

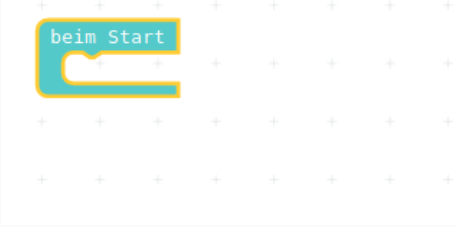
02_02_Auffrischen_DoIt

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

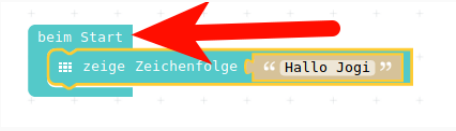
Frühjahr 2019


Beim Start



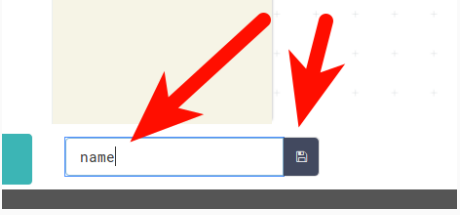
Beim Start

Zeige Zeichenfolge

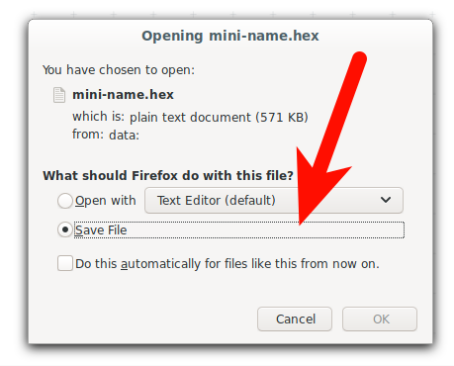


Zeige Zeichenfolge

Speichern unter

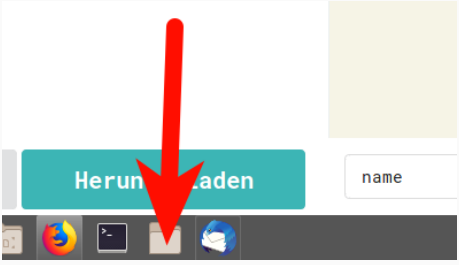


Speichern unter


Opening mini-name.hex
You have chosen to open:
mini-name.hex
which is: plain text document (571 KB)
from: data:
What should Firefox do with this file?
Open with
Text Editor (default)
Save File
Do this automatically for files like this from now on.
Cancel
OK

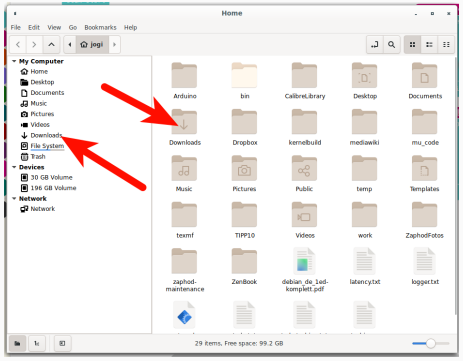
Safe File

Dateimanager



Dateimanager

Downloads Verzeichnis

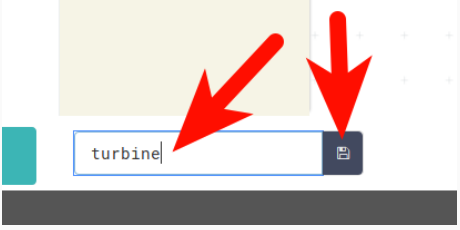


Verzeichnis Downloads

Dauerhaft

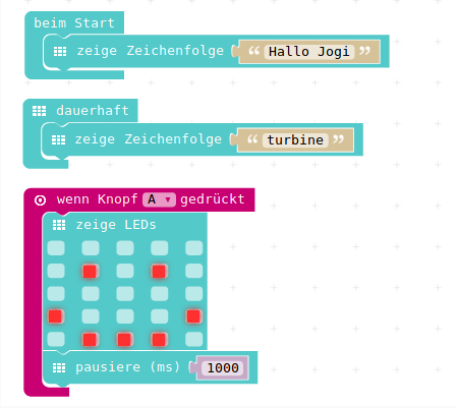
- Aus dem **Grundlagen**- Menu
- Holen wir uns einen **Dauerhaft**-Start-Block
- und holen uns dazu ein **zeige Zeichenfolge**
- und schreiben einen zweiten Text, z.B. **“turbine”** rein

Speichern Turbine



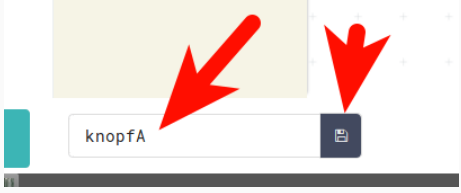
- Wir **Speichern** das Programm unter dem Namen **Turbine**
- und **laden** es in den **Calliope**

Knopf A



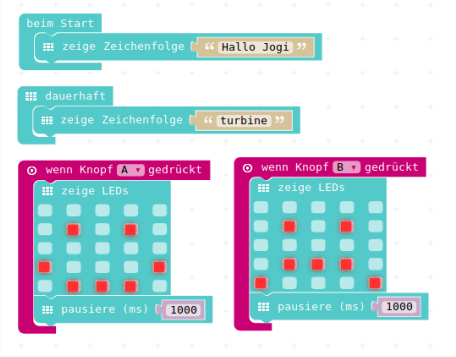
Knopf A : Zeige LEDs

Speichern Knopf A



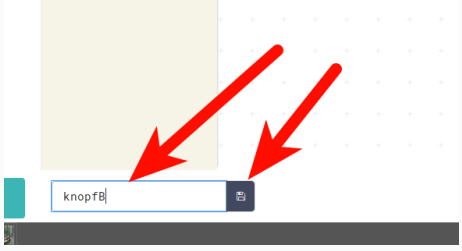
- **Speichern** des Programms als **KnopfA**
- und in den **Calliope laden**

Knopf B



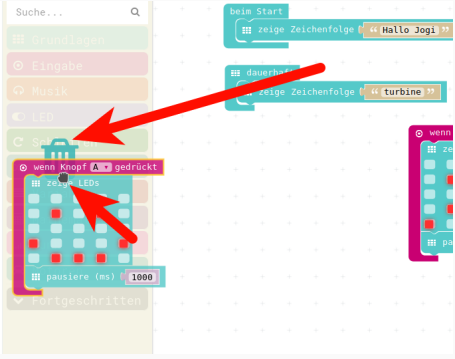
Knopf B : Zeige LEDs

Speichern Knopf B



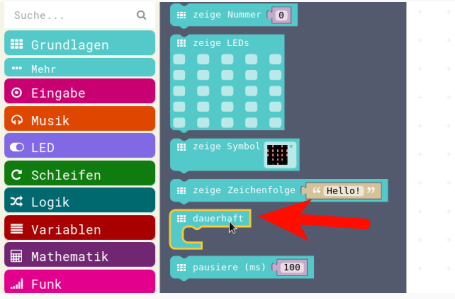
- **Speichern** des Programms als **KnopfB**
- Und in den **Calliope laden**

Aufräumen : Mülleimer



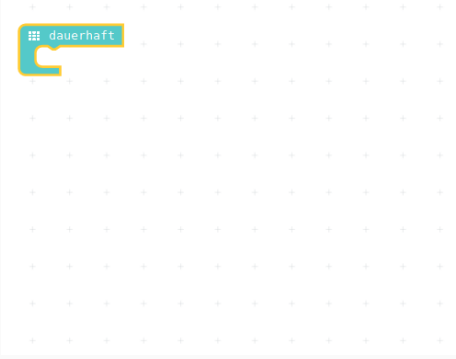
- da wir **ALLES** gesichert haben
- und wissen wie wir das wieder herstellen können
- (Siehe Tag 1 **Kapitel 11**
- Können wir das **ALLES**
- in den **Mülleimer** ziehen

Dauerhaft-Schleife



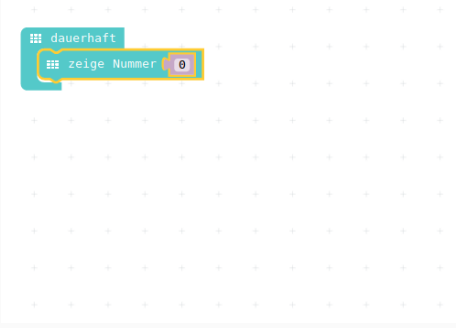
- Aus dem Menu **Grundlagen**
- Holen eines **Dauerhaft** - Start-Blocks

Dauerhaft



- **Dauerhaft** auf dem leeren Arbeitsbereich

Zeige Nummer

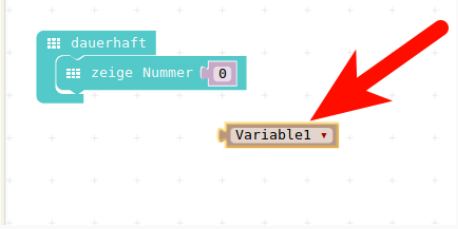


Zeige Nummer

Variablen anlegen

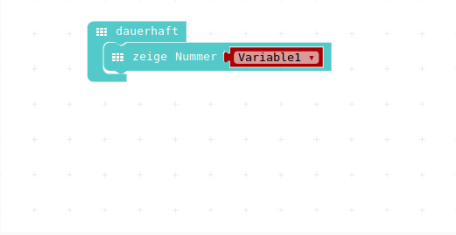
Variablen anlegen

Variable in Arbeitsbereich



Variable in Arbeitsbereich

Variable in Start-Block

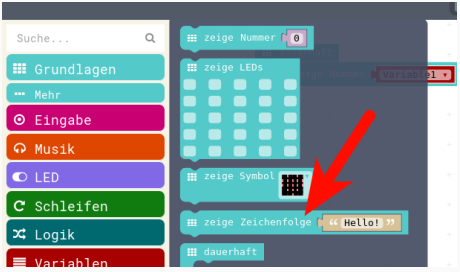


Variable in StartBlock

Speichern als Variable1

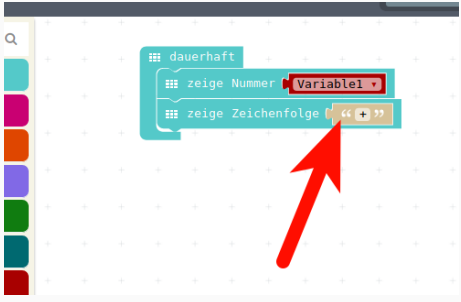
Speichern Variable 1

Zeichenfolge



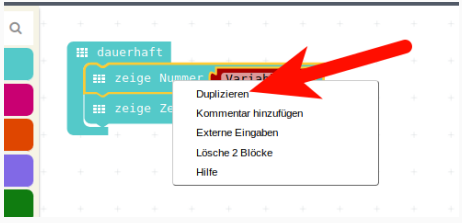
zeige Zeichenfolge aus Menu

+ - Zeichen



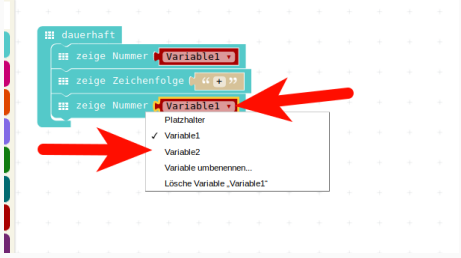
+ Zeichen machen

Duplizieren rechte Maustaste



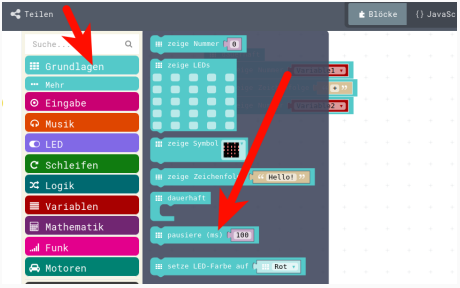
Duplizieren rechte Maustaste

Variable umbenennen “Dreieck”



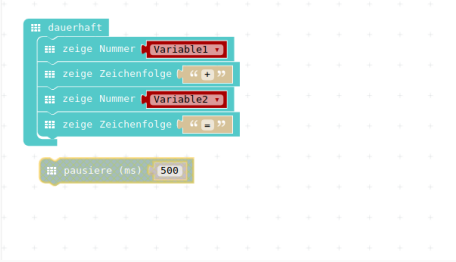
Variable umbenennen mit Dreieck rechts des Variablen-Namens

“pausiere ms” holen



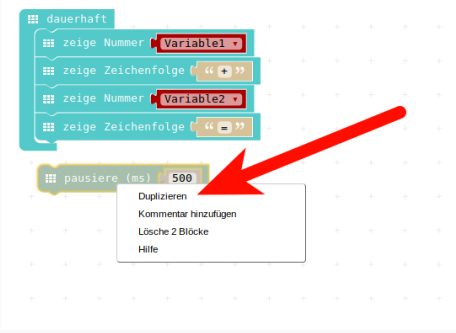
- **pausiere ms** holen und auf den **Arbeitsplatz** legen
- Wert auf **500 ms** ändern

Zeichenfolge =



- **Zeichenfolge** entweder aus obigem “+” kopieren
- oder aus Menu **Grundlagen** holen
- Auf **Gleichheits-Zeichen** “=” abändern
- unten **einklicken.**

“pausiere ms” 4 mal Duplizieren



- Mit **rechter Maustaste** auf “**pausiere ms 500**”
- **4 mal kopieren**

4 mal kopiert

- **Pausieren** ist 4 mal **kopiert**

4 mal eingefügt

- **Pausieren** ist 4 mal **eingefügt**

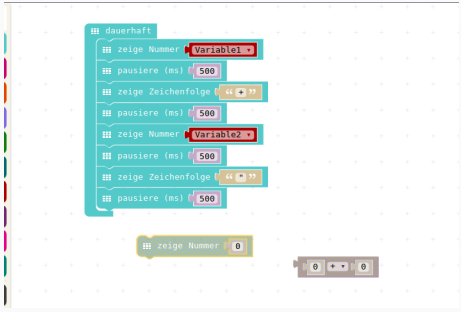
Addition holen aus Mathematik

- Aus dem Menu **Mathematik**
- Holen einer **Addition**

Addition auf Arbeitsbereich

- **Addition** liegt auf dem Arbeitsbereich

Zeige Nummer (Grundlagen-Menu)

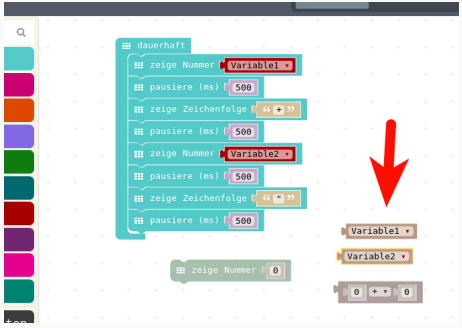


- **Zeige Nummer** aus dem **Grundlagen**-Menu holen
- und auf Arbeitsbereich legen

Zwei Variablen anlegen

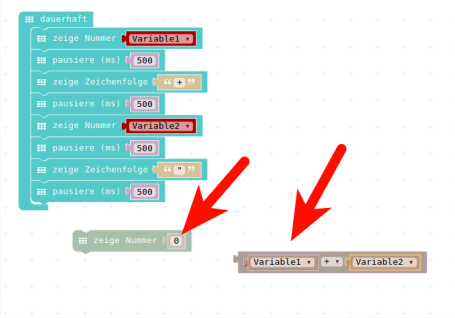
- Im Menu **Variablen** zwei Variablen :
- **Variable1** und
- **Variable2** anlegen

Variablen im Arbeitsbereich



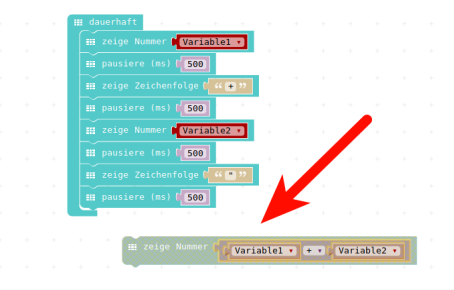
- Beide **Variablen**
- sind in den **Arbeitsbereich** gezogen

Variablen in die Addition eingezogen



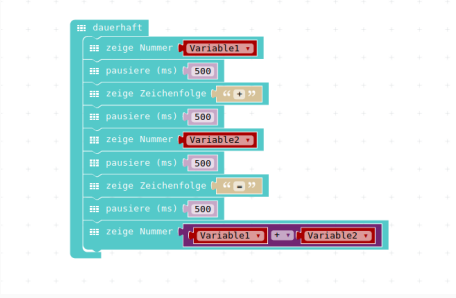
- Beide **Variablen**
- wurden in die **Addition** reingezogen

Addition in “Zeige Nummer”



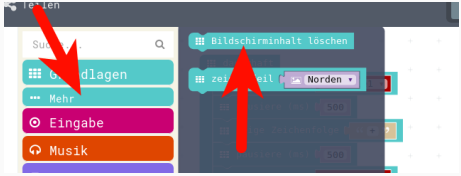
- **Addition** mitsamt den **Variablen**
- wurde in **Zeige Nummer** eingeklickt

Zeige Nummer angehängt



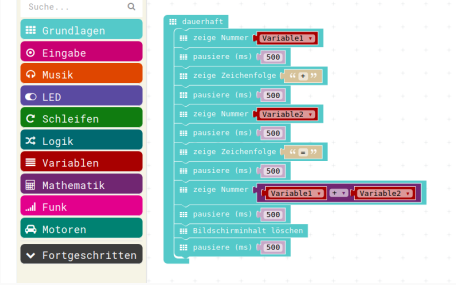
- **“Zeige Nummer”**
- wurde unten angehängt

Bildschirminhalt löschen



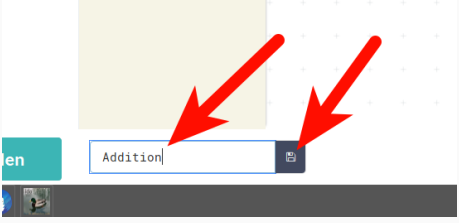
- Aus dem Menu **Grundlagen** -> **Mehr**
- den Befehl **Bildschirminhalt löschen** holen
- und noch zweimal "**pausiere ms**" darum

Addition Teil 1 fertig



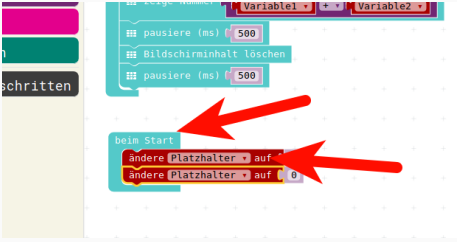
- unser Rechner **Addition Teil 1** ist fertig

Speichern als Addition



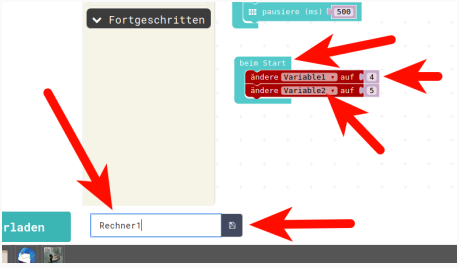
- Speichern als **Addition**

Beim Start Variablen belegen



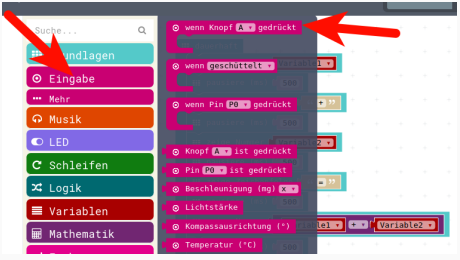
- Aus dem **Grundlagen**-Menu :
- **Beim Start** holen
- Aus dem Menu **Variablen** :
- **Ändere Platzhalter auf 0** holen

Variablen ändern



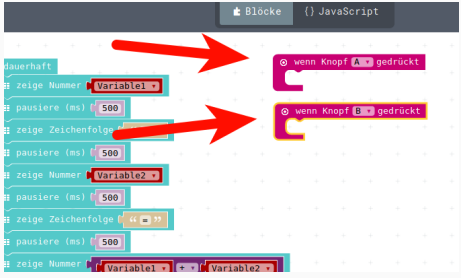
- Variablen-Namen auswählen auf **Variable1** und **Variable2**
- (Klick auf das **Dreieck** neben dem Namen der Variable)
- **Werte** auf **4** und **5** ändern
- Programm unter dem Namen **Rechner1** abspeichern
- Programm in den **Calliope laden**

Knopf A gedrückt



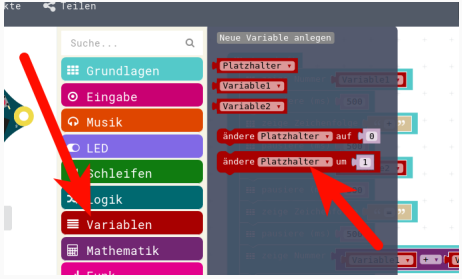
- Aus dem Menu **Eingabe**
- zweimal **wenn Knopf A gedrückt**
- auf die **Arbeitsfläche** holen

Auf Knopf B ändern



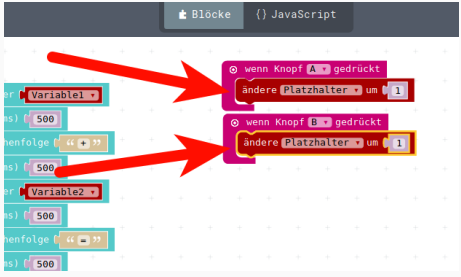
- Den einen Befehl auf:
- **Wenn Knopf B gedrückt** ändern

Ändere Platzhalter UM



- Aus dem Menu **Variablen**
- 2 mal den Befehl :
- **ändere Platzhalter um 1**
- holen und in die beiden
- **Knopf-Blöcke** einrasten

Ändere Platzhalter um

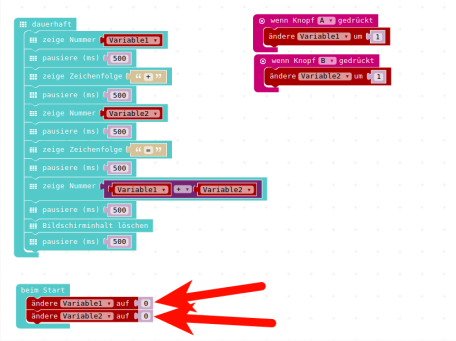


- **Ändere Platzhalter um 1**
- in den beiden **Knopf gedrückt** - Blöcken

Variablen verwenden

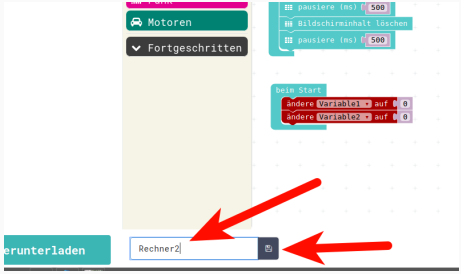
- Mittels Klick auf das **Dreieck** neben dem Variablen-Namen
- die beiden richtigen Variablen :
- **Variable1** und **Variable2** auswählen

Beim Start auf 0



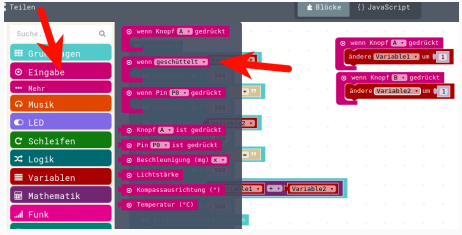
- in **Beim Start**
- die beiden **Variablen** auf **0** setzen

Rechner 2



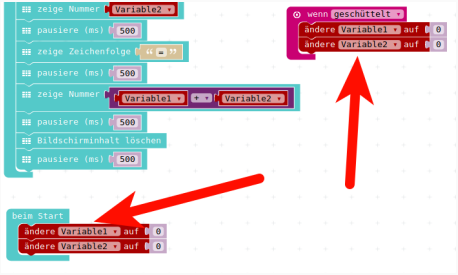
- Das Programm als **Rechner2** abspeichern
- und in den **Calliope laden**

Wenn geschüttelt



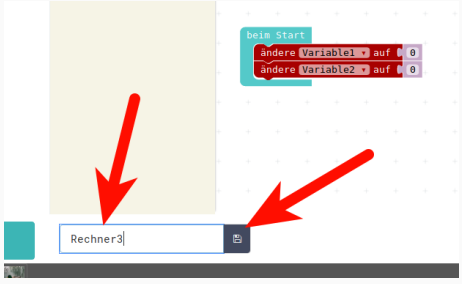
- Aus dem Menu **Eingabe**
- holen wir uns **wenn geschüttelt**
- in den Arbeitsbereich

Kopieren auf 0-Setzen



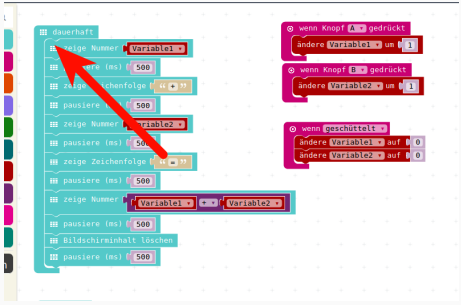
- Aus dem **beim Start** - Block
- **kopieren** wir uns die beiden:
- **ändere Variable auf 0** - Befehle
- und **klicken** sie in den :
- **wenn geschüttelt** - Block

Rechner 3



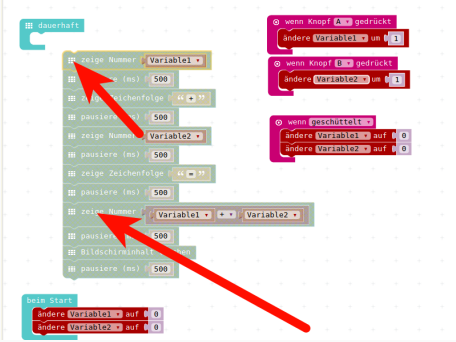
- Wir speichern das Programm :
- Als **Rechner3**
- und laden es in den Calliope

Umwandlung in Auf/Ab-Zähler



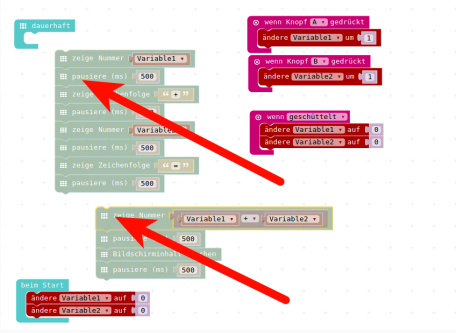
- Alles innerhalb des **dauerhaft** -Startblocks
- herausziehen

Herausgezogen



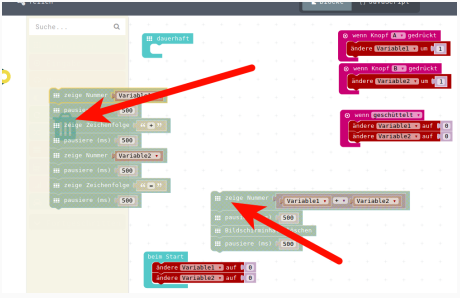
- Unten ab **zeige Nummer**
- auch noch wegziehen

Oberer Teil : Mülleiemer



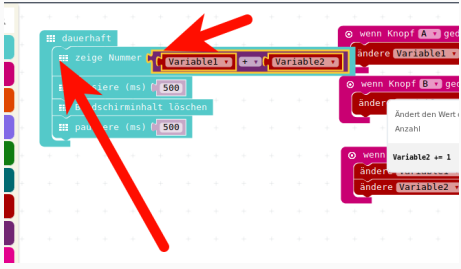
- den ganzen oberen Teil ab
- **zeige Nummer** bis
- **pausiere ms** 500 kann direkt in den Müll gezogen werden

Oberer Teil : gelöscht



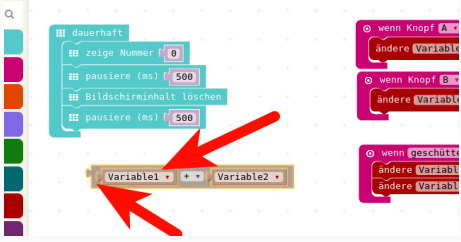
- der ganze obere Teil wurde in den **Mülleimer** gezogen
- der untere Teile ab **zeige Nummer** wird wieder
- in den **dauerhaft** Block eingeklickt

Rausziehen Addition



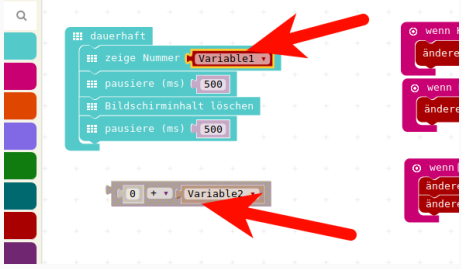
- Nun wird die **Addition** nochmal rausgezogen

Variable zurückschieben



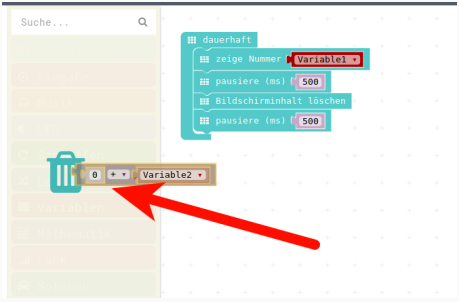
Selection_047.png

Variable zurück, Addition : Müll



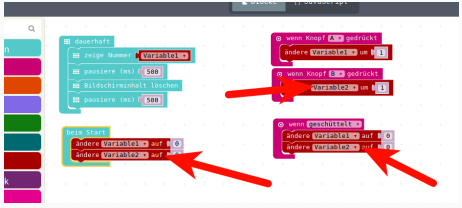
- Die Variable **Variable1** ist wieder in
- **zeige Nummer**
- Die **Addition** kann in den Müll

Addition im Müll



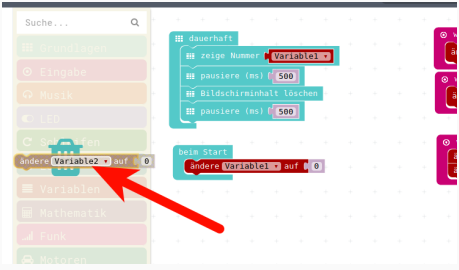
- **Addition** wird in den **Müll** geschoben

Variablen anpassen/löschen



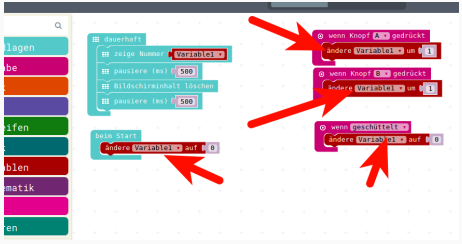
- wir wollen nur noch **Variable1** haben
- **beim Start** : den Teil mit Variable2 löschen
- **wenn geschüttelt** :den Teil mit Variable2 löschen
- **wenn Knopf B** : Variable auf **Variable1** ändern

Unnötiges löschen



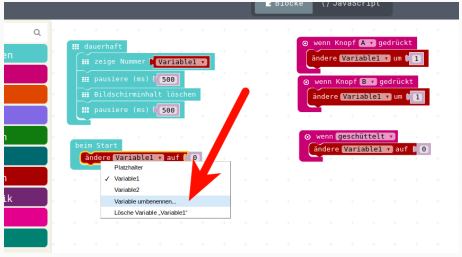
- Alles unnötige kommt in den **Müll**

Nur noch Variable 1



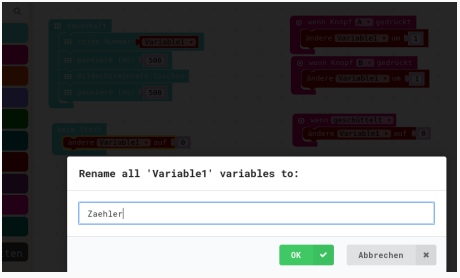
- Nun ist nur noch **Variable1** benutzt

Umbenennen



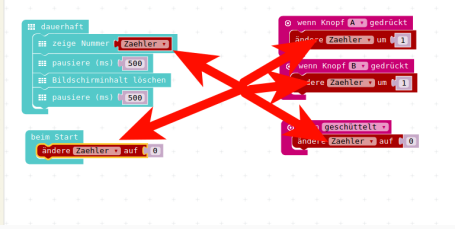
- Durch Klick auf das **Dreieich**
- Kann die Variable **umbenannt** werden

Neuer Namen: Zaehler



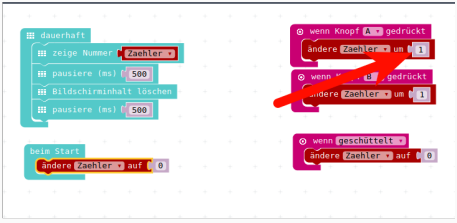
- umbenennen in **Zaehler**

Alle Verwendungen umbenannt



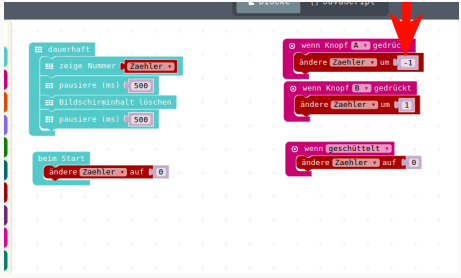
- Durch das Umbenennen sind alle **Variable1**
- umbenannt auf **Zaehler**

Knopf A : eins abziehen



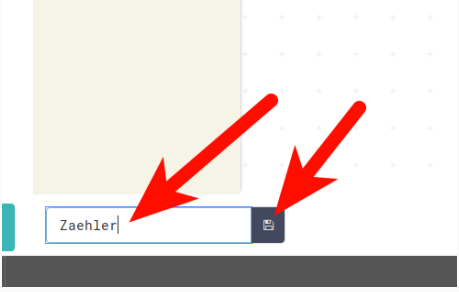
- **ändere Zaehler um -1**

-1 = eins abziehen



- bei **ändere Zaehler um**
- in das Zahlenfeld eine **-1** eingeben
- Fertig ist der Auf/Ab-Zähler

Speichern



- Speichern als **Zaehler**
- vom Download-Ordner auf den **Calliope** programmieren

02_03_Elektronik_Spannungsquelle

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Elektronik Grundlagen

Spannungsquelle (1a)

Eine Spannungsquelle die Ihr alle kennt, ist die Steckdose.

Spannungsquelle (1b)

ACHTUNG : Die Steckdose ist der falsche Weg, um mit Elektronik und Spannung zu experimentieren!

Die Gründe dafür sehen wir gleich.

Spannungsquelle (2)

Also nehmen wir etwas, das Ihr auch alle kennt, das aber etwas ungefährlicher ist und Ihr vermutlich von Taschenlampen etc kennt: **Batterien**

Batterien gibt es in unterschiedlichen Ausführungen.

Spannungsquelle (3)

Diese Batterien unterscheiden sich jedoch nicht nur in der Grösse und den Bauformen, sie unterscheiden sich auch in den **Spannungen** , die diese Batterien liefern.

- Spannung wird angegeben in **Volt** .
- Das ist die sogenannte **Einheit** .

Vergleiche zum Beispiel :

- Entfernungen werden in Meter (**m**) oder Kilometer (**km**) angegeben,
- Gewicht wird in Gramm (**g**) oder Kilogramm (**kg**) gemessen bzw. angegeben.

So sagt man :

- die Einheit der Entfernung ist Meter,
- die Einheit des Gewichts ist Gramm
- die Einheit der Spannung ist Volt.

Spannungsquelle (4)

Die Spannungen von normalen, handelsüblichen Batterien, wie Ihr sie hier seht, reichen von 1.5 Volt bis zu 9 V, eine davon hat sogar 12 V!

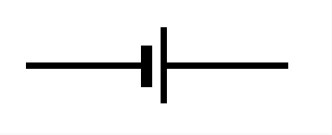
Die Batterie mit 9V liefert damit immerhin schon das 6 fache der kleinsten Batterie, die Batterie mit 12 V sogar das 8-fache!!

Zwei Pole/Anschlüsse (1)

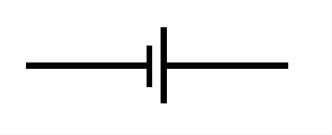
- Die Spannungsquellen, die wir anschauen, haben immer zwei Anschlüsse, auch Pole genannt.
- Der eine Anschluss wird Plus-Pol genannt (+),
- der andere Anschluss wird Minus-Pol geannt (-).

Zwei Pole/Anschlüsse (2)

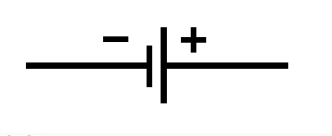
Als Schaltbild-Symbol sieht man Batterien oft so:



oder so:



oder so:



Zwei Pole/Anschlüsse (5)

- Der längere Anschluss am Batterie-Symbol ist **immer** der Plus-Pol.
- Normalerweise ist bei Kabeln der **Plus-Pol** immer **ROT**
- Der **Minus-Pol** wird entweder mit **BLAUEM** oder **SCHWARZEM** Kabel angezeigt.

Spannung fühlen? (1)

Kann man Spannungen fühlen? **JA** man kann! Allerdings sind glücklicherweise die Spannungen mit denen wir arbeiten, so gering, dass wir sie nur mit Tricks fühlen können

Wer traut sich?

- 1.5 V Batterie mit Hilfskabel oder ähnlichem an die Zunge
- 4.5 V Batterie mit der Zunge an die Pole
- 9V Batterie mit der Zunge an die Pole.

Wie man merkt, je grösser die Spannung um so mehr prickelt es.

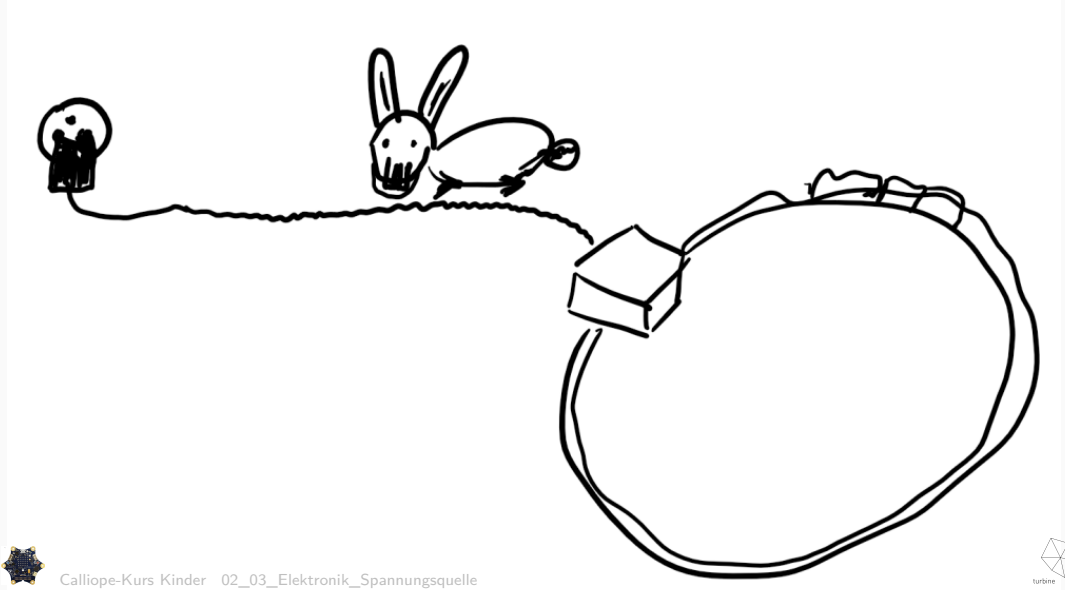
Spannung fühlen? (2)

Was passiert, wenn man grössere Spannungen versucht zu fühlen?

- Zumindest jedes Kind hier in der Schweiz kennt Globi.
- Da gibt es ein tolles Bild bei **Globi bei der Feuerwehr**.
- Leider hat mir der Globi-Verlag nicht erlaubt, das hier abzubilden, drum habe ich es halt selbst gemalt.
- Wie Ihr seht, bin ich kein besonders guter Zeichner...
  (Ein bisschen besser kann ich schon noch, wenn ich mir etwas mehr Mühe gebe...)

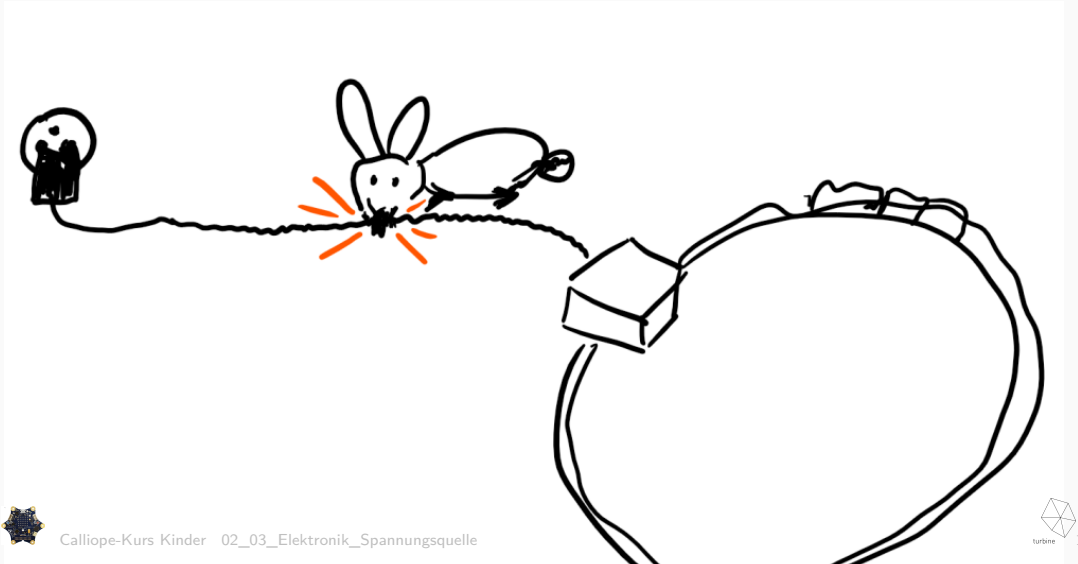
Spannung fühlen? (3)

Ein kleiner Hase nagt an der Strom-Zuleitung (220 V) für einen Eisenbahn-Trafo:



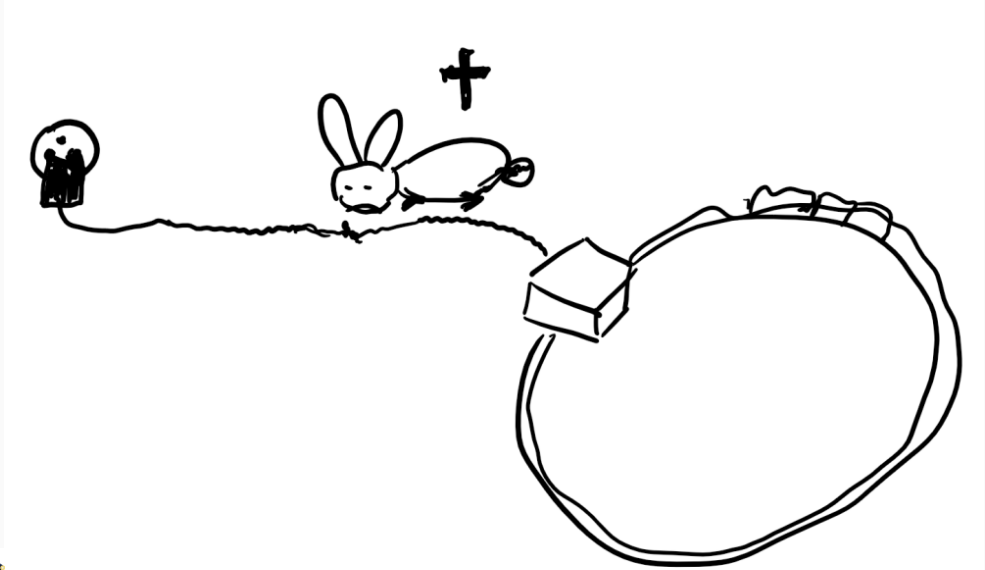
Spannung fühlen? (4)

Beim Durchbeissen gibt es einen Blitz (und ein Feuer, das dann Globi als Feuerwehrmann löschen muss)



Spannung fühlen? (5)

Nachdem das Feuer gelöscht wird, ist der Hase **tot** ! :



Vergleich : Gewicht auf dem Kopf tragen (1)

- An der Steckdose sind hierzulande **220 Volt**,
- das ist fast das **150-fache** von unserer **1.5 V** Batterie!

Vergleich:

- Ihr versucht etwas auf dem Kopf zu tragen

Vergleich : Gewicht auf dem Kopf tragen (2)

1 kg Mehl auf dem Kopf tragen

Ein **150 kg** schwerer Mann auf dem Kopf

(https://pixabay.com/de/sumoringer-athlet-ringer-sport-3196755 , CC0 Creative Commons)

Vergleich : Gewicht auf dem Kopf tragen (3)

- So wie Ihr ein Kilo Mehl locker auf dem Kopf balancieren könnt sind auch kleine Batterien normalerweise ungefährlich.
- Aber ein 150kg-Mann macht Euch **platt** !
- Ebenso sind Steckdosen auch 150 mal stärker und damit **tabu**!
- Was bei Steckdosen passieren kann, seht Ihr oben bei Globi.
- Steckdosen sind lebensgefährlich!
- Darum: **FINGER WEG von der STECKDOSE!**

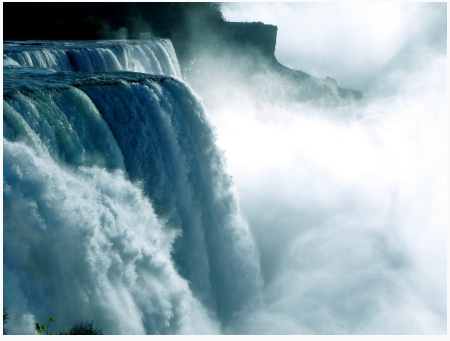
Vergleich mit Wasser

Beim elektrischen Strom wird oft ein Vergleich mit Wasser gemacht.

- Strom fliesst, Wasser fliesst
- Spannung “fällt ab”, Wasser fällt
- usw

Wenn man diesen Vergleich bildlich verwenden will, dann kann man das evt mit Wasserfällen machen.

Sehr Grosse Spannung (1)



(https://pixabay.com/de/niagarafälle-wasserfall-wasserkraft-218591 , CC0 Creative Commons)

Sehr hoher Wasserfall

Sehr Grosse Spannung (2)

(https://pixabay.com/de/stromleitungen-energie-stromleitung-804880 , CC0 Creative Commons)

Sehr grosse Spannung!

Eine typische Freiland Hochspannungs-Leitung hat 110 kV = **110 000** = **110 Tausend Volt.**

Das ist 500 mal so viel wie in der normalen Steckdose, die schon **tödlich** ist !

Grosse Spannung (1)

(https://pixabay.com/de/kaskade-island-landschaft-berg-1868687 CC0 Creative Commons)

Ein hoher Wasserfall, da möchte man nicht mit dem Boot runterfallen.

Grosse Spannung (2)

Grosse Spannung, Steckdose. Zu gefährlich um damit zu experimentieren!

Niederspannung (1)

(https://pixabay.com/de/wasserhahn-brunnen-wasserspender-1684902 CC0 , Creative Commons)

Das ist ein kleiner, “handlicher” Wasserfall.

Niederspannung (2)

Das sind Spannungen mit denen wir arbeiten, Niederspannung.

Das geht für uns von 1,5 V bis maximal 9V oder evt 12V.

Ganz gute Einführung in Strom und Spannung

Eine ganz gute Einführung in Strom und Spannung gibts in der Sendung mit der Maus

https://www.youtube.com/watch?v=Je22SgH8TCk

Mehr Links auf der Übersicht

Lizenz/Copyright-Info

02_04_Elektronik_Verbraucher

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Elektronik Grundlagen

Verbraucher

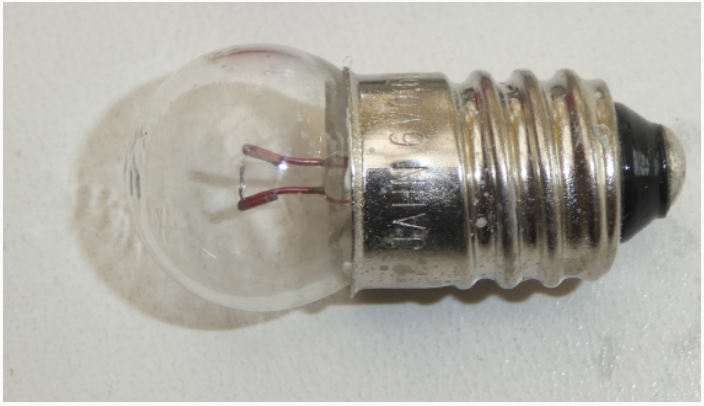
Eine Spannungsquelle alleine bringt noch gar nichts. Man will ja irgendetwas antreiben, sehen, erhitzen ... Also z.B.

- Einen Motor im Staubsauger antreiben
- Eine Lampe/Taschenlampe leuchten sehen
- Einen Toaster erwärmen
- ...

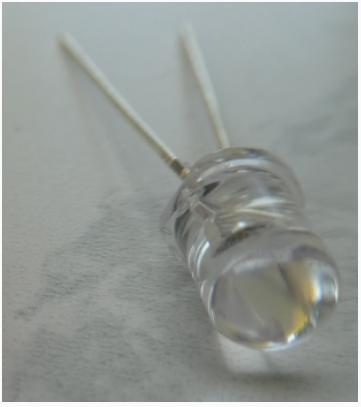
Das sind die sogenannten Verbraucher.
Sie verbrauchen den Strom, den die Batterie liefert.

Beispiele - für unsere Zwecke - von Verbrauchern:


Eine kleines Birnchen.

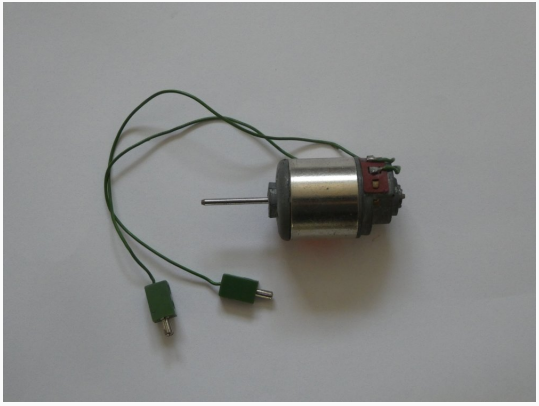


Eine einzelne LED.

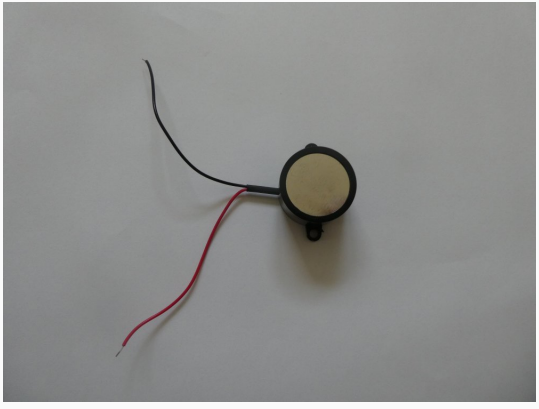


Ein kleiner Servo-Motor

Ein kleiner Motor



Ein “grässlicher” Summer



Auch zwei Anschlüsse

Wie ihr vielleicht seht, haben auch die Verbraucher normalerweise 2 Anschlüsse, der Servo-Motor hat sogar 3, da gehen wir aber jetzt nicht weiter darauf ein.

Auch bei den Verbrauchern gibt es oft die Unterscheidung zwischen + **Plus** und - **Minus**. Darauf **muss** man achten!

Manchmal funktioniert etwas einfach nicht, wenn man es falsch herum anschliesst, es kann aber auch schlimmer kommen und man macht den Verbraucher **kaputt**. Also Vorsicht beim Anschliessen der Verbraucher!

Spannungs - Angaben

So wie bei den Spannungs-Quellen, die es in sehr unterschiedlichen Grössen gibt, muss man auch bei den Verbrauchern auf die Spannungs-Angaben und auf die Grössen achten. Diese Lampen hier:

sind ausgelegt für Spannungen von 220 V (Achtung, Steckdose!) bis herunter zu 3 V.

Vergleich mit Wasser

Um auch hier wieder den Vergleich mit Wasser zu bringen, können wir uns die Verbraucher als Wasser-Räder vorstellen.

So wie bei den Spannungs-Quellen, den Wasserfällen, die es in sehr verschiedenen Grössen gibt, gibt es eben auch Wasser-Räder in sehr unterschiedlichen Grössen.

Ein sehr grosses Wasserrad

(https://pixabay.com/de/japan-waterwheel-826639 , CC0 Creative Commons)

Ein normales Wasserrad

(https://pixabay.com/de/mhle-wasserrad-alt-wasserkraft-2909252 CC0 Creative Commons)

Ein kleines Wasserrädchen

(https://pixabay.com/de/wasserrad-holz-bach-modell-778801 , CC0 Creative Commons)

Lizenz/Copyright-Info

02_05_Elektronik_Stromkreis

Calliope-Kurs Kinder

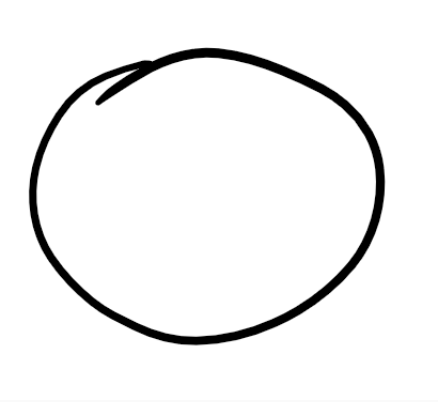
Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Elektronik Grundlagen

Der Stromkreis (1)

Im Stromkreis



(das ist natürlich kein Stromkreis, das ist ein Kreis)

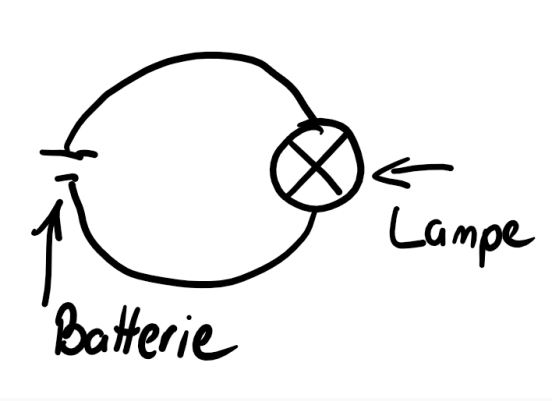
kommen nun die Spannungsquelle und der Verbraucher zusammen.

Der Stromkreis (2)

- Hier zeichnen wir links die Batterie und rechts die Lampe.
- Dafür gibt es aber keine “Vorschrift”, man kann die Batterie auch auf der rechten Seite zeichnen.
- Der Pluspol der Batterie wird mit einem Anschluss des Lämpchens verbunden,
- der Minus-Pol der Batterie mit dem anderen Anschluss.
- Bei einem normalen (altmodischen) Glühlämpchen, ist es egal, welchen Anschluss des Lämpchens man mit Plus und welchen Anschluss man mit Minus verbindet.
- Bei einigen anderen Verbrauchern ist das nicht egal, darauf kommen wir später noch zurück.

Der Stromkreis (3)

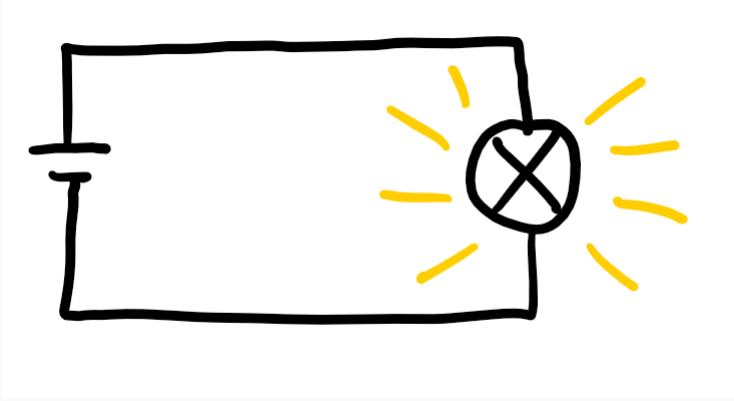
Als Kreis ist es natürlich etwas unpraktisch...



Aber man kann sich - einmal so gezeichnet - den Kreis als Strom-Kreis ganz gut vorstellen.

Der Stromkreis (4)

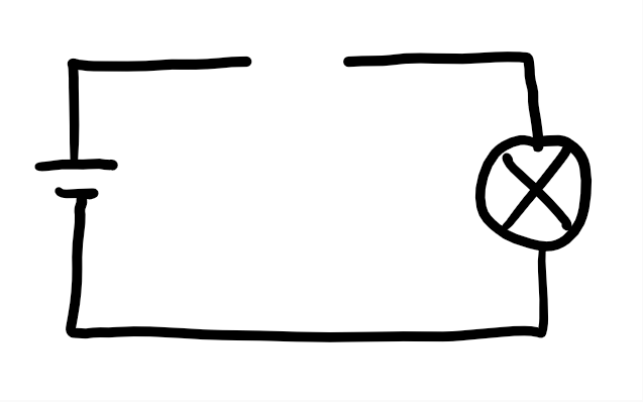
Normalerweise wird das mit geraden Strichen gezeichnet.



Wenn der Stromkreis - so wie hier - geschlossen ist, dann leuchtet die Lampe

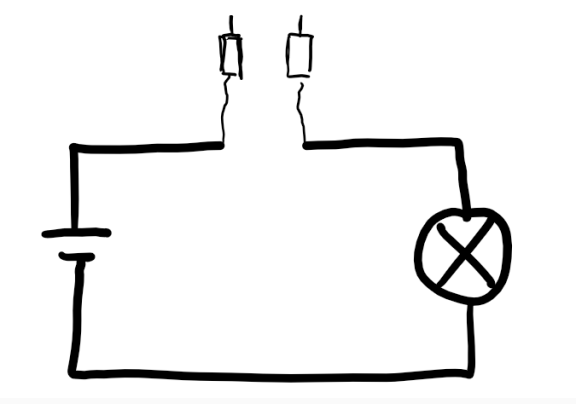
Der Stromkreis (5)

Wenn der Stromkreis - so wie unten - offen ist, dann leuchtet die Lampe nicht. Dies Leuchten der Lampe mit den gelben Strichen sieht man normalerweise in den Schaltbildern nicht!
Aber wir sind ja hier am Lernen dieser Dinge, da darf man das schon reinmalen.



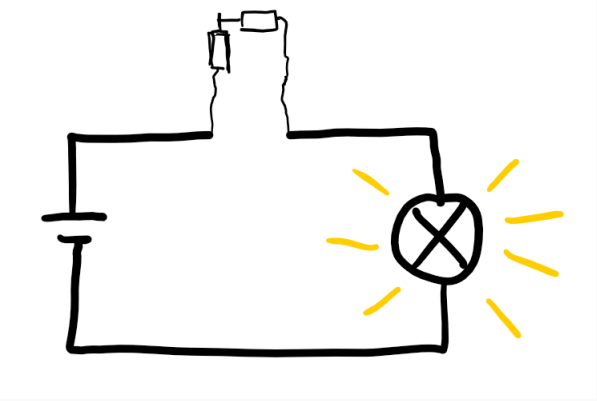
Der Stromkreis (6)

Nun kann man - **aber nur bei den niederen Spannungen mit denen wir arbeiten** - zwei Kabel mit z.B. sogenannten “Bananen-Stecker” an die offene Stelle befestigen



Der Stromkreis (7)

Wenn man nun die beiden Bananenstecker verbindet, leuchtet die Lampe!

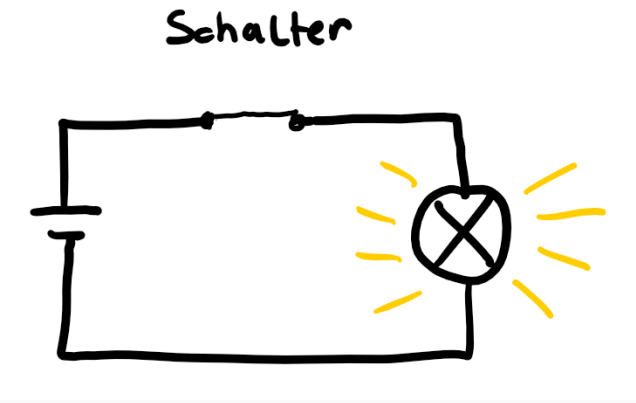


Der Stromkreis (8)

Praktischerweise gibt es dafür auch Bauelemente, um die beiden Kontakte zu verbinden.

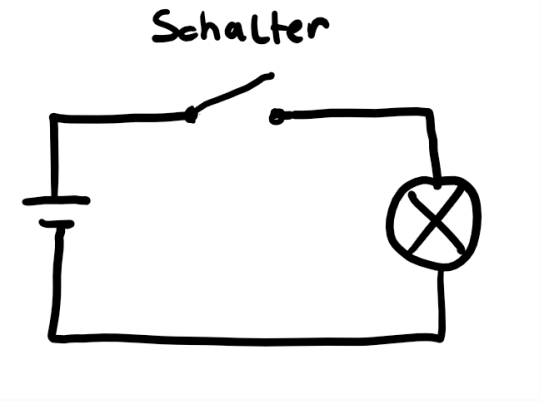
Das ist einfach ein **Schalter**!

Schalter geschlossen : Lampe leuchtet



Der Stromkreis (9)

Schalter offen : Lampe leuchtet nicht



Der Stromkreis (10)

Zusammenfassung :

- Geschlossener Stromkreis: Verbraucher läuft/Lampe leuchtet
- Offener Stromkreis: Verbraucher läuft nicht/Lampe leuchtet nicht
- Dass eine Lampe leuchtet/nicht leuchtet zeigt man normalerweise in einem elektronischen Bild **NICHT** an, aber wir können das schon tun...


Elemente müssen zusammenpassen

Wenn nun also Spannungs-Lieferant und Verbraucher zusammenkommen ist es sehr wichtig, dass diese zusammenpassen.

Zu hohe Spannung (1)

Wir möchten ja weder, dass die zu starke Spannungsquelle unseren Verbraucher zerstört:

(https://pixabay.com/de/niagaraf

(https://pixabay.com/de/wasserrad-holz-bach-modell-778801 ,

CC0 Creative Commons)

Würde man versuchen, das kleine Wasserrad an den Niagara-Fällen zu betreiben, würde das Wasserrad das wohl nicht “überleben”

Zu geringe Spannung (1)

Ebenso möchten wir natürlich, dass sich unser Verbraucher “bewegt”, oder leuchtet oder . . .

(https://pixabay.com/de/wasserhahn-brunnen-wasserspender-1684902 CC0 , Creative Commons)

(https://pixabay.com/de/japan-waterwheel-826639 , CC0 Creative Commons)

Das Wasser aus diesem Mini-“Wasserfall” wird sicher unsere grosses Wasserrad nicht treiben können.

Zu geringe Spannung (2)

Dieser Fall ist allerdings sowohl bei unserem Wasservergleich als auch beim Arbeiten mit Stromkreisen im Allgemeinen der weniger schlimme Fall:

Unser Verbaucher wird wahrscheinlich einfach gar **nicht** oder nicht ganz wie erhofft **arbeiten**.


Zusammenfassung

- Im Stromkreis müssen die Verbraucher und die Spannungs-Quelle die selben Spannungen aufweisen, um sinnvoll zu funktionieren.
- Ist die Spannungsquelle zu gross, wird ziemlich sicher irgendetwas zerstört werden.
- Ist die Spannungsquelle zu klein, wir meist nichts passieren, aber auch nichts “funktionieren”

Lizenz/Copyright-Info

02_06_Elektronik_Action

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Praktisches Arbeiten

Wir nehmen die Batterien

Wir sortieren Batterien der Grösse nach

Wir sortieren Batterien der Spannung nach

Die kleine Batterie ganz rechts hat **12 Volt** !!!
Das ist extrem beeindruckend für alle, die das sehen!

Wir sortieren Verbraucher der Spannung nach

- Verschiedene Lämpchen, LEDs, Motoren, ...
- Der Spannung nach sortieren
- Ist hier als Bild nicht so beeinduckend wie die Batterien, darum ohne Bild

Was passiert wenn Verbraucher und Batterie zusammenpassen

- Verschiedene Batterien mit verschiedenen Verbrauchern.
- Darauf achten, dass die Spannungen zusammenpassen

Was passiert wenn Batterie-Spannung kleiner als Verbraucher

- Wir hängen z.B. eine 5V Glühlämpchen an eine 1.5 V oder eine 3 V Batterie
- Wir hängen eine 5V-LED (z.B. mit eingebauten Vorwiderstand) an 3V CR2032
- Wir hängen einen 5V Motor an eine 1.5 V Batterie

Was passiert, wenn der Verbraucher eine geringere Spannung hat als die Batterie

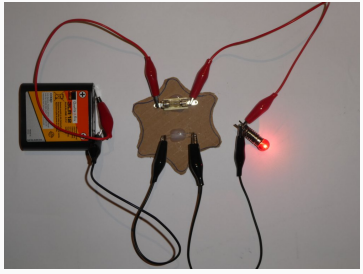
- Das ist der Teil, der am meisten Spass macht...
- Wir lassen ein paar LEDs und ein paar Glühbirnchen "platzen"
- Zuerst mit geringfügig höheren Spannungen der Batterie, dann kann man schon sehen, dass das der Verbraucher nicht lange aushalten wird...
- Dann mit viel höheren Spannungen (z.B. 9V Batterie an normaler 2V-LED)

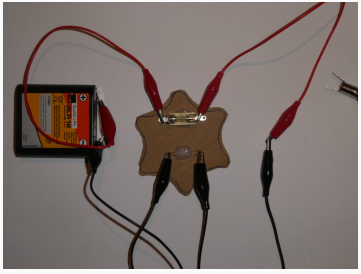

Was passiert bei einem Kurzschluss der Batterie

- Wir schliessen eine kleine Batterie mit einem Kabel kurz.
- **ACHTUNG**: Üblicherweise wird sehr schnell das Kabel warm!
- Kurzschluss kann selbst bei einer kleinen Batterie zu grosser Hitze und zu Flammen führen.
- **NIEMALS** mit einem Akku machen, der brennt **SEHR** schnell!

Was passiert bei einem Kurzschluss mit dem Calliope

- Wir schauen, was bei einer Schmelz-Sicherung passiert, wenn ein Kurzschluss gemacht wird
- Wir stellen uns vor, die Schmelz-Sicherung ist unsere Elektronik (unser Calliope) und wir schliessen an den Calliope einen Verbraucher an und machen dabei einen Kurzschluss...





Nach dem Kurzschluss, bei dem die Sicherung durchbrennt, ist unser "Calliope", dargestellt durch die kleine Schmelz-Sicherung, kaputt!

Fazit

- Strom und Spannung können auch bei den Spannungen mit denen wir arbeiten, zerstörerisch sein!
- In der Hobby-Elektronik sind zur Zeit Spannungen von 3.3 V und Spannungen von 5V üblich
- Bauelemente die nur für 3.3 V ausgelegt sind, gehen mit 5V ziemlich sicher kaputt
- Unser Calliope kann, wenn man an den falschen Stellen Kurzschlüsse macht, kaputtgehen.
- Lieber einmal mehr vorsichtig sein und versuchen etwas über das Bauteil, die Spannung herauszufinden, bevor man Dinge verkabelt, die man nicht versteht.
- Kurzschlüsse können auch bei unseren kleinen Spannungen einiges anrichten!

Lizenz/Copyright-Info

02_07_ExterneLED

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Externe LEDs ansteuern

Nochmals: Spannungen (1)

Der Grund, warum wir uns überhaupt in den letzen Minuten/letzte halbe Stunde mit Elektronik-/Elektrotechnik-Grundlagen beschäftigt haben, ist folgender:

Der Calliope hat schon sehr viele Dinge (die wir auch noch genauer anschauen werden) auf dem Board. Aber manchmal reicht das nicht und man will etwas an den Calliope anschliessen.

Und damit man dabei weder den Calliope, noch das was man anschliesst, zerstört, haben wir ein paar Basis-Dinge gelernt.,

Nochmals: Spannungen (2)

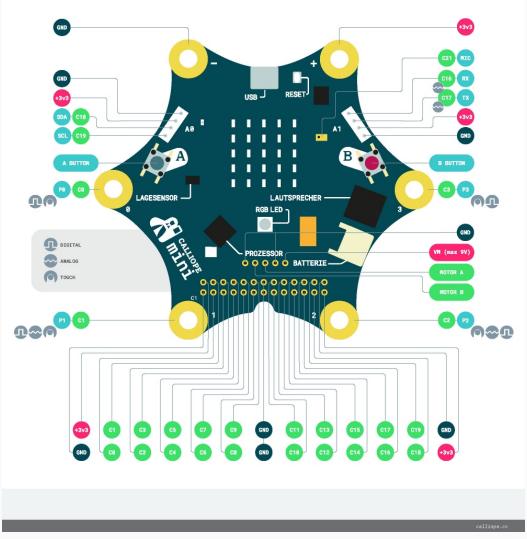
- Wenn Spannung von Lieferant und Verbraucher nicht übereinstimmen, dann funktioniert es nicht.
- Wenn die Spannung vom Lieferanten höher ist als das, was der Verbraucher "verträgt", dann muss damit gerechnet werden, dass der Verbraucher kaputtgeht.
- Wenn die Spannung vom Lieferant kleiner ist, als das was der Verbraucher braucht/verträgt, dann ist i.A. der Schaden klein, es funktioniert einfach nicht!
- Wenn man an den falschen Stellen Kabel zusammenbringt, dann verursacht man einen Kurzschluss.
- Ein Kurzschluss führt dazu, dass viel Strom fliesst, was im guten Fall nur die Batterie erwärmt, im schlechten Fall geht dabei der Calliope kaputt oder die Batterie wird zu warm und fängt Feuer!

Also Vorsicht !

Wir werden nun also die Anschlüsse des Calliope ausprobieren.

Das offizielle Calliope-Layout

So sieht das offizielle Layout des Calliope aus:



(https://calliope-mini.github.io/assets/v10/img/Calliope_mini_1.0_pinout_fin.jpg)

Fähigkeiten der Pins

Wenn man die Anschlüsse auf dem Calliope etwas genauer anschaut, dann sieht man an

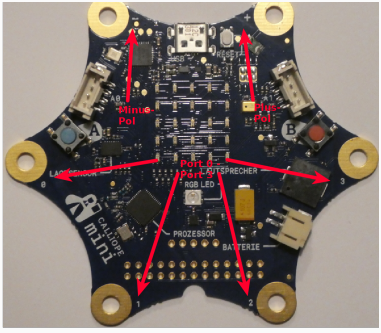
- P0: Digital und Touch
- P1: Digital, Analog und Touch
- P2: Digital, Analog und Touch
- P3: Digital und Touch

Pins als Ausgang oder Eingang

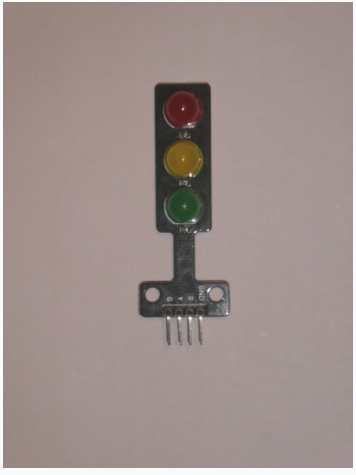
Wichtig ist an dieser Stelle:
Die **Pins** kann man vom Programm aus sowohl als **Ausgang** schalten, d.h. wir können z.B, eine LED ein und ausschalten, als auch kann man die **Pins** als **Eingang** schalten, sprich man kann vom Programm aus abfragen, ob von aussen eine Spannung angelegt wurde, ob der Eingang mit dem Finger berührt wurde und ähnliches.

Externe LED ansteuern

Wir werden nun LEDs an die Anschlüsse anschliessen und schauen, ob wir die LEDs selbst ansteueren können.

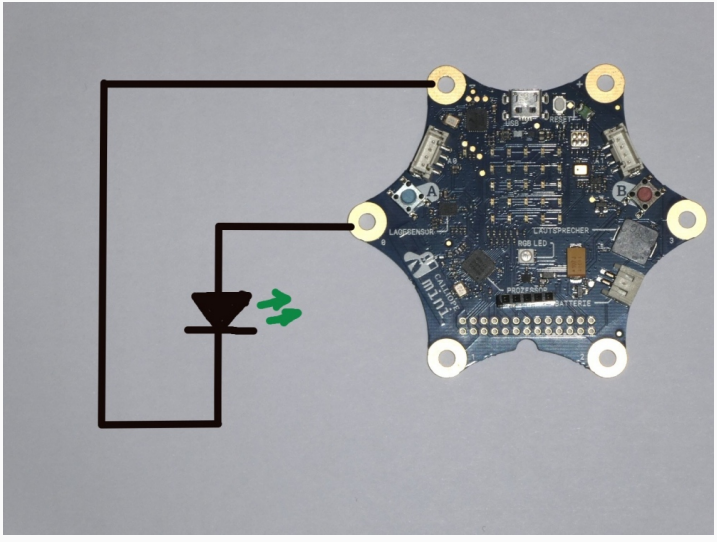


Die Ampel

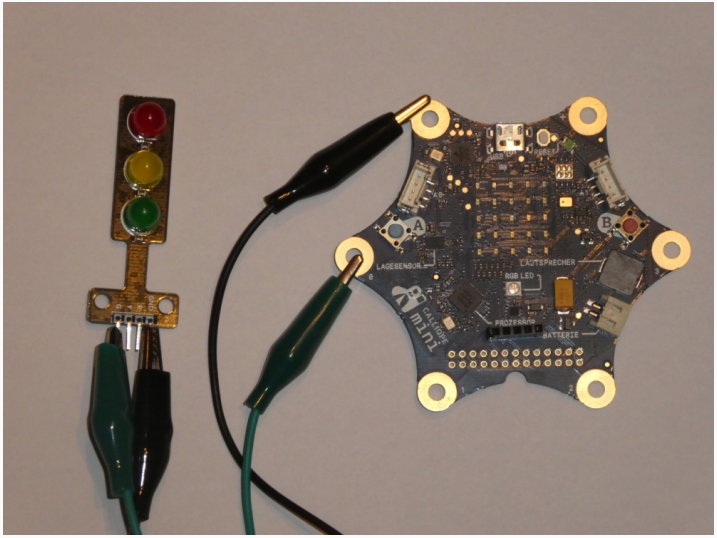


So sieht unsere Ampel aus. Sie hat drei LEDs, die einzeln geschaltet werden können. Sie sind am Minus-Pol (=GND) verbunden

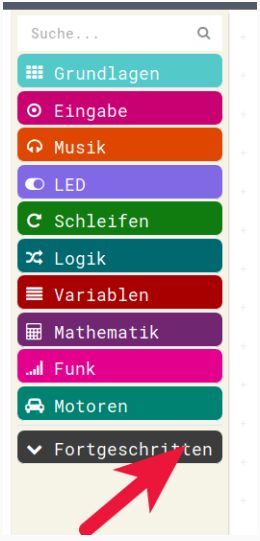
Schaltbild



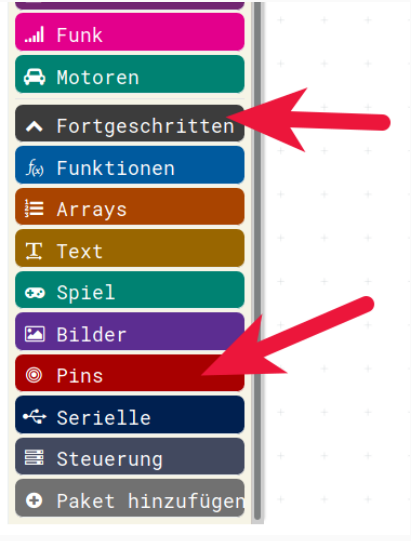
Angeschlossen mit Krokos



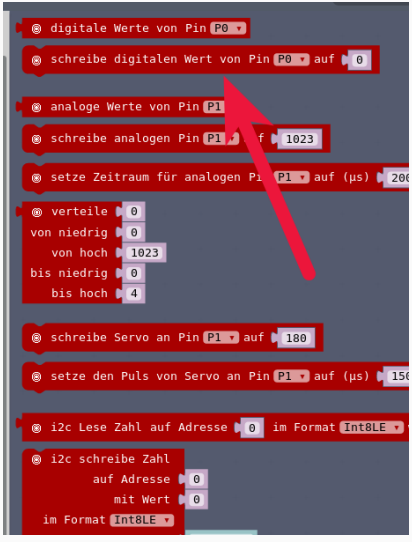
Menu : Fortgeschritten



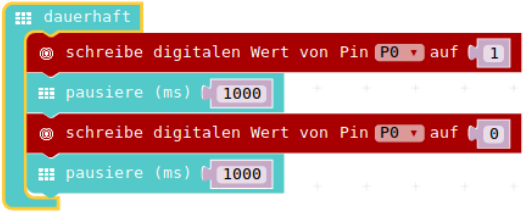
Menu Pins


Funk
Motoren
Fortgeschritten
Funktionen
Arrays
Text
Spiel
Bilder
Pins
Serielle
Steuerung
Paket hinzufügen

Schreibe digitalen Wert



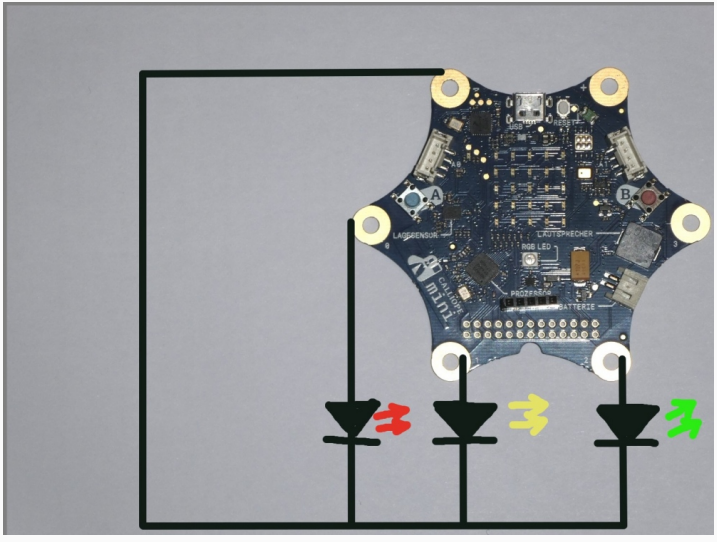
Ein einfacher Blinker



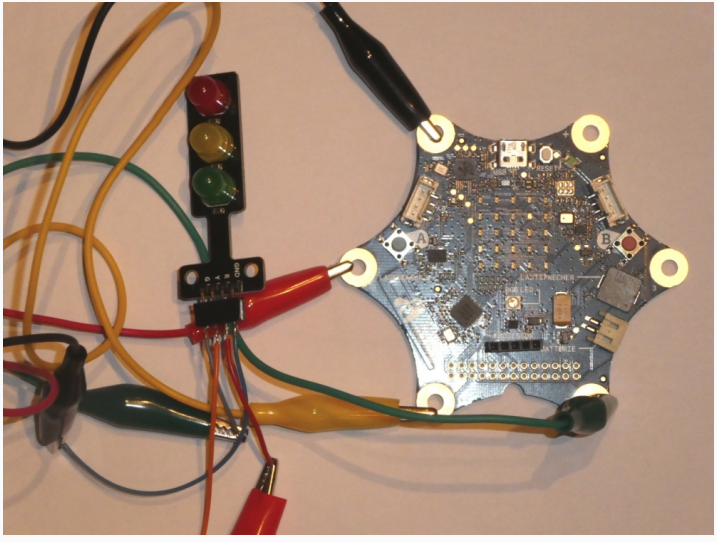
Blinker abspeichern

- Das Programm unter dem Namen : **BlinkenExtLED** abspeichern
- Das Programm in den **Calliope laden**
- Geht's ?
- Auch mal die anderen LEDs testen
- Gleicher Pin am Calliope, anderer an der LED

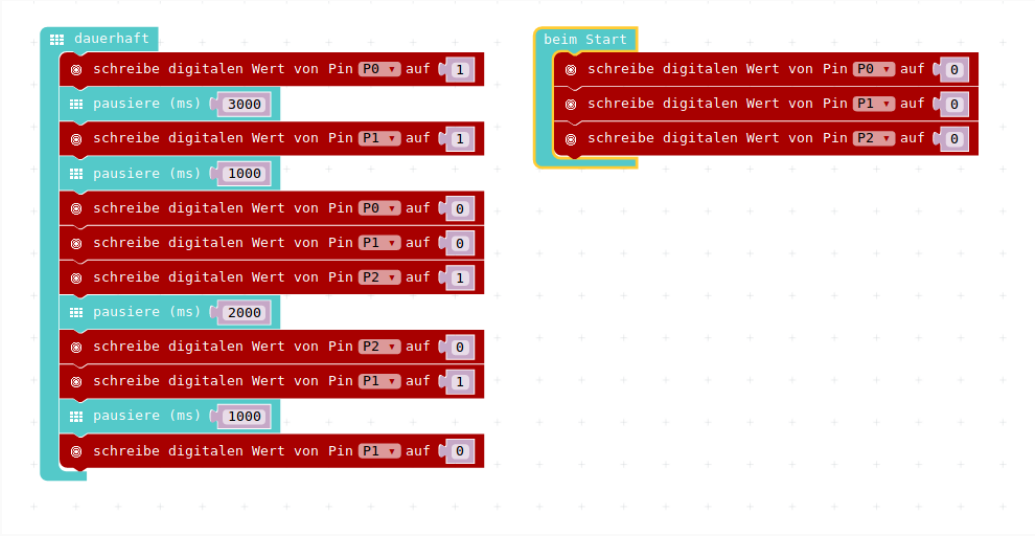
Vollständige Ampel



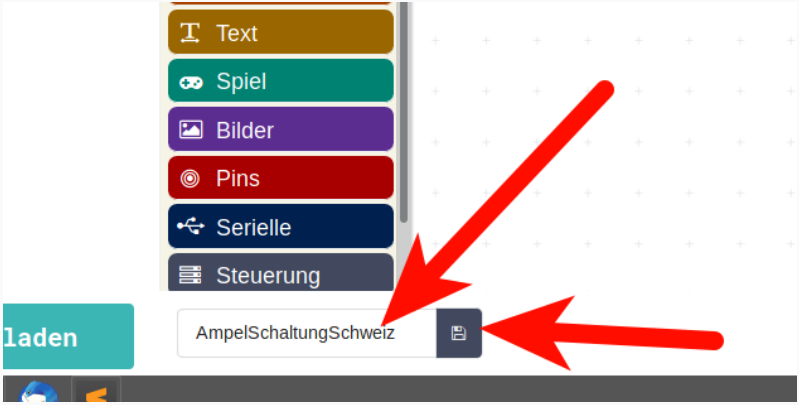
Vollständige Ampel mit Krokos



Einfaches Ampel-Programm



Ampel-Programm speichern



- Speichern unter dem Namen : **AmpelSchaltungSchweiz**
- In den **Calliope laden**

Österreichische Ampel

Wer war schon mal in Österreich?

Da sieht die Ampel-Schaltung ein kleines bisschen anders aus:

https://www.youtube.com/watch?v=dnmarj_TWDc

Österreichische Ampel Programm

Österreichische Ampel speichern

- Speichern unter dem Namen **AmpelSchaltungOesterreich**
- In den **Calliope laden**

Lizenz/Copyright-Info

02_08_PINs

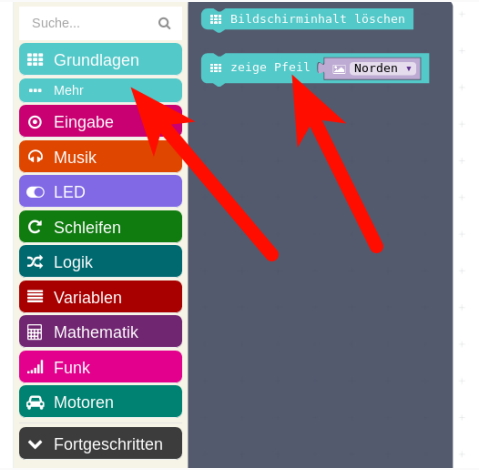
Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

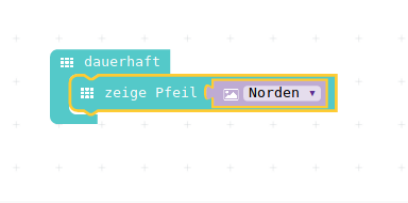
Frühjahr 2019


PINS

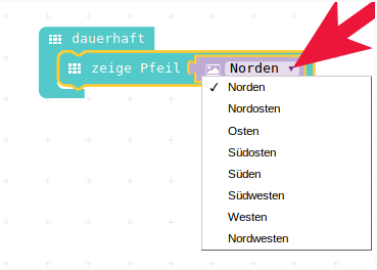
Zeige Pfeil


Suche...
Grundlagen
Mehr
Eingabe
Musik
LED
Schleifen
Logik
Variablen
Mathematik
Funk
Motoren
Fortgeschritten
Bildschirminhalt löschen
zeige Pfeil
Norden

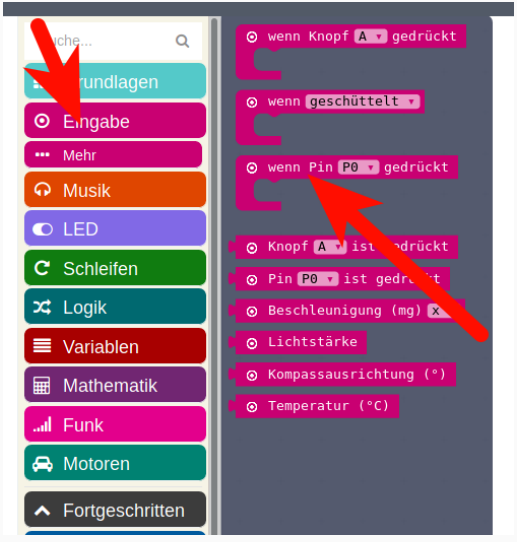
In der “dauerhaft”-Schleife



- Verschiedende Pfeil-Richtungen ausprobieren



wenn Pin gedrückt



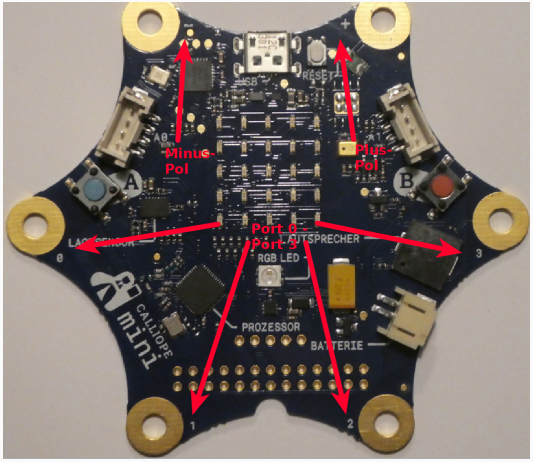
Fertiges Programm mit Pins

Programm speichern

- Programm speichern
- Programm auf den **Calliope laden**

Programm ausführen

Achtung! Ihr müsst den Minus-Pol oben links **und** einen der vier programmierten Pins anfassen!



Lizenz/Copyright-Info

Für alle Bilder auf diesen Seite/Folien, soweit nicht unter dem Bild anders gekennzeichnet, gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

03_01_Zufall

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

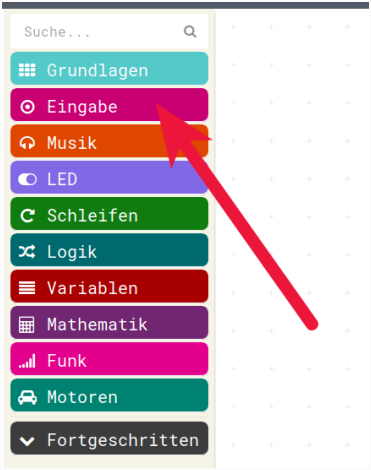
Frühjahr 2019


Zufallszahlen

Wir wollen einen Würfel bauen

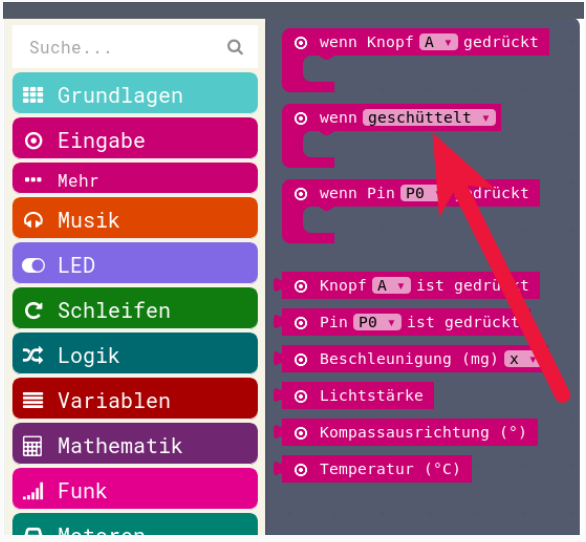
- Beim Schütteln des Calliope
- Eine Zufallszahl erzeugen
- Wie echter Würfel : zwischen 1 und 6
- Die Zahl soll angezeigt werden
- Bei erneutem Schütteln:
- Von Vorne

Menu : Eingabe



Wenn geschüttelt

Wenn geschüttelt :

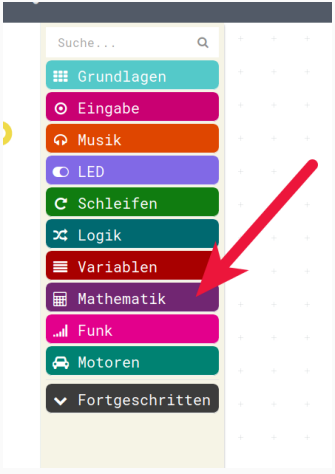


Start des Programms

Auf die **Arbeitsfläche**

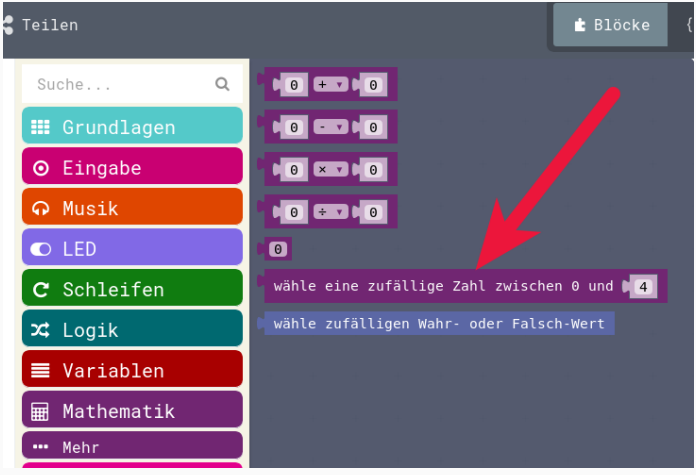
Zufall: Menu Mathematik

Menu Mathematik



Menu Mathematik

Zufalls-Zahlen

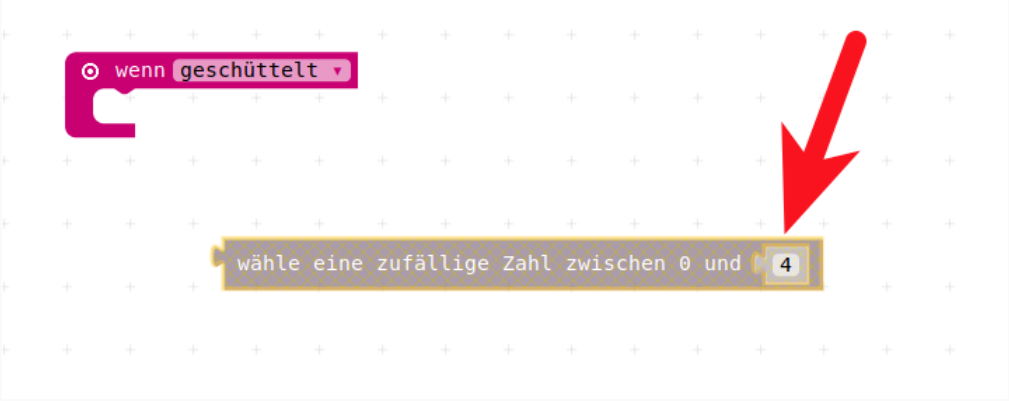


Zufalls-Zahlen ab 0!

- Zufallszahlen beginnen bei 0 !
- Die maximale Zahl ist wählbar

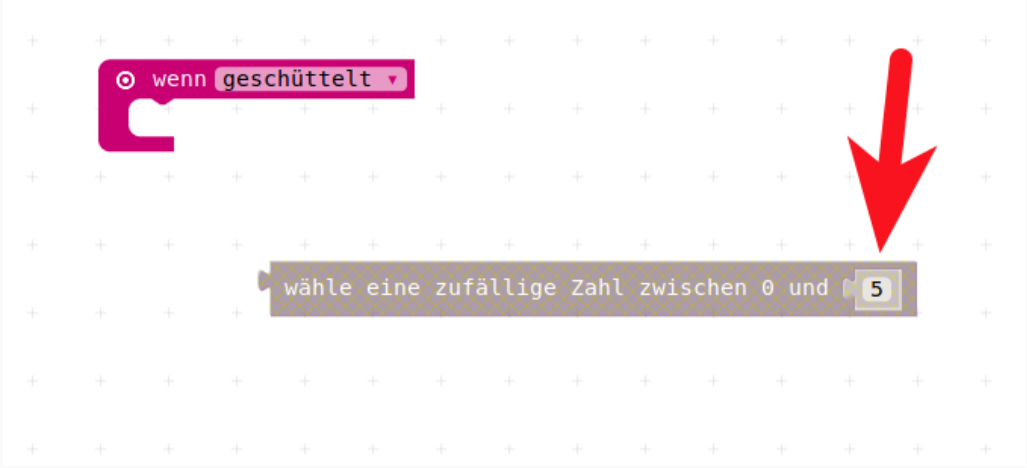
Im Programm

Auf die **Arbeitsfläche** gezogen



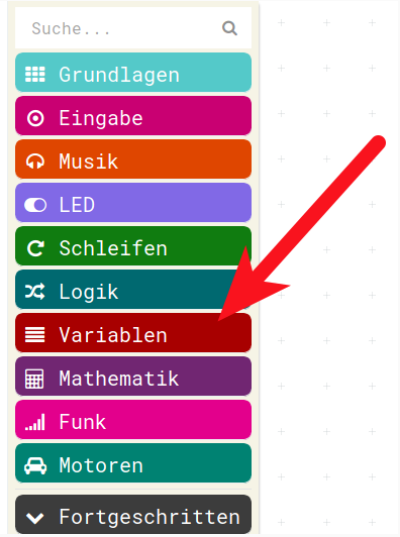
1 - 6 : geht nicht

also 0 - 5 und dann später 1 drauf addieren...



Variable anlegen

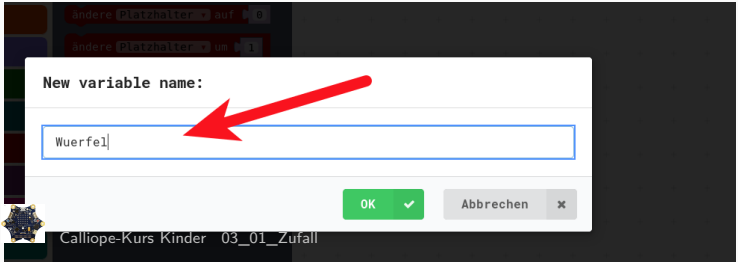
Wir brauchen eine Variable, also legen wir sie an...



Im Menu Variablen

Neue Variable anlegen

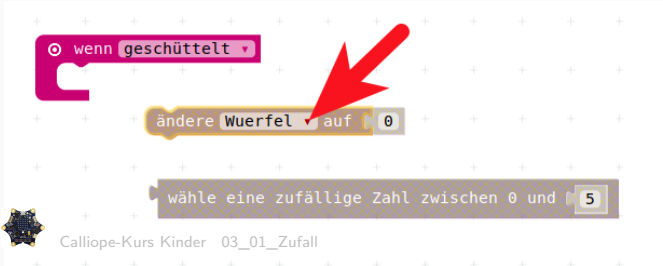
Benennen (zum Beispiel Wuerfel)



Variable auf Arbeitsfläche

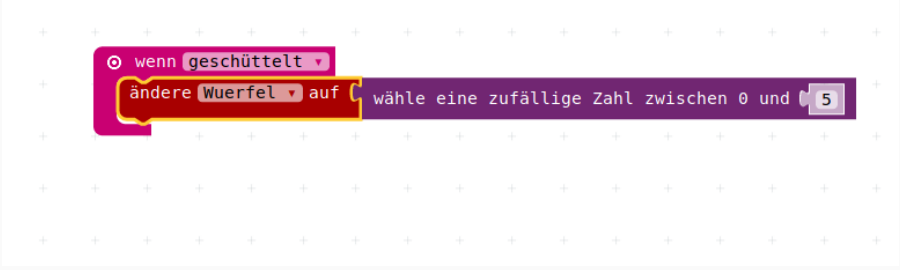
Wir ziehen aus dem Menu Variablen “ändere Platzhalter” auf die Arbeitsfläche

richtige Variable **Wuerfel** benutzen (am Dreieck auswählen)



Zusammenbau 1

Zusammenbau 2



Nun haben wir eine Zufallszahl zwischen 0 und 5 wenn der Calliope geschüttelt wird.

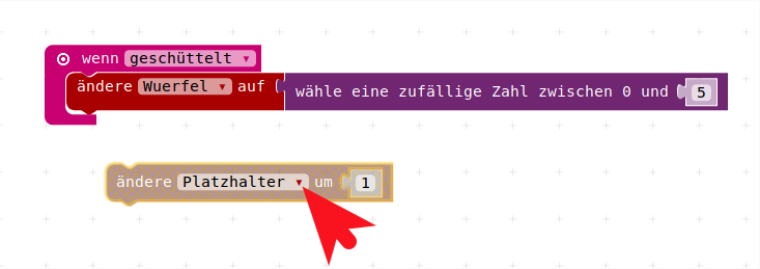
Wert um 1 erhöhen

Aus dem Menu Variablen holen wir uns **ändere Platzhalter um xxx**

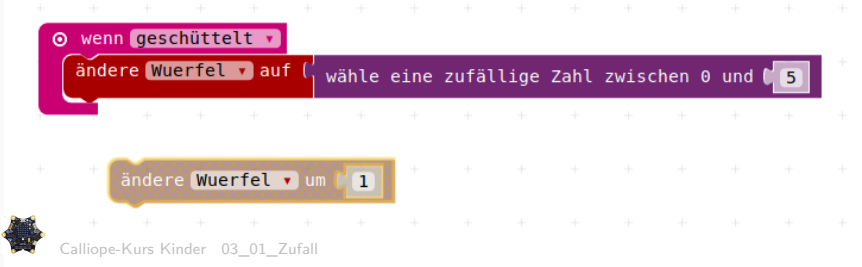
Wichtig: **um** , nicht **auf**

Auf den Arbeitsbereich

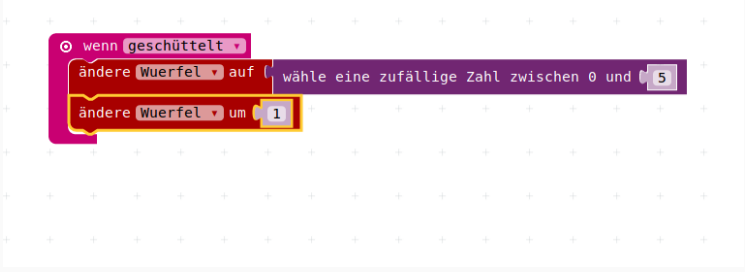
Wir ziehen das auf den Arbeitsbereich:



und ändern die Variable auf **Wuerfel**:

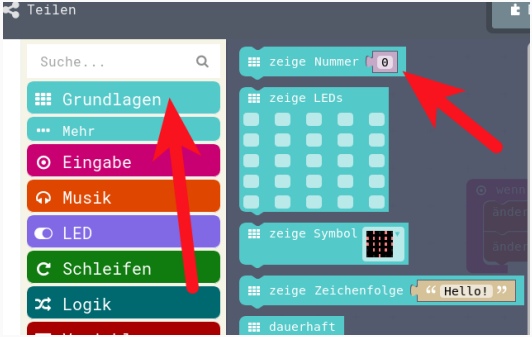


Zusammenbau 3

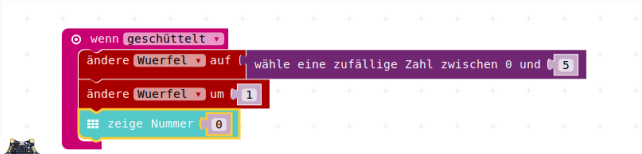


Variable anzeigen

Aus dem Grundlagen-Menu:



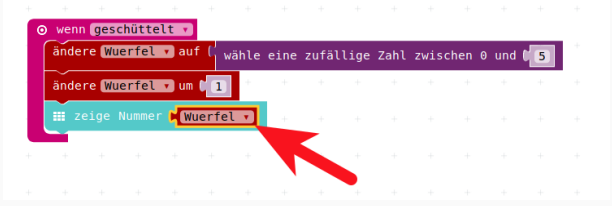
in das Programm



Variable, nicht die 0, anzeigen

Die **Variable** holen

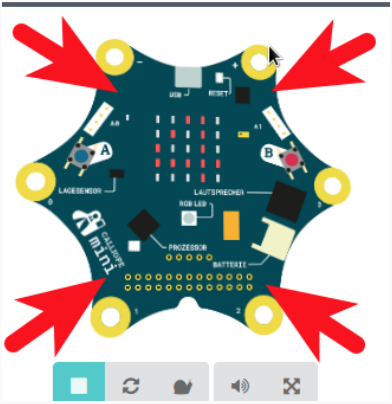
und die 0 damit ersetzen



Im Simulator

Im Simulator kann man diesen Würfel nun ausprobieren

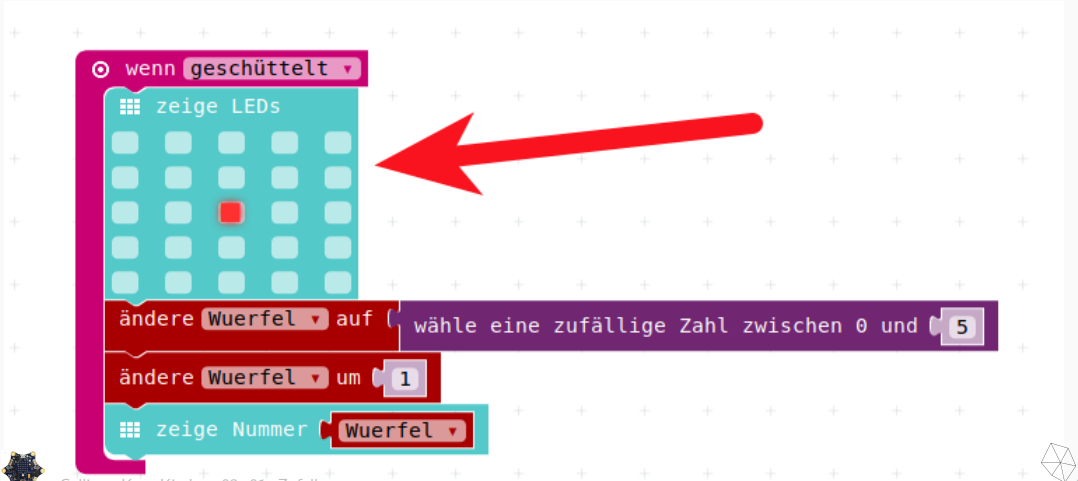
Mit der Maus über die Ecken fahren simuliert Schütteln



Wird gewürfelt?

Vor allem im Simulator sieht man nicht, ob die gleiche Zahl nochmal gewürfelt wurde, oder ob das Schütteln nicht erkannt wurde.

Darum zeigen wir beim Schütteln irgendetwas anderes an, z.B.:



In den Calliope laden

- Spätestens jetzt ist es Zeit, das Programm auch in den Calliope zu laden.
- Das wird hier nun nicht mehr gezeigt...


Lizenz/Copyright-Info


03_02_Die_Aufgabe

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


“Ferien-Hausaufgabe”

Worum gehts

Über die Ferien soll noch eine kleine Auffrischungs-Aufgabe gelöst werden, deren Endprodukt dann auch gleich noch zum Auffrischen für die Schule dienen könnte...


Der “Kleines-Einmal-Eins” - Trainer.

Beim Drücken des linken Knopfes soll Euch eine Multiplikations-Aufgabe aus dem kleinen Einmal-Eins gestellt werden, die Ihr dann selbst im Kopf lösen sollt.
Beim Drücken der rechten Taste zeigt Euch der Calliope das Ergebnis an und Ihr könnt selbst überprüfen, ob Ihr richtig gerechnet habt.
Wenn wieder die linke Taste gedrückt wird, wird eine neue Rechen-Aufgabe gestellt.

Etwas detaillierter: Teil1

- Beim **Drücken** der linken Taste müssen **zwei Platzhalter** mit **Zufallswerten** zwischen 1 und 10 belegt werden.
- Platzhalter sind im Menu Variablen,
- Zufalls-Zahlen sind im Menu Mathematik. Allerdings werden Zufalls-Zahlen ab 0 erzeugt. Ihr müsst also zusätzlich noch jeweils eine 1 addieren...
- Die beiden Zufallszahlen sollen dann **angezeigt** werden, am besten mit einem "*" - Zeichen dazwischen.
- Und vielleicht noch mit " = ? " danach?
- Alle diese **Ausgaben** für Zahlen und Texte findet Ihr im **Menu Grundlagen**.


Etwas detaillierter: Teil2

- Beim **Drücken** der rechten Taste sollen die beiden Platzhalter miteinander **multipliziert (malnehmen)** werden.
- Multiplikationen sind im Menu Mathematik
- Das Ergebnis soll und in einer **dritten Ergebnis-Variable** abgelegt werden.
- Dieses Ergebnis, die Ergebnis-Variable soll dann auch wieder angezeigt werden.
- (Vielleicht noch mit " = " davor. . .)


Viel Spass beim Tüfteln

Viel Spass beim Tüfteln

03_03_Teil1_Hinweise

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Teil 1 : Tipps

Einleitung

Die meisten Tipps stehen schon in der Aufgabe selbst, hier nochmal die Menus, die verwendet werden sollen und ein paar zusätzliche Tips


Taste Drücken

- Beim **Drücken** der linken Taste => **Menu Eingabe**

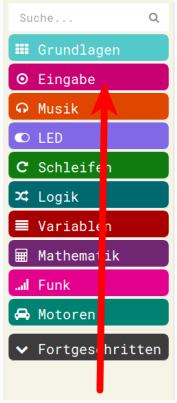


Figure 1: 01_Menu_Eingabe.png

Zwei Platzhalter anlegen

- **zwei Platzhalter => Menu Variablen**

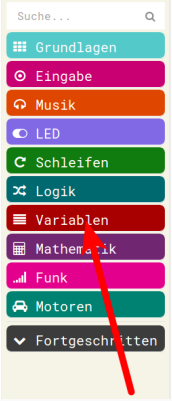


Figure 2: 03_Menu_Variablen.png

Zwei Platzhalter anlegen

- Variablen müssen angelegt werden => **neue Variable anlegen** (sinnvolle Namen vergeben)
- Wir wollen die Variablen **belegen** => nicht die Variable so wie sie da steht benutzen, das ist, wenn man die Variable **lesen** will
  => **ändere Platzhalter auf** aus dem Menu Variablen„ wir wollen **schreiben, belegen, ändern**. . .
- Im Menu Variablen findet man das **nicht direkt** für unsere Variablen, man muss das für **Platzhalter** benutzen und dann per **Drop-Down** (das kleine Dreieck neben dem Namen) den Namen ändern.


Zufallswerte erzeugen

- **Zufallswerte => Menu Mathematik**

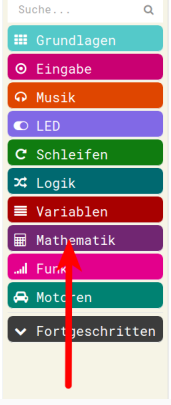


Figure 3: 13_Menu_Mathematik.png

Zufallswerte erzeugen

- erzeugt zwischen 0 ... irgendwas, wir wollen 1 ... 10
- Also erzeugen zwischen 0 ... 9 , dann noch 1 drauf addieren.
- Addieren kann man
  - entweder anschliessend : **ändere Platzhalter um**
  - direkt beim belegen, durch Mathematik, es gibt ein **Additions** Puzzle-Teilchen, das man verwenden kann.


Zahlen und Texte anzeigen

- Zahlen und Texte anzeigen finden sich im Menu **Grundlagen**

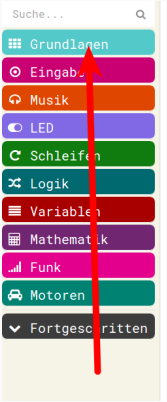


Figure 4: 23_MenuGrundlagen.png

- Zahlen **angezeigen** macht man mit **Zeige Nummer**
- Texte, die angezeigt werden sollen (das “=”-Zeichen, das “+”-Zeichen) werden mit **zeige Zeichenfolge** als Laufschrift angezeigt

Lizenz/Copyright-Info


03_04_Teil1_Loesung

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Teil 1 : “Muster”

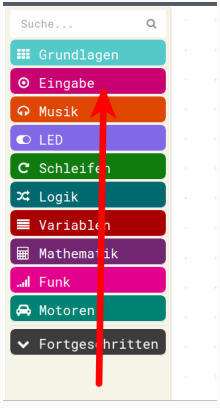
Disclaimer

ACHTUNG: Das “Musterlösung” ist mit Absicht in Hockomma/Gänsefüsschen gesetzt.

- Beim Programmieren gibt es **NICHT** die eine, richtige Lösung.
- Viele verschiedene Lösungen erledigen das selbe, darum ist die **Musterlösung** eine von vielen möglichen!
- Wir werden aber zum Teil sogar unterschiedliche Varianten anschauen, um zu zeigen, dass es verschiedene Möglichkeiten gibt.

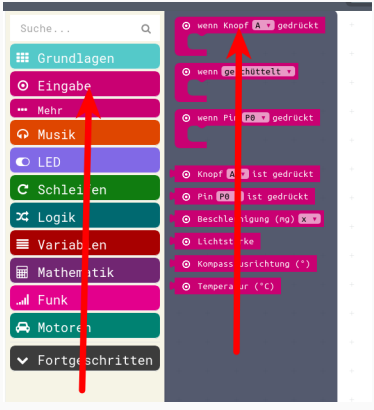

Knopfdruck / Eingabe

- Beim **Drücken** der linken Taste => **Menu Eingabe**



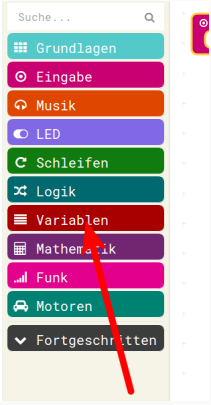
Knopfdruck / Eingabe

- Dort gibt es die Klammer : **Wenn Knopf A gedrückt**
- So wie in unserer Standard-Klammer alle Befehle **immer** ausgeführt werden, wird diese Klammer auch gleich benutzt
- nur werden eben die Befehle beim Drücken der Taste **nur einmal** durchgeführt



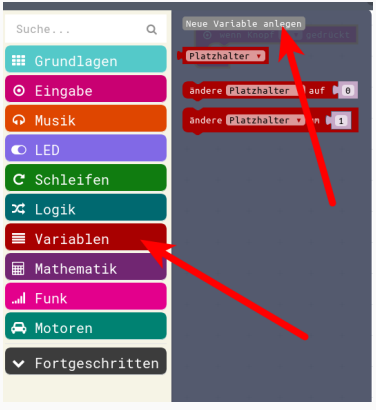
Platzhalter/Variablen

- zwei Platzhalter => Menu Variablen



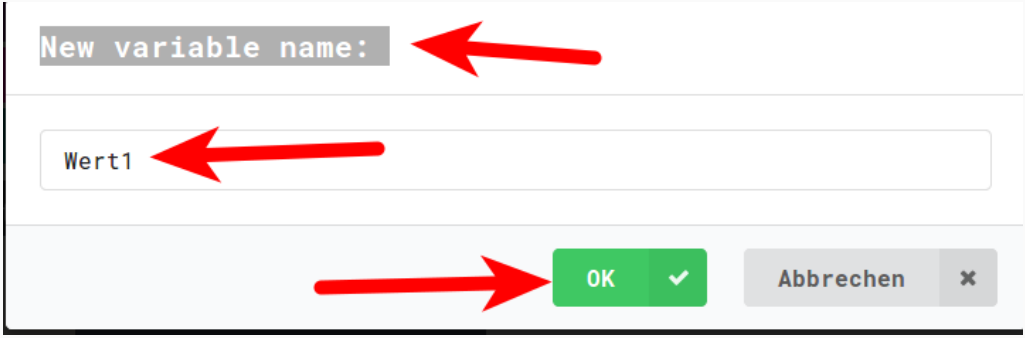
Platzhalter/Variablen

- Variablen müssen angelegt werden => **Neue Variable anlegen** (sinnvolle Namen vergeben)



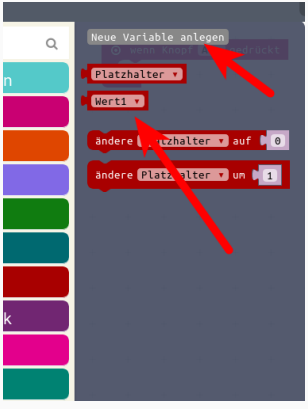
Platzhalter/Variablen

- Das Anlegen einer neuen Variable
- Zuerst “Wert1”



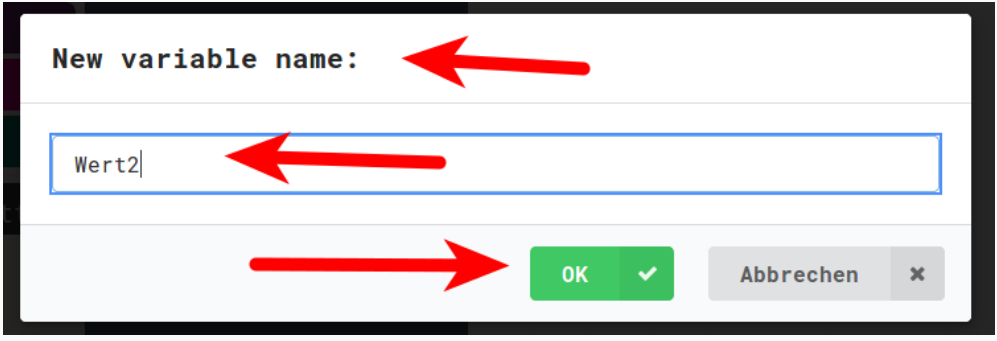
Platzhalter/Variablen

- Wie man sieht ist nun die Variable Wert1 angelegt.
- Nun wird nochmal via **Neue Variable anlegen** eine Variable angelegt



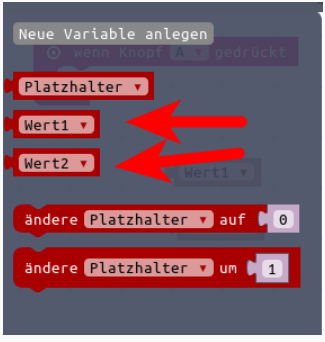
Platzhalter/Variablen

- Wert2 wird angelegt



Platzhalter/Variablen

- Wert2 ist angelegt worden
- Nun haben wir die zwei neuen Variablen : Wert1 und Wert2 im Variablen-Menu zur Verfügung



Platzhalter/Variablen

- **Achtung Falle** :
- Wir wollen zuerst die beiden neuen Variablen mit Zufallswerten “beschreiben”
- Die Puzzleteile sind aber **Lesen** der Variablen!

Platzhalter/Variablen

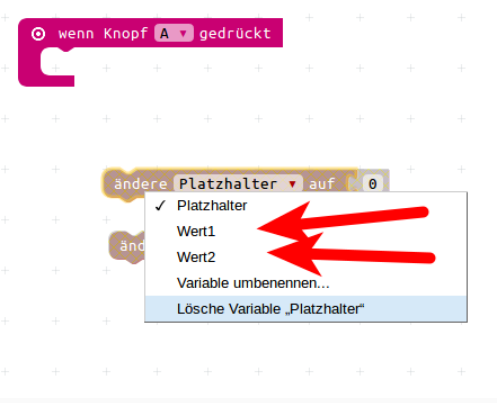
- Darum holen wir uns anstatt dessen das Puzzleteil zum **Schreiben/Setzen** der Variablen aus dem Menu
- Da gibt es allerdings **kein Puzzleteil**, das uns direkt das Schreiben des **Wert1** erlaubt

Platzhalter/Variablen

- Darum müssen wir dieses mit “Platzhalter” nehmen und dann den Platzhalter in unsere Variable **Wert1** abändern
- Das geschieht durch Klick auf das **kleine Dreieck** neben dem Namen **Platzhalter**

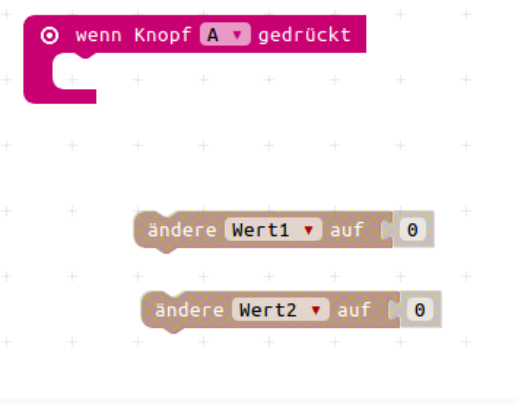
Platzhalter/Variablen

- Dadurch öffnet sich eine sogenanntes Drop-Down-Menu, dieses enthält unsere beiden Variablen Wert1 und Wert2



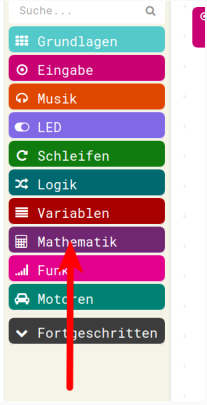
Platzhalter/Variablen

- Wenn diese beiden Änderungen für unsere zwei Variable Wert1 und Wert2 gemacht sind, dann sieht unsere erstes Programm-Fragment so aus.



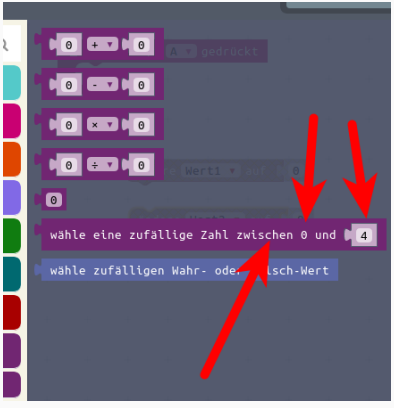
Zufallswerte

- Zufallswerte befinden sich im => **Menu Mathematik**



Zufallswerte

- Dort gibt es einen Zufallsgenerator, der zwischen 0 und irgendwas erzeugt



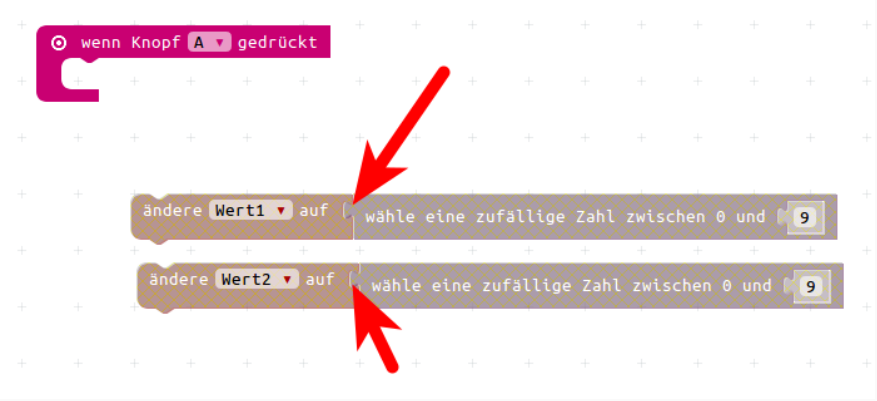
Zufallswerte

- Diesen Befehl holen wir uns jetzt zweimal in die Arbeitsfläche

Zufallswerte

Der Zufallszahlen-Generator erzeugt zwischen 0 ... irgendwas, wir wollen 1 ... 10

- Also erzeugen wir zwischen 0 ... 9 , und werden dann noch 1 drauf addieren.
- So sieht es aus, wenn der Zufall eingeklickt ist und die grösste Zahl noch auf 9 korrigert ist



Werte erhöhen

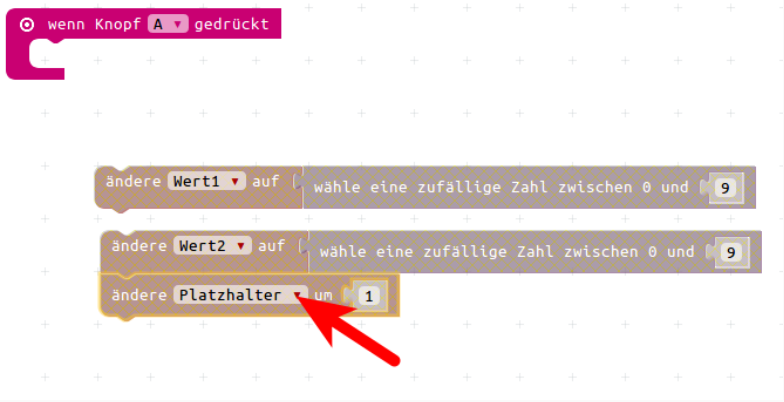
- Um den um eins zu niedrigen Zufallswert zu korrigieren, muss man eine **1 addieren**
- Dazu gibt es 2 Möglichkeiten
  - entweder anschliessend : **ändere Platzhalter um**
  - direkt beim belegen, durch Mathematik, es gibt ein **Additions** Puzzle-Teilchen, das man verwenden kann.
- Wir werden hier beide Möglichkeiten benutzen, um sie zu zeigen.
- Im Normalfall entscheidet man sich für eine Möglichkeit und verwendet diese dann immer...


Werte erhöhen

- **Variante 1:** Variable anschliessend erhöhen via **Ändere Platzhalter um 1**

Werte erhöhen

- Auch hier gibt es diesen Befehl **NICHT** mit Wert1 oder Wert2
- sondern wir verwenden den mit “Platzhalter” und ändern anschliessend wieder via “Drop-Down-Menu” (Kleines Dreieck)

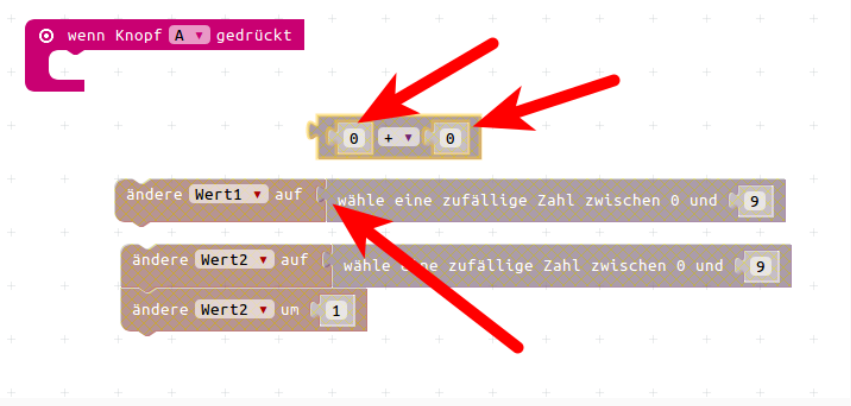


Werte erhöhen

- **Variante 2:** Variable direkt beim Zuweisen durch eine Addition mit 1
- Dazu holen wir eine Addition aus dem Menu Mathematik

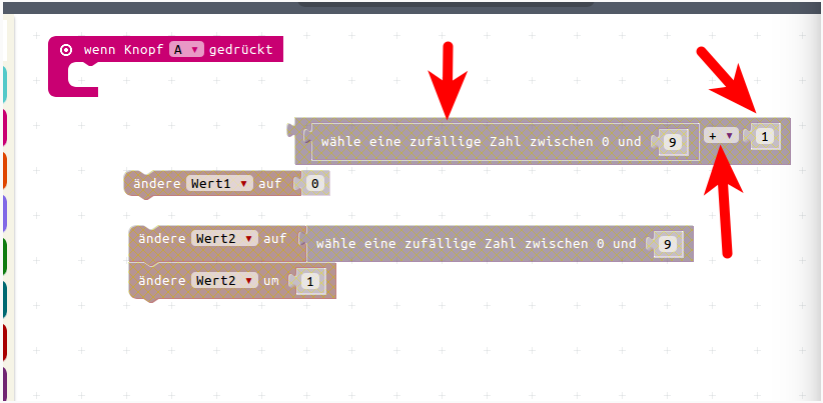
Werte erhöhen

- Wir ziehen den Zufall nach oben als erste Zahl in die Addition rein
- Den anderen Teil der Addition belegen wir mit 1



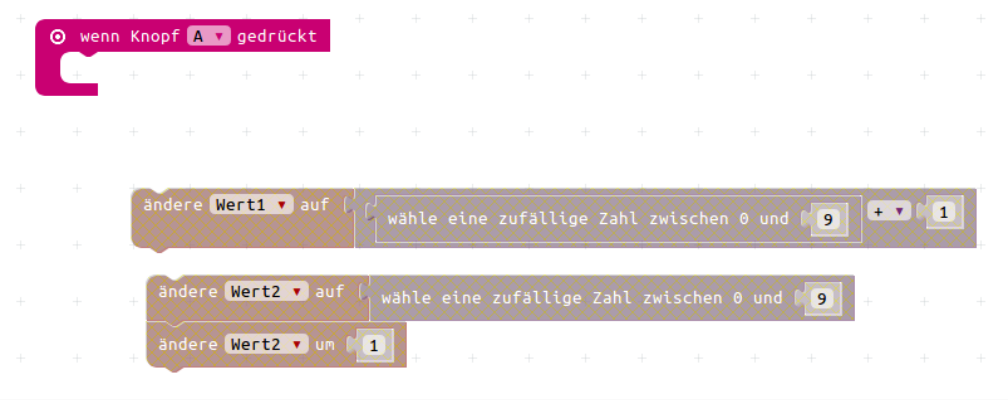
Werte erhöhen

- Anschliessend können wir die gesamte Addition wieder in die Variablen-Zuweisung reinziehen.



Werte erhöhen

- Dann sieht das Gesamtergebnis so aus:



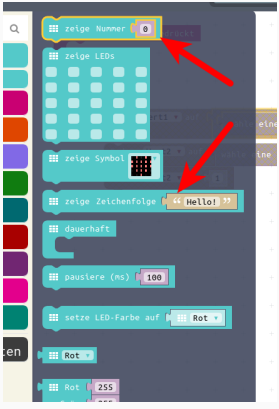
- Wert1 = Zufallszahl zwischen 0...9 und gleich 1 dazu addiert
- Wert2 = Zufallszahl zwischen 0...9
- Wert2 wird erhöht um 1

Zeichen und Zahlen anzeigen

- Zahlen und Texte anzeigen finden sich im Menu **Grundlagen**

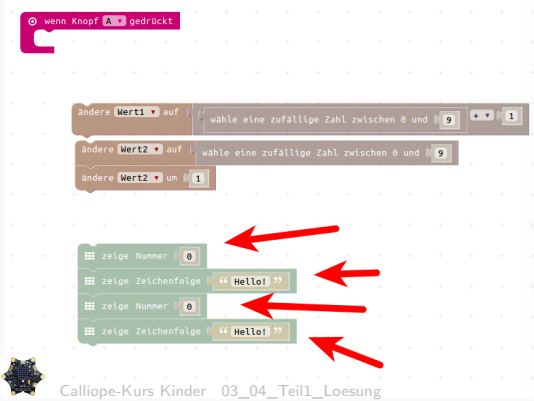
Zeichen und Zahlen anzeigen

- Dort verwenden wir:



Zeichen und Zahlen anzeigen

- Zahlen **angezeigen** macht man mit **Zeige Nummer**
- Texte, die angezeigt werden sollen (das "="-Zeichen, das "+"-Zeichen) werden mit **zeige Zeichenfolge** als Laufschrift angezeigt
- Also holen wir uns zweimal "**zeige Nummer**" und zweimal "**zeige Zeichenfolge**" in den Arbeitsbereich.

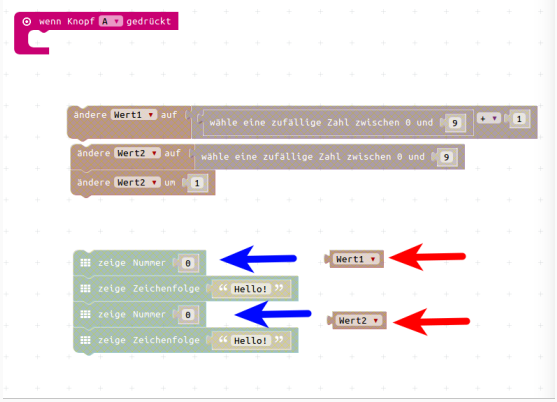


Zeichen und Zahlen anzeigen

- Nun wollen wir - im Vergleich zu vorher - die Variablen lesen um sie anzuzeigen.
- Nun können wir also aus dem “Variablen” - Menu die beiden Variablen “**Wert1**” und “**Wert2**” auf den Arbeitsplatz ziehen.

Zeichen und Zahlen anzeigen

- Wenn wir sie auf dem Arbeitsplatz liegen haben, können wir sie jeweils in die Befehle "**zeige Nummer**" reinziehen, dort ersetzen sie jeweils die vorbelegte "0"

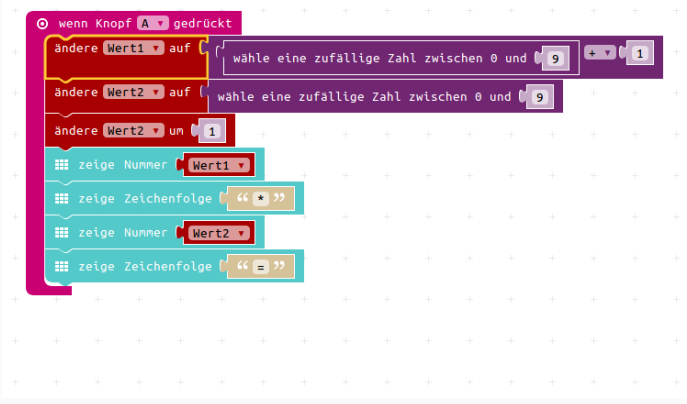


Zeichen und Zahlen anzeigen

- Wenn wir nun noch die beiden Texte "*" und “=” in die Zeichenfolge reinschreiben, dann sind wir eigentlich auch schon fertig.

Zeichen und Zahlen anzeigen

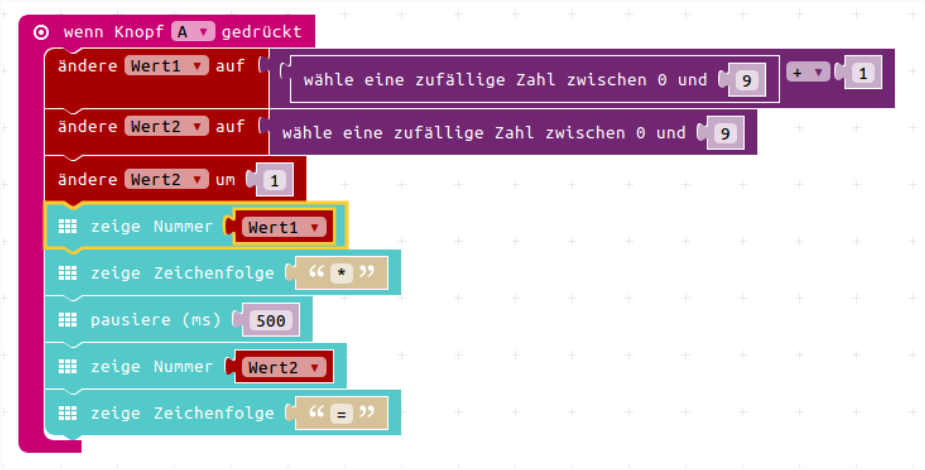
- Wir ziehen alles so in unsere Arbeits-Klammer rein und schauen uns im Simulator an, was passiert.



Zeichen und Zahlen anzeigen

- Wenn man das im Simulator laufen lässt und mittels Klick auf die Taste startet, sieht es ganz gut aus, nur das Multiplikations-Zeichen "*" sieht man nicht.
- die unterschiedlichen LED-Ausgabe-Funktionen sind unterschiedlich schnell, darum muss man nach dem **zeige Zeichenfolge** noch eine Pause einbauen
- So sieht dann das fertige Programm Teil 1 aus:

Muster-Lösung Teil 1



Muster-Lösung Teil 1

- Das können wir nun im Simulator ausprobieren und auch in den Calliope laden und uns daran erfreuen.
- Leider ist es nur der erste Teil, ob wir richtig gerechnet haben, können wir damit noch nicht überprüfen
- aber wir können uns schonmal Zufalls-Zahlen als Rechenaufgabe geben lassen.

Lizenz/Copyright-Info


03_05_Teil2_Hinweise

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Teil 2 : Hinweise

Teil2, die Aufgabe

- Beim **Drücken** der rechten Taste sollen die beiden Platzhalter miteinander **multipliziert (malnehmen)** werden.
- Multiplikationen sind im Menu Mathematik
- Das Ergebnis soll und in einer **dritten Ergebnis-Variable** abgelegt werden.
- Dieses Ergebnis, die Ergebnis-Variable soll dann auch wieder angezeigt werden.
- (Vielleicht noch mit " = " davor. . .)


Teil2, die Hinweise

- Das meiste, was hier gemacht werden muss, wurde gerade auch schon in Teil 1 gemacht, darum hier nur zwei Hinweise:
- Abfragen auf Knopf B gibt es nicht direkt im Menu, das muss man sich aus dem Menu mit Knopf A holen und dann per Drop-Down-Menu (das kleine Dreieck) umstellen


Abfragen im Menu Eingabe

- Abfragen im Menu Eingabe

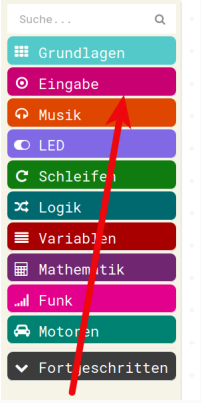


Figure 1: Menu Eingabe

Abfragen im Menu Eingabe

- Kein Knopf B, also nehmen wir “Wenn Knopf A gedrückt” auf die Arbeitsfläche

Figure 2: Kein Knopf B

Abfragen im Menu Eingabe

- Dort gibt es auch wieder die Möglichkeit mit dem kleinen Dreieck ein Drop-Down-Menu zu öffnen
- Wir nehmen das **B** aus dem Menu

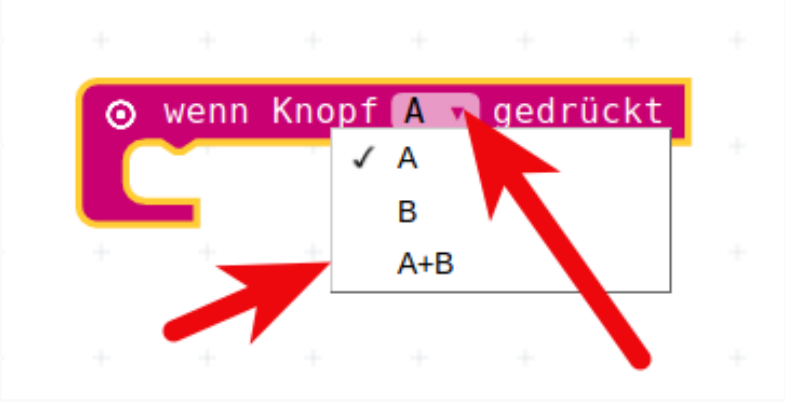


Figure 3: DropDown

Abfragen im Menu Eingabe

- Und haben unser benötigtes “Wenn Knopf B gedrückt”

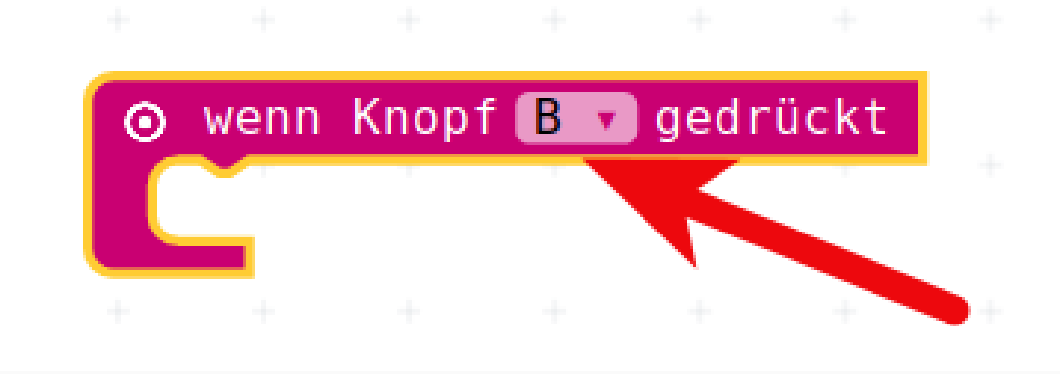


Figure 4: Knopf B

Letzter Hinweis

- Und noch ein Hinweis zum Malnehmen:
- Auch das befindet sich im **Menu Mathematik**

Lizenz/Copyright-Info


03_06_Teil2_Loesung

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Teil 2 : “Musterlösung”

Kommt noch ...

Kommt noch ...


04_01_Auffrischen_Variablen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Wiederholung / Auffrischen
Variablen

Platzhalter / Variablen

Da Variablen sozusagen der Grundpfeiler beim Programmieren sind, schauen wir uns in dieser Rückblende **nochmal** genau die Verwendung von Variablen und Ihren Einsatz in einfachen mathematischen Aufgaben an:

Platzhalter / Variablen dienen - wie der Name andeutet - dazu, veränderliche Werte aufzunehmen. Man will aber den Variablen nicht nur Werte “reingeben”, sondern man will dann auch die Variablen wieder “fragen” : Was für einen Wert hast Du denn? In der Programmierung sagt man:

- der Variable wird ein Wert **zugewiesen**
- der Wert der Variablen wird **abgefragt**


Zuweisung an eine Variable

Vorstellen kann man sich das auch als Zuruf mit dem Megaphon:
Hey Du, **Variable_A**, merk Dir doch bitte mal die **2**!

Zuweisung an eine Variable

Oder man stellt sich die Variable als Obstkorb vor, der mit Äpfeln “belegt” wird:

Korb_A :

Zuweisung an eine Variable

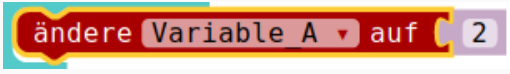
Belege den Korb_A mit 2:

Zuweisung an eine Variable

In einer “normalen” Programmiersprache : **Variable_A** $= 2$
Zuweisung an **Variable_A**, die Variable steht auf der linken Seite des Gleichheitszeichens, der Wert mit dem die Variable belegt werden soll, steht auf der rechten Seite des Gleichheitszeichens.

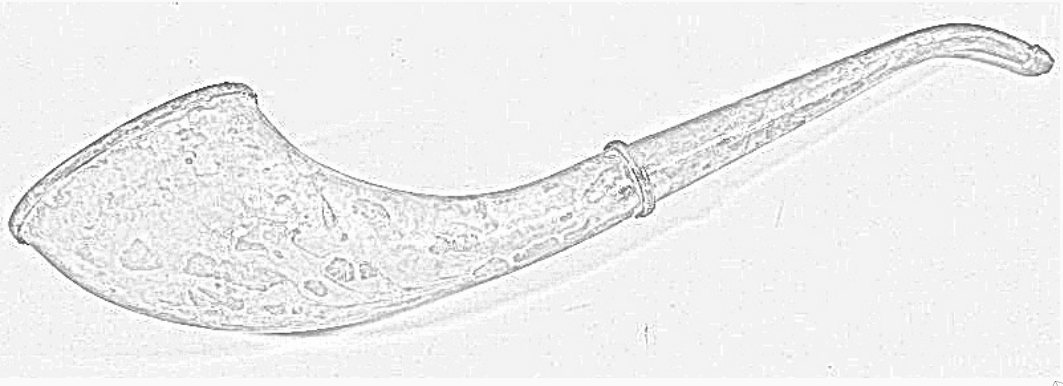
Zuweisung an eine Variable

Die Zuweisung an eine Variable in unsere Calliope-Programmier-Sprache sieht so aus:



Abfrage/Benutzen einer Variable

So wie man sich vorher die Zuweisung mit einem Megaphon vorstellen kann, so kann man sich das Auslesen/Benutzen der Variable vorstellen, dass man mit einem altmodischen Hör-Rohr fragt:
Hey Du, **Variable_A**, was war es, was Du Dir vorher merken solltest? Sag es mir doch bitte.



Abfrage/Benutzen einer Variable

Oder man stellt sich vor, wie man in den Obstkorb reinschaut, wieviele Äpfel denn dort drin sind, man **fragt** die **Belegung ab**:

Was ist der Inhalt von Korb_A :

Abfrage/Benutzen einer Variable

So wie in unserem Korb-Beispiel der Korb nur angeschaut wird, ist das auch beim Programmieren:

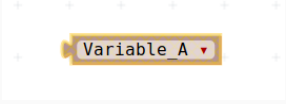
- Der Inhalt des Korbes ändert sich durchs Anschauen nicht
- Der Inhalt der Variable ändert sich durchs Auslesen, durch die Benutzung in einer Formel **nicht**.

In einer “normalen” Programmiersprache wird das üblicherweise z.B. so dargestellt:

- Variable_B = **Variable_A**
  - Der Wert der Variable: **Variable_A** wird erfragt, sie steht auf der rechten Seite des Gleichheitszeichens.


Abfrage/Benutzen einer Variable

In unserer grafischen Calliope-Programmiersprache wird das Benutzen der Variable einfach durch das Puzzleteil mit der Variable dargestellt:



Abfrage/Benutzen einer Variable

Dieses einzelne Puzzleteil kann man dann irgendwo anstatt festen Werten einklicken, also wenn man z.B. die Rechnung mit der **Variable_B** von oben nochmals benutzen will:

- **Variable_B** = **Variable_A**
  - Der Wert der Variable: **Variable_A** wird erfragt, sie steht auf der rechten Seite des Gleichheitszeichens.
  - Auf der linken Seite des Gleichheits-Zeichens wird das der Variable **Variable_B** zugewiesen,
  - die **Variable_B** wird mit dem selben Wert belegt.
- 

Ein paar Rechnungen

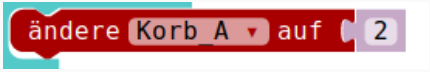
Nun haben wir also mehrere Darstellungs-Möglichkeiten für Variablen-Belegung und Variablen-Abfrage gesehen:

- “Körbchen”-Darstellung
- Darstellung mit Text
- Grafische Calliope-Darstellung

Damit wollen wir noch ein paar ganz einfache Belegungen, Abfragen und Rechnungen zeigen.

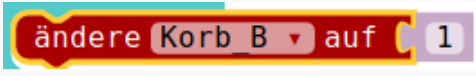

Korb A mit 2 belegen

Korb_A = 2



Korb B mit 1 belegen

Korb_B = 1



Korb C = Summe Korb A + B

Korb_C = Korb_A + Korb_B

Korb C = Summe Korb A + B

=

+

Korb B erhöhen um 1

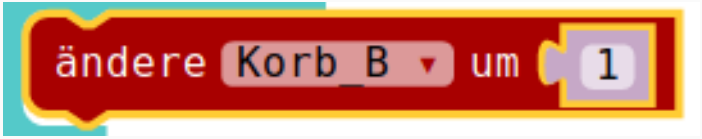
=

+

Korb B erhöhen um 1

Korb_B = Korb_B + 1

oder andere Möglichkeit, die auch leider einfach mit dem Setzen einer Variable zu verwechseln ist:



Was enthält nun Korb C?

Vorher haben wir den Korb C mit der Summe von Korb A und Korb B belegt haben.
Das Ergebnis war 3. Nun haben wir den Korb B um eins erhöht.
Ist nun der Wert des Korb C auch um 1 erhöht worden?
NEIN!
Die Berechnung wurde vorher ausgeführt.
Das Ergebnis ändert sich nicht mehr.
Der Korb C enthält immer noch die 3.

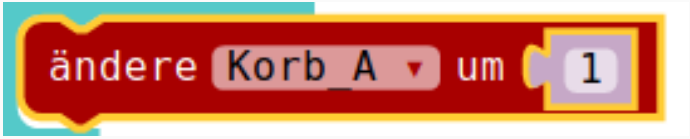
Wir schauen nochmal nach :-)

Korb A erhöhen um 1

Ebenso können wir nun natürlich Korb A um eins erhöhen

Korb_A = Korb_A + 1

oder andere Möglichkeit, die auch leider einfach mit dem Setzen einer Variable zu verwechseln ist:



Korb A erhöhen um 1

=

+

Korb C = Summe Korb A + B

Nun können wir nochmal die Berechnung von vorher durchführen: Korb C soll die Summe von Korb A und von Korb B enthalten.

In “normaler” Programmiersprache:

Korb_C = Korb_A + Korb_B

- Korb A enthält 3 Äpfel
- Korb B enthält 2 Äpfel
- Korb C enthält vor der Berechnung schon 3 Äpfel
- Wenn wir nun die Berechnung durchführen, was enthält denn dann Korb C?
  - **5 Äpfel** = 3 Äpfel von Korb A und 2 Äpfel von Korb B ? Oder etwa:
  - **8 Äpfel** = 3 Äpfel von Korb A und 2 Äpfel von Korb B und die 3 eigenen Äpfel von vorher?
- ?
- ?
- ?


Antwort : Korb C = Summe Korb A + B

Lösung: So wie wir die Berechnung formuliert haben:

Korb C = Korb A + Korb B

ist der vorherige Inhalt von Korb C egal! Er wird quasi vorher ausgeleert!

Also sieht unsere Korb-Rechnung nun so aus:


Antwort : Korb C = Summe Korb A + B

=

+

Antwort : Korb C = Summe Korb A + B

Und in Calliope-Rechnung sieht das ganz wieder genau gleich wie vorher aus:

Lizenz/Copyright-Info

Für alle Bilder auf dieser Seite gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

04_02_Auffrischen_Ampel

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Wiederholung / Auffrischen Ampel

Spannung / Verbraucher 1

Hier nur die wichtigsten Dinge bezüglich Spannung / Verbraucher vom letzten Nachmittag:

- Die Steckdose ist **tabu** !
- Unsere Spannungs-Bereiche liegen zwischen 1.5 Volt und ca 12 Volt, die Steckdose hat **220 Volt**
- Der Calliope arbeitet normalerweise mit 3.3 Volt
- Die Grösse einer Batterie sagt **NICHTS** über ihre Spannung aus!

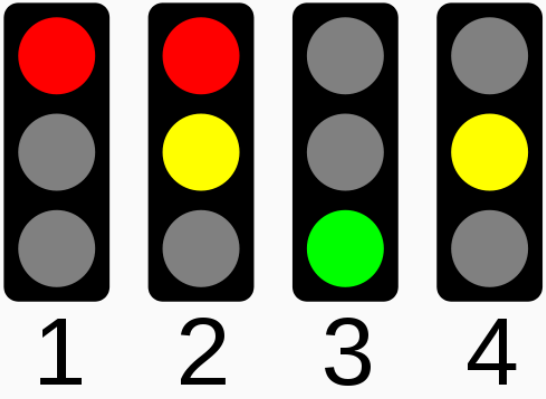
Spannung / Verbraucher 2

- Spannung von Lieferant (Batterie) und Verbraucher (z.B. LED etc) müssen übereinstimmen
- Bei Nichtübereinstimmung kann etwas kaputt gehen, wenn es dumm läuft ist das der Calliope
- Kurzschluss heisst der Pluspol und der Minus-Pol eines Spannungs-Lieferanten werden zusammengehalten/“kurzgeschlossen”
- Beim Kurzschluss geht üblicherweise etwas kaputt, wenn es dumm läuft, ist das der Calliope.

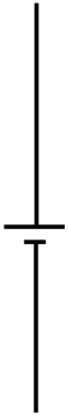

Ampel

- Die Ampel und die Ansteuerung der Pins ging letztes Mal vielleicht ein kleines bisschen (zu?) schnell.
- Darum in diesem Auffrischen das Ganze nochmal Schritt für Schritt

Wir möchten gerne eine Ampel haben, die folgendes ermöglicht:

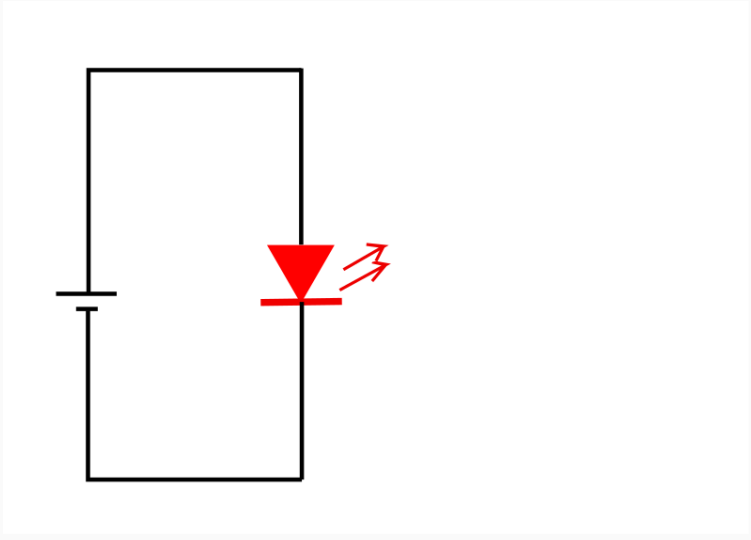


Ohne Calliope 1



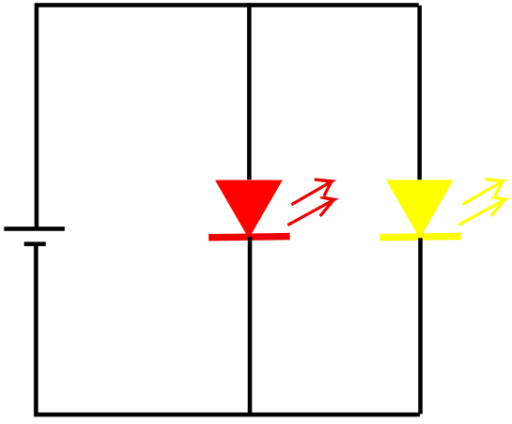
Wir brauchen eine Batterie

Ohne Calliope 2



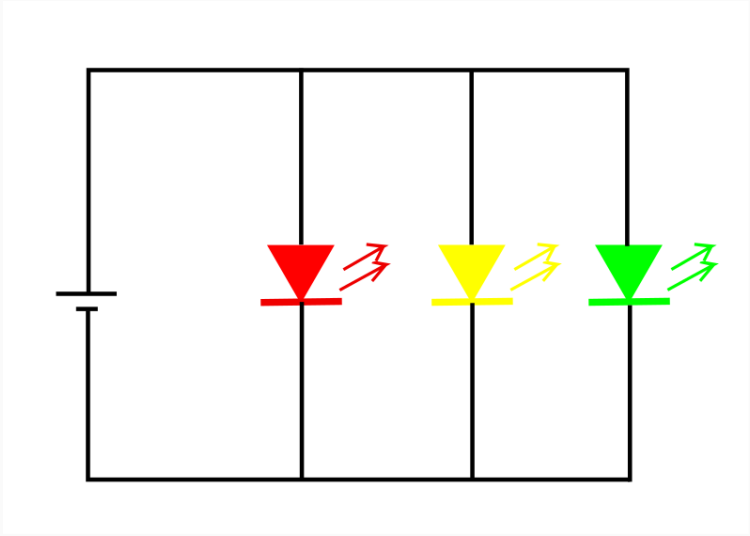
Wir brauchen eine rote LED

Ohne Calliope 3



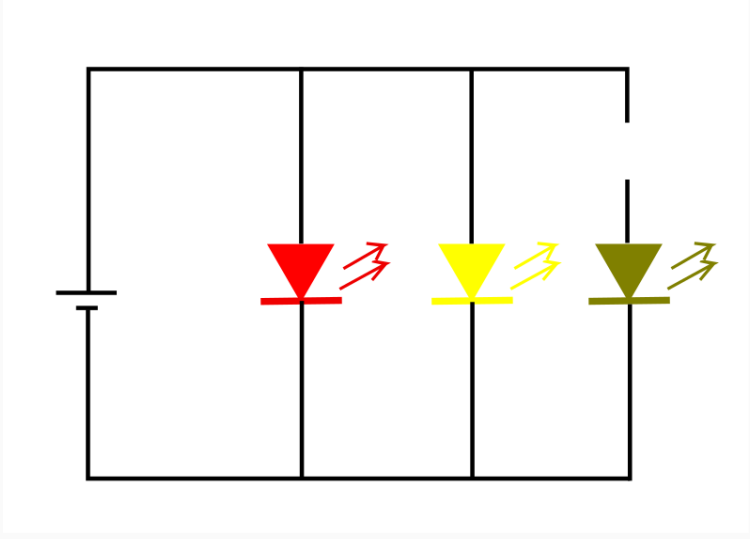
Wir brauchen eine gelbe LED

Ohne Calliope 4



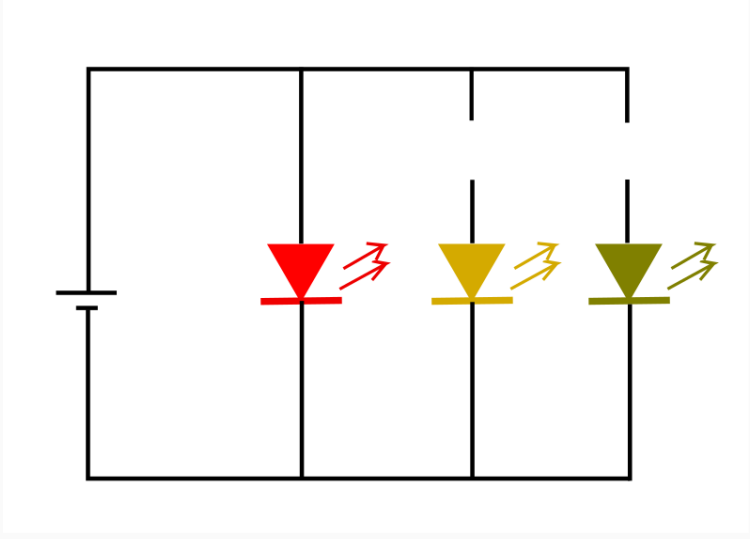
Wir brauchen eine grüne LED

Ohne Calliope 5



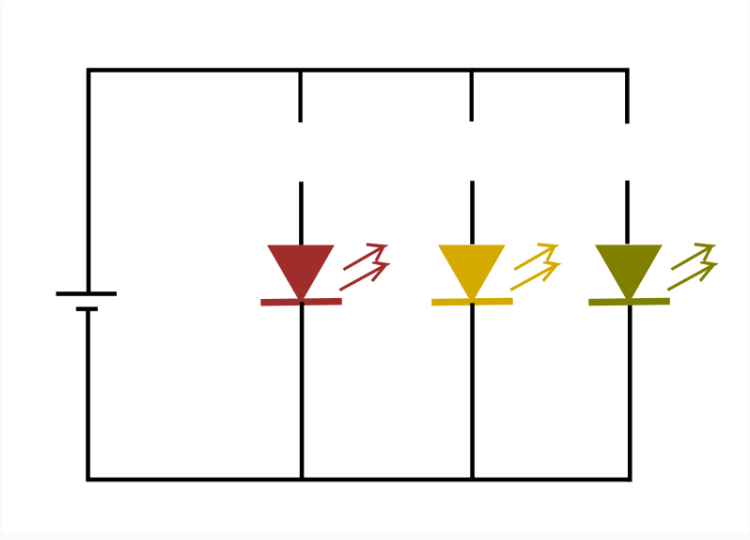
Wir trennen Grün auf

Ohne Calliope 6



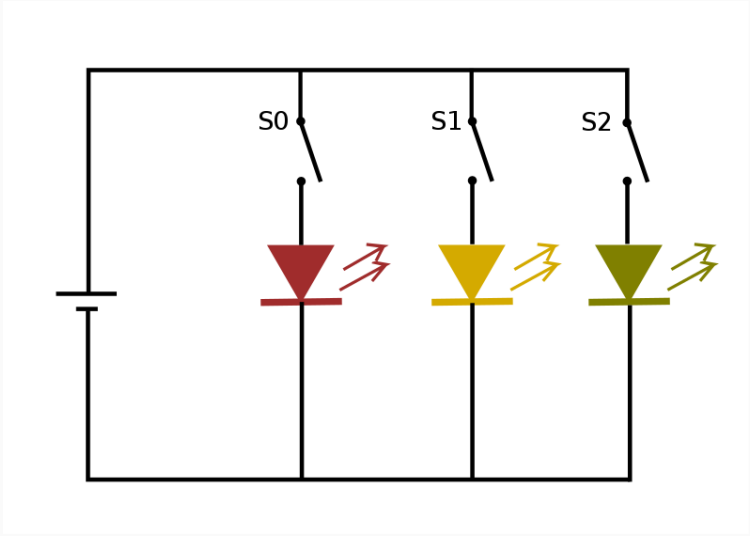
Wir trennen Gelb auf

Ohne Calliope 7



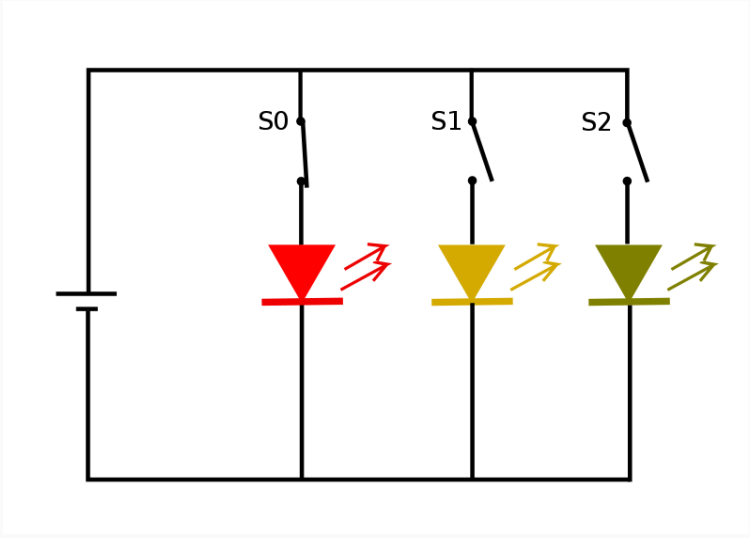
Wir trennen Rot auf

Ohne Calliope 8



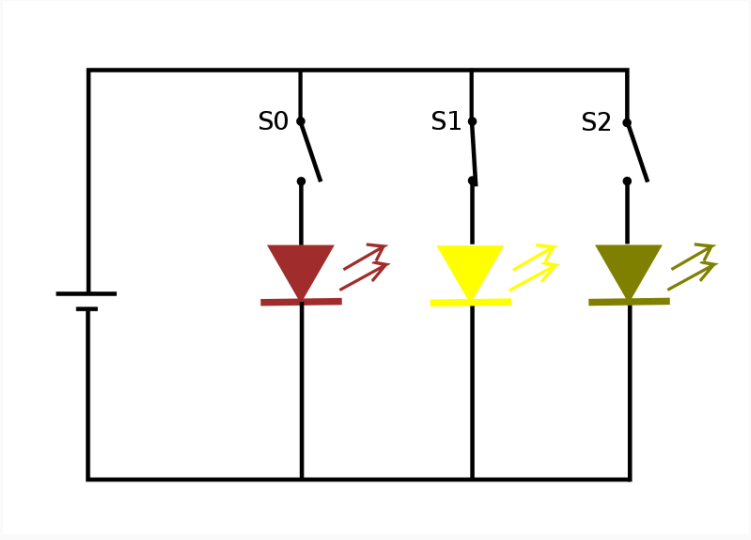
Wir bauen Schalter ein (S0 - S2)

Ohne Calliope 9



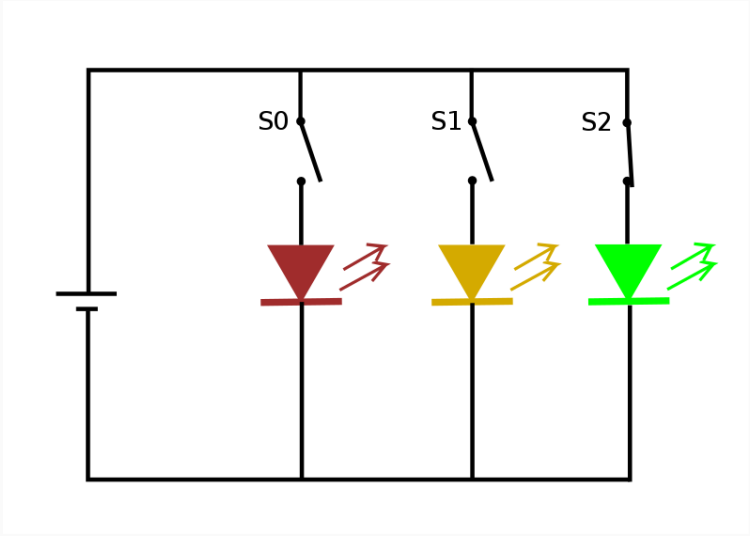
Schalter S0 einschalten : Rot

Ohne Calliope 10



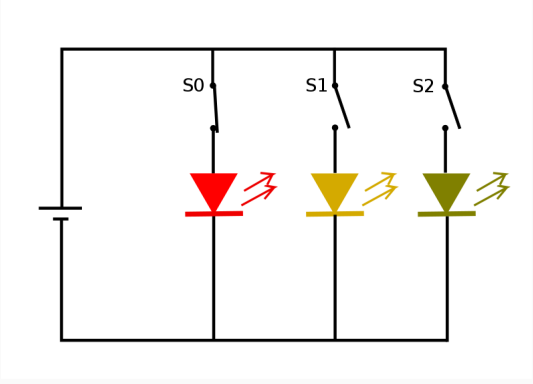
Schalter S1 einschalten : Gelb

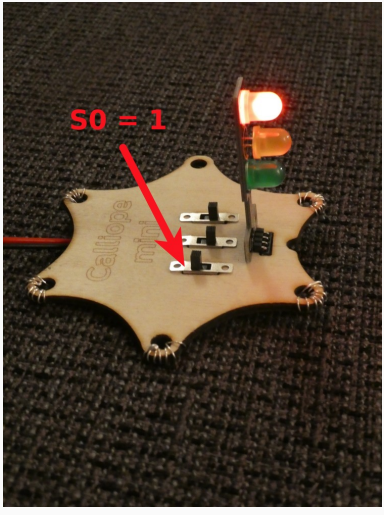
Ohne Calliope 11



Schalter S2 einschalten : Grün

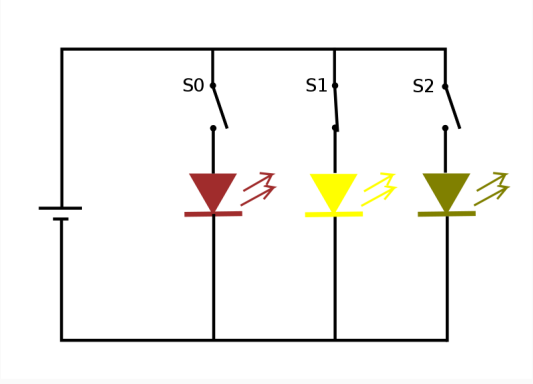
Ohne Calliope 12

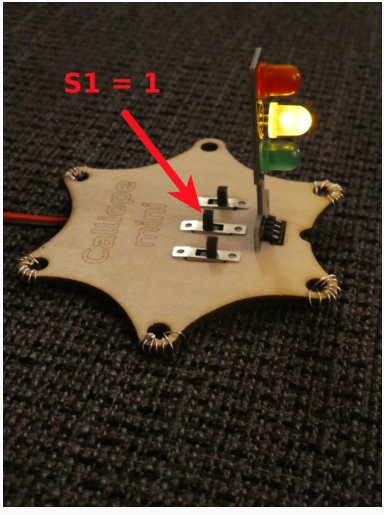




Schalter S0 einschalten : Rot

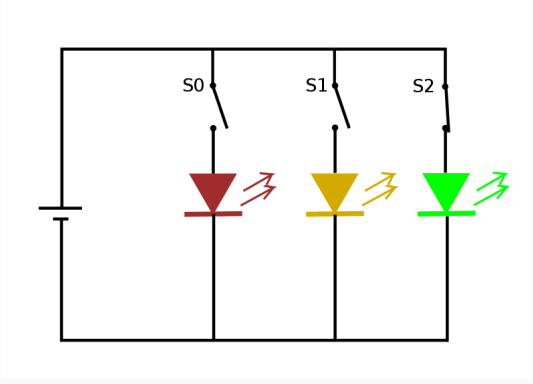
Ohne Calliope 13

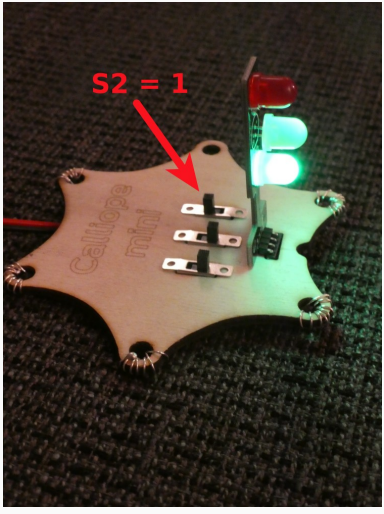




Schalter S1 einschalten : Gelb

Ohne Calliope 14





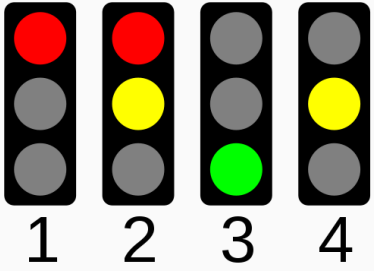
Schalter S2 einschalten : Grün

Wie die Ampel funktioniert

- Alle Schalter aus => Alle Lampen aus
- S0 = 0 : Rot aus
- S0 = **1** : Rot **ein**
- S1 = 0 : Gelb aus
- S1 = **1** : Gelb **ein**
- S2 = 0 : Grün aus
- S2 = **1** : Grün **ein**


Ein Ampel-Zyklus (1)

- Alle Lampen aus
- Rot ein => **Ampel Rot (1)**
- Rotzeit abwarten (z.B. 5 sek)
- Gelb ein => **Ampel Rot-Gelb (2)**
- Rot-Gelbzeit warten (z.B. 1 sek)
- Rot aus
- Gelb aus
- Grün ein => **Ampel Grün (3)**
- Grünzeit warten (z.B. 5 sek)
- Grün aus
- Gelb ein => **Ampel Gelb (4)**
- Gelbzeit warten (z.B. 1 sek)
- Wieder von vorne



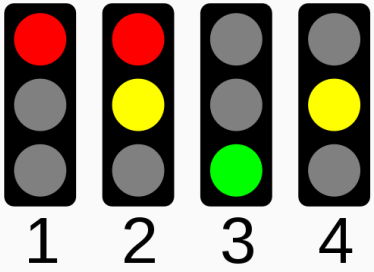
Ein Ampel-Zyklus (2)

- Alle Lampen aus
- Rot ein => **Ampel Rot**
- Rotzeit abwarten (z.B. 5 sek)
- Gelb ein => **Ampel Rot-Gelb**
- Rot-Gelbzeit warten (z.B. 1 sek)
- Rot aus
- Gelb aus
- Grün ein => **Ampel Grün**
- Grünzeit warten (z.B. 5 sek)
- Grün aus
- Gelb ein => **Ampel Gelb**
- Gelbzeit warten (z.B. 1 sek)
- Wieder von vorne

- S0,S1,S2 =0 => Lampen aus
- S0=1 => **Ampel Rot**
- Rotzeit abwarten (z.B. 5 sek)
- S1=1 => **Ampel Rot-Gelb**
- Rot-Gelbzeit warten (z.B. 1 sek)
- S0=0
- S1=0
- S2=1 => **Ampel Grün**
- Grünzeit warten (z.B. 5 sek)
- S2=0
- S1=1 => **Ampel Gelb**
- Gelbzeit warten (z.B. 1 sek)
- Wieder von vorne

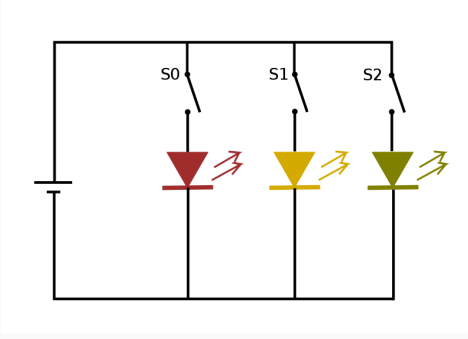

Ein Ampel-Zyklus (3)

- S0,S1,S2 =0 => Lampen aus
- S0=1 => **Ampel Rot (1)**
- Rotzeit abwarten (z.B. 5 sek)
- S1=1 => **Ampel Rot-Gelb (2)**
- Rot-Gelbzeit warten (z.B. 1 sek)
- S0=0
- S1=0
- S2=1 => **Ampel Grün (3)**
- Grünzeit warten (z.B. 5 sek)
- S2=0
- S1=1 => **Ampel Gelb (4)**
- Gelbzeit warten (z.B. 1 sek)
- Wieder von vorne

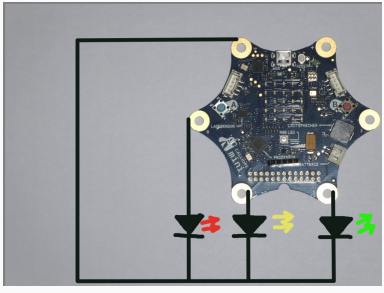


Ersetzen Hardware

Wir ersetzen Schalter

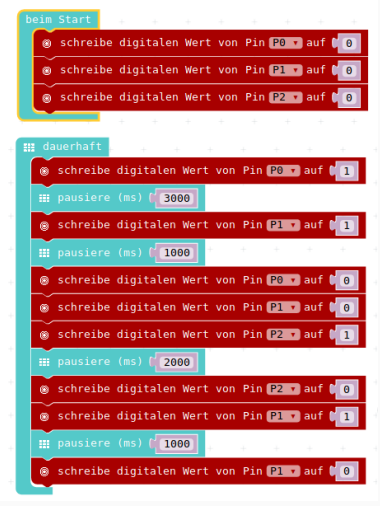


Durch den Calliope



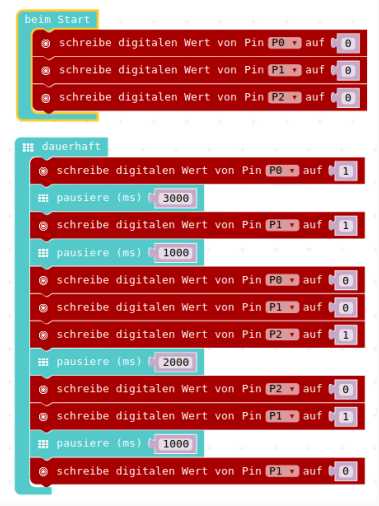
Ersetzen Software : Schalter S durch Pin P

- S0,S1,S2 =0 => Lampen aus
- S0=1 => **Ampel Rot (1)**
- Rotzeit abwarten (z.B. 5 sek)
- S1=1 => **Ampel Rot-Gelb (2)**
- Rot-Gelbzeit warten (z.B. 1 sek)
- S0=0
- S1=0
- S2=1 => **Ampel Grün (3)**
- Grünzeit warten (z.B. 5 sek)
- S2=0
- S1=1 => **Ampel Gelb (4)**
- Gelbzeit warten (z.B. 1 sek)
- Wieder von vorne

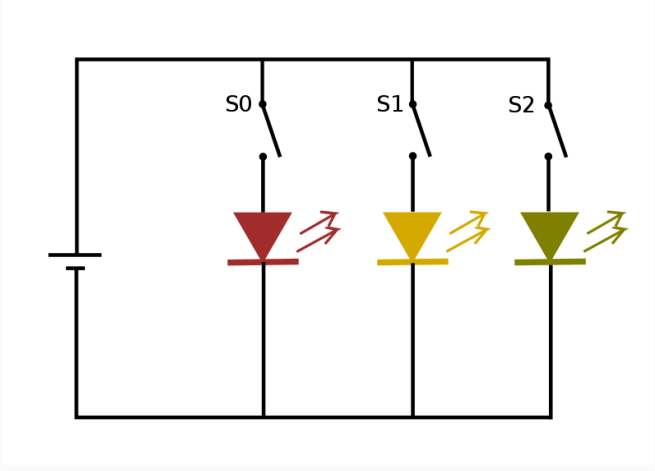


Ersetzen Software : Schalter S durch Pin P

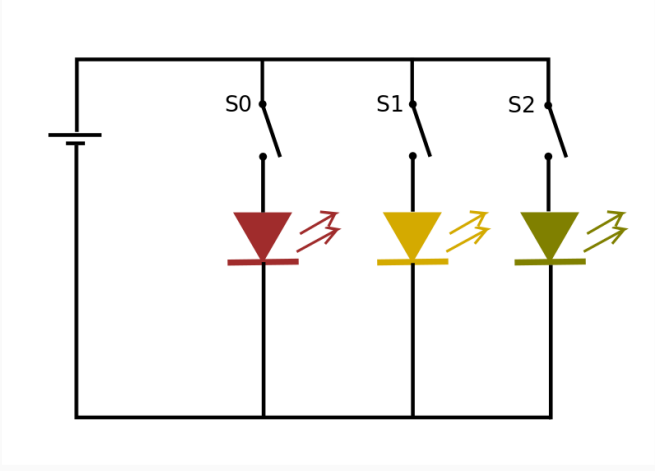
- Das Bild auf der rechten Seite
- ist das Programm vom letzten Mal
- es entspricht fast der Anweisung mit den Schaltern
- Im Programm setzen wir unten Grün auf aus
- In der Schalt-Anweisung:
- Schalten wir oben **ALLE** aus



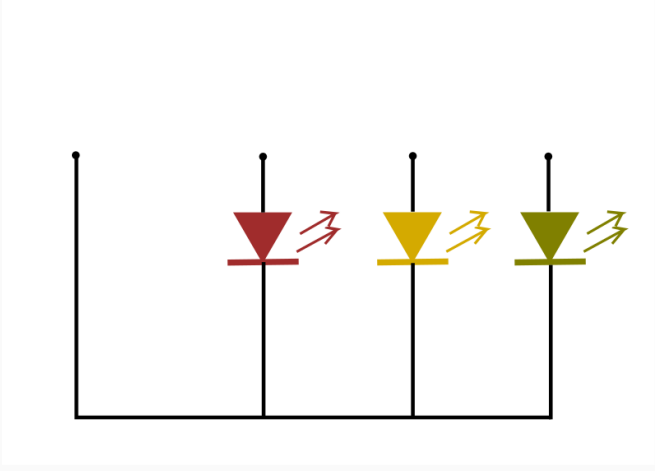
Anschluss LEDs (1)



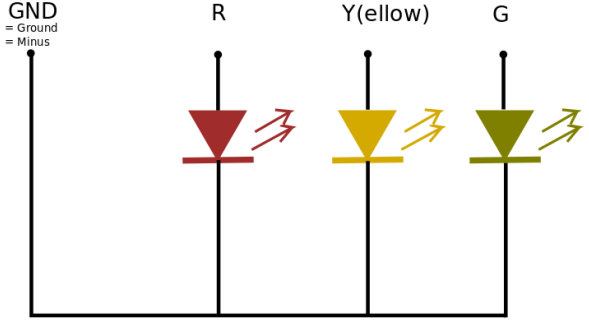
Anschluss LEDs (2)



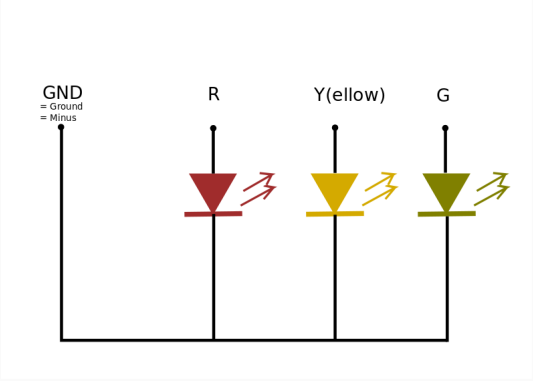
Anschluss LEDs (3)



Anschluss LEDs (4)



Anschluss LEDs (5)



Lizenz/Copyright-Info


04_03_Wenn-Dann

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Logik, Vergleiche, Wahrheit

Weltmeisterschaft / Fussball-Wetten / Vergleiche

- Wer hat die Fussball-Weltmeisterschaft geschaut?
- Wer hat dabei Wetten gemacht" ?
- **Wenn** die Schweiz ein Tor mehr schiesst als Deutschland,
- **dann** bekomme ich von Dir zwei zusätzliche Panini-Bilder?


Weltmeisterschaft / Fussball-Wetten / Vergleiche

- Wer streitet hin und wieder mit seinem Freund?
- **Wenn** Du mir das Lego nicht gibst,
- **dann** bekommst Du von mir keine Schokolade mehr.

und so weiter...


Das alles sind Vergleiche, die normalerweise dann im Anschluss überprüft werden können und sich entweder als wahr oder falsch erweisen.

Je nachdem, ob wahr oder falsch, wird dann etwas gemacht:

- Du bist nicht mehr mein Freund
- Du musst mir zwei Panini geben
- Ich muss Dir eine Schoki geben
- ...


Das ist eines der wichtigen Eigenschaften auch beim Programmieren:

- Einen **Vergleich** machen, der ein Ergebnis hat,
- dies ist im Allgemeinen **wahr** oder **falsch**
- Basierend auf dem **wahr** oder **falsch**
- wird dann etwas unterschiedliches **gemacht**,
- es wird eine **Aktion** ausgelöst.

Das wollen wir jetzt auch machen


Das Menu Logik

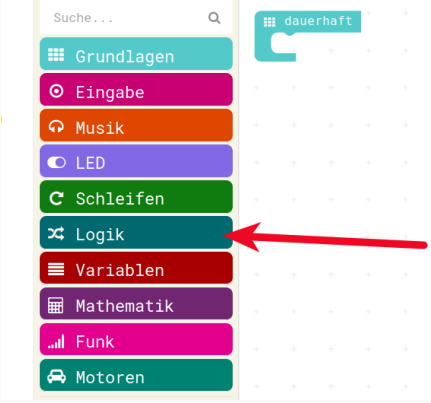


Figure 1: LogikMenu

Das Menu Logik

enthält die Wenn-Dann Programmierung, es enthält die Vergleiche die wir machen wollen und es enthält auch “Wahr” und “Falsch” - Werte

Figure 2: Logik Wenn Dann Ansonsten

Wenn Dann in der Dauerschleife

Das Wenn-Dann ziehen wir in den Arbeitsbereich in die Dauer-Schleife

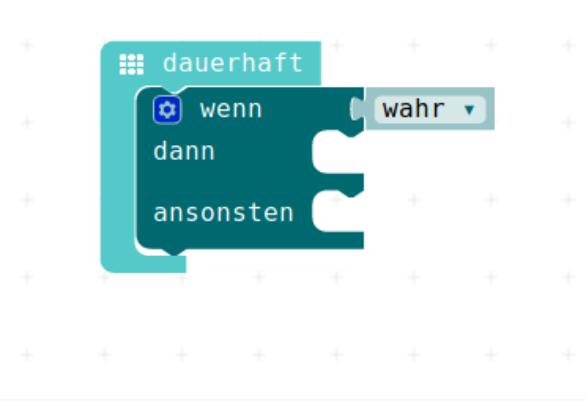


Figure 3: Wenn Wahr

Symbole in der Wenn-Dann

Nun können wir mit zwei einfachen Symbolen auf unserem “Display” anzeigen, wie sich das Wenn-Dann verhaält

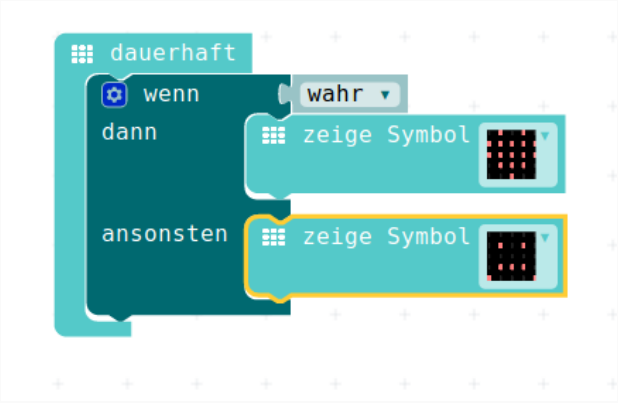


Figure 4: Wenn Wahr Symbol

Wenn-Dann Ergebnis Wahr

In der Wenn-Dann-Abfrage kommt immer oben die **Aktion**, die gemacht werden soll, wenn die Aussage **wahr** ist, darunter kommt das, was gemacht werden soll, wenn die Aussage sich als **falsch** erweist.

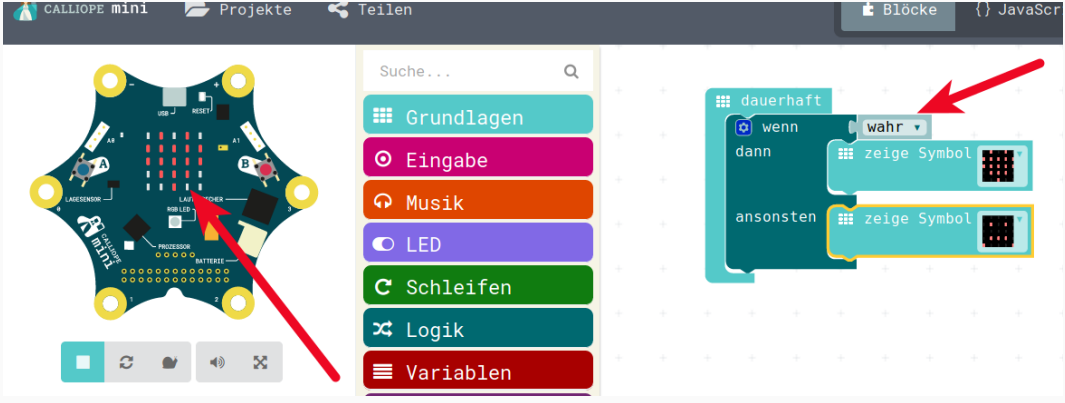


Figure 5: Wenn Wahr Symbol_Wahr

Wenn-Dann Ergebnis Falsch

So sieht das Ganze aus, wenn die Aussage **Falsch** ist.

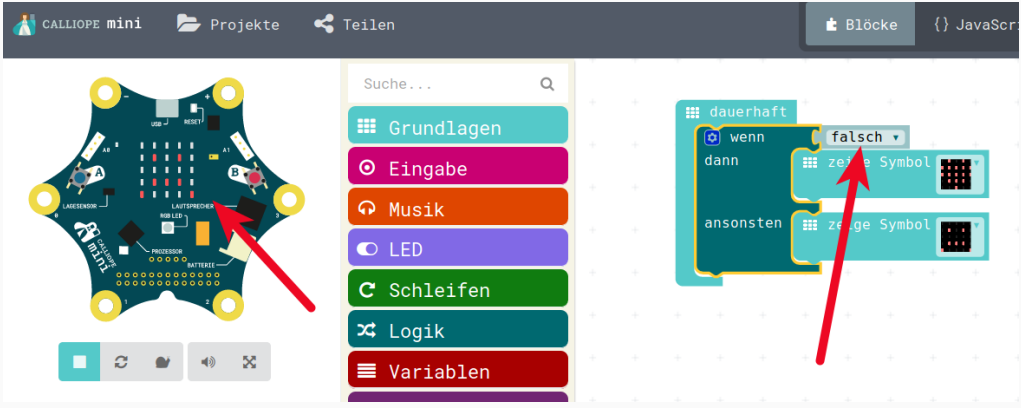


Figure 6: Wenn Wahr Symbol Falsch

Einsatz der farbigen RGB-Leucht-Diode

Wir haben ja auch eine farbige Leuchtdiode auf dem Calliope, die soll nun zum Einsatz kommen.

Anstatt Symbole auf dem 5x5 - roten LED-Display wollen wir die LED in unterschiedlichen Farben leuchten lassen.

Die LED befindet sich auch unter Grundlagen (auch zu erkennen an der Farbe!)

Farbigen RGB-Leucht-Diode in Wenn-Dann

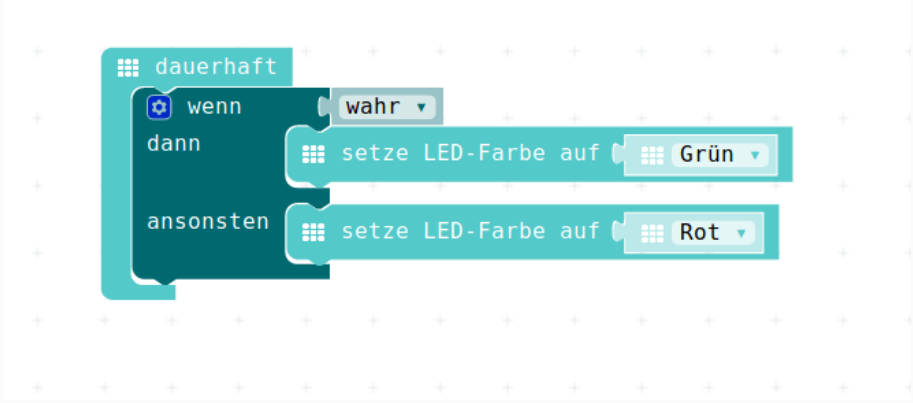


Figure 8: RGB Led In Wenn Wahr

Farbigen RGB-Leucht-Diode leuchtet Grün

Da momentan der Wert **Wahr** in die Abfrage reingeben wird, leuchtet unsere RGB-Led in Grün.

Jetzt ist übrigens ein guter Moment, um das nicht nur im Simulator auszuprobieren, sondern das Programm als HEX-Datei zu speichern und auf dem Calliope-Board auszuprobieren.

Vergleiche

Nun wollen wir aber **echte** Vergleiche machen, dazu können wir zum Beispiel zwei Zahlen miteinander vergleichen und das Ergebnis auswerten:

- Zwei ist grösser als Fünf : **Falsch**
- Sechs ist grösser als Fünf : **Wahr**
- Sechs ist grösser als Sechs : **Falsch**
- Zehn ist kleiner als Sechs : **Falsch**
- Zehn ist gleich Zehn : **Wahr**
- Acht ist gleich Neun : **Falsch**


Vergleiche

Das kleiner-Zeichen habt Ihr in Mathematik wahrscheinlich auch noch nicht gehabt, aber es ist eigentlich **selbst** sprechend:

- Kleinere Zahl $<$ Grössere Zahl
- Grössere Zahl $>$ Kleinere Zahl

Damit wird:

- Zwei ist grösser als Fünf : $2 > 5$: **Falsch**
- Sechs ist grösser als Fünf : $6 > 5$: **Wahr**
- Sechs ist grösser als Sechs : $6 > 6$: **Falsch**
- Zehn ist kleiner als Sechs : $10 < 6$: **Falsch**
- Zehn ist gleich Zehn : $10 = 10$: **Wahr**
- Acht ist gleich Neun : $8 = 9$: **Falsch**

Vergleiche

Figure 10: Logik Vergleiche

Vergleich aus dem Menu holen

und anstelle von “**Wahr**” in die Wenn-Dann reinsetzen

Nun haben wir einen - **noch sinnlosen** - Vergleich:

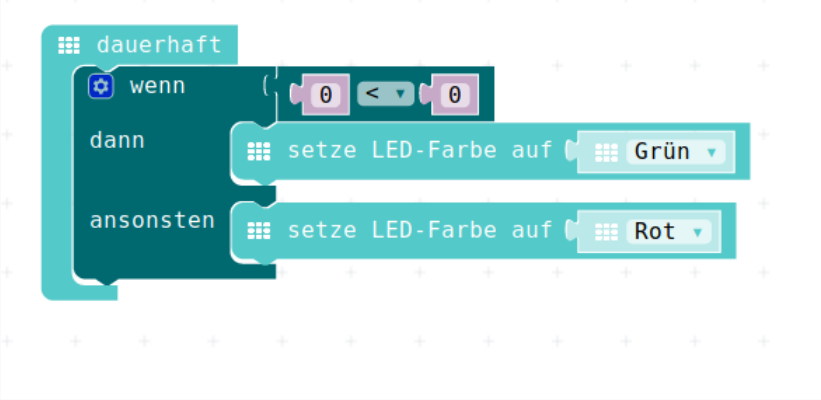


Figure 11: Wenn Dann Vergleich

Vergleich auf > grösser setzen

Mit dem kleinen Dreieck beim Vergleich können wir nun denn kleiner-Vergleich auf einen Grösser-Vergleich umbauen.

Figure 12: Wenn Dann Groesser

Vergleich auf sinnvolle Werte

Nun nehmen wir zwei Werte in den Vergleich.

Die Werte sind eigentlich egal, ich habe 22 und 21 genommen, das wäre gut, wenn Ihr das auch macht, dann können wir später sehen warum...

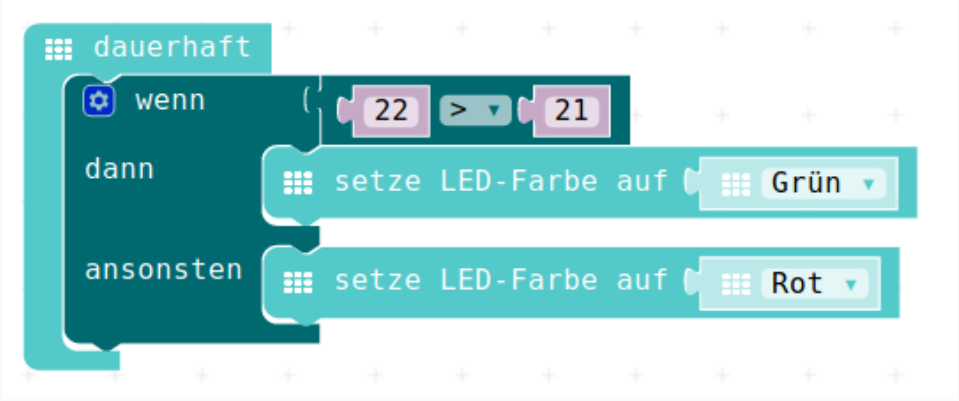


Figure 13: Wenn Dann Groesser:True

Vergleich ergibt grünes LED

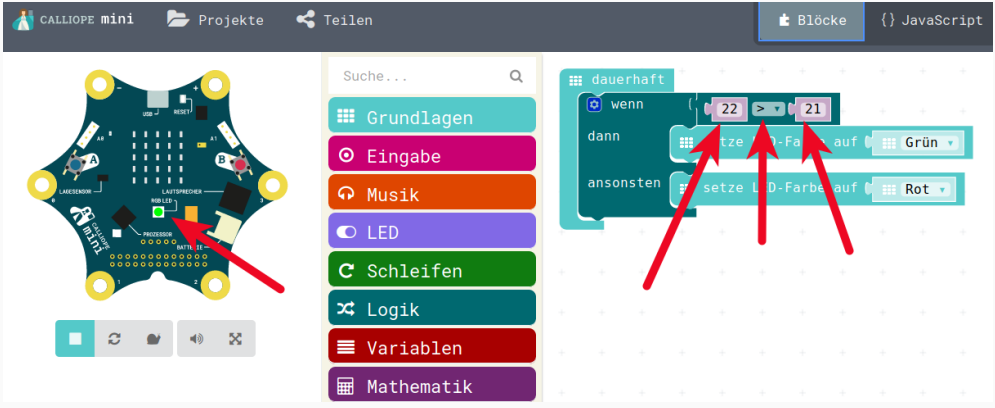


Figure 14: Wenn Dann Groesser:True

JavaScript-Code

Java-Script-Code

```
basic.forever(() => {
    if (22 > 21) {
        basic.setLedColor(Colors.Green)
    } else {
        basic.setLedColor(Colors.Red)
    }
})
```

Download Hex-Code:

Hex-code


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

04_04_TemperaturSensor

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Der Temperatur-Messer

Sensor/Eingang/Input

Neben vielen anderen Sensoren/Eingängen, ein paar davon haben wir schon kennengelernt, hat der Calliope auch einen Temperatur-Sensor.

Diesen wollen wir nun als sinnvollen Eingangs-Wert für unsere Wenn-Dann Abfrage benutzen.

Davor wollen wir aber den Sensor selbst kurz kennenlernen, sehen wie man den abfrägt und was er für Werte liefert.

Dazu räumen wir unsere Dauerschlefe frei:

Freiräumen der Dauerhaft-Schleife

Herausziehen der bisherigen **Wenn-Dann** -Programmierung zur Seite. **NICHT** löschen, wir wollen das später noch benutzen

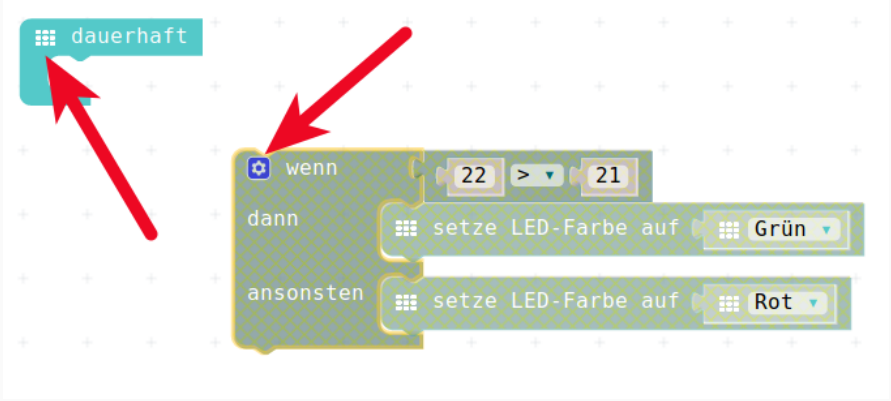


Figure 1: Schleife frei räumen

Das Menu Eingabe

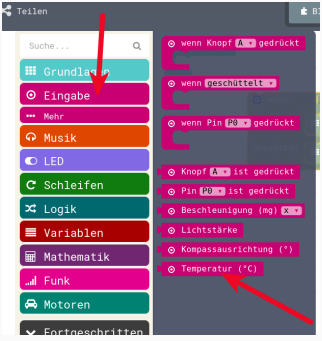


Figure 2: Menu Input

enthält eine Spezial-Variable, eine Input-Variable, namenes **Temperatur**. Um diese sinnvoll weiter zu verwenden, legen wir Menu **Variablen** eine eigene Variable an, die wir zum Beispiel **AktuelleTemperatur** nennen.

Variable anlegen

Nun legen wir uns wieder eine neue Variable namens **AktuelleTemperatur** an.

Figure 3: Variable anlegen

Variable benennen

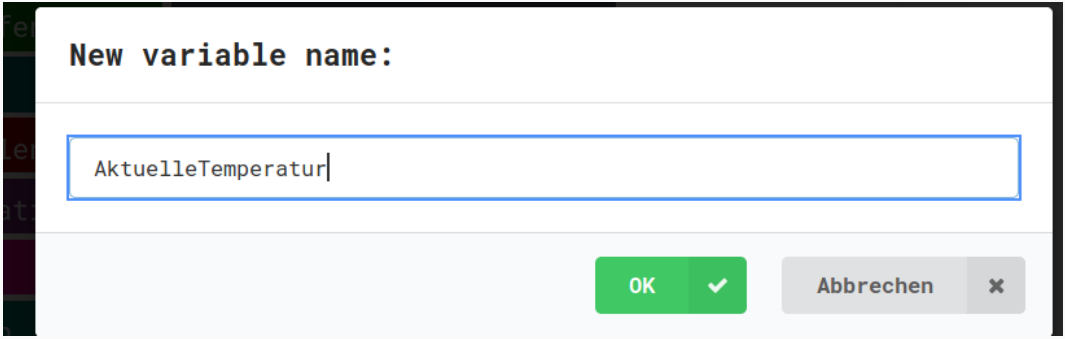


Figure 4: Variable benennen

Temperatur auslesen

Nun belegen wir also die neu angelegte Variable **AktuelleTemperatur** mit der Temperatur, wie sie aus dem Eingabe-Menu kommt.

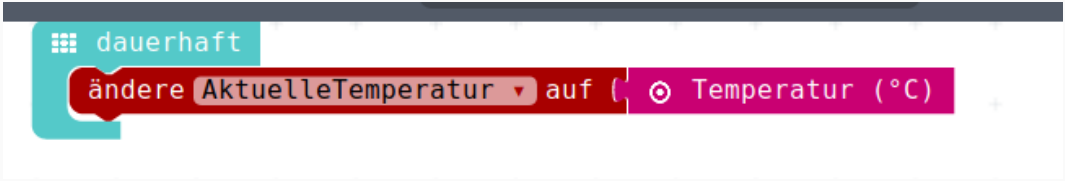


Figure 5: Variable zuweisen

Temperatur anzeigen

und holen uns noch aus dem Grundlagen-Menu das **zeige Nummer** und zeigen damit die Variable an, die die aktuelle Temperatur beinhaltet.

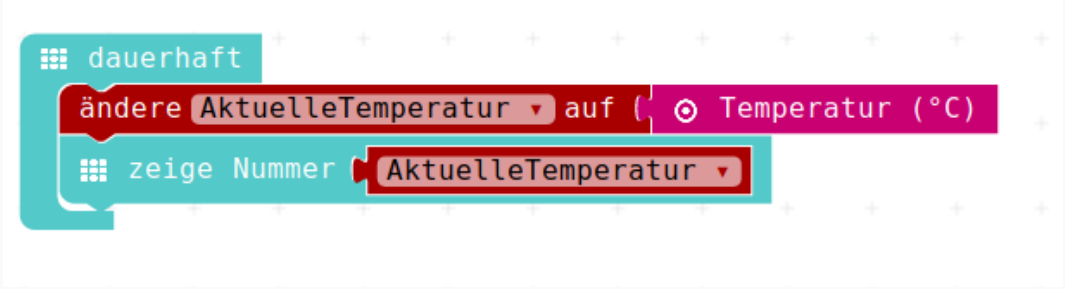


Figure 6: Temperatur anzeigen

Temperatur im Simulator

Nun schauen wir uns das im Simulator an:

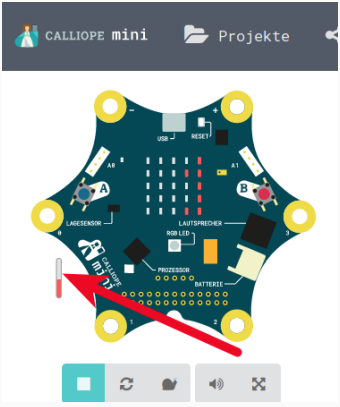


Figure 7: Temperatur im Simulator

Mit der Maus kann man an diesem Thermometer die Temperatur verändern.

Temperatur im Simulator

Die Ausgabe kann noch etwas verbessert werden:

- die Zahlen “rauschen” nur durch und man weiss nicht ob das nun 21 oder 12 sind
- Man sieht nicht, dass es sich um eine Temperatur in Grad Celcius (°C) handelt.

Verbesserung der Anzeige

Mit einem vereinfachten “**°C**” als Symbol und ein paar “warte ms” und Bildschirmlöschen (das ist alles im Menu Grundlagen, zum Teil in “. . . Mehr”) gibt die Anzeige dann schon was her.

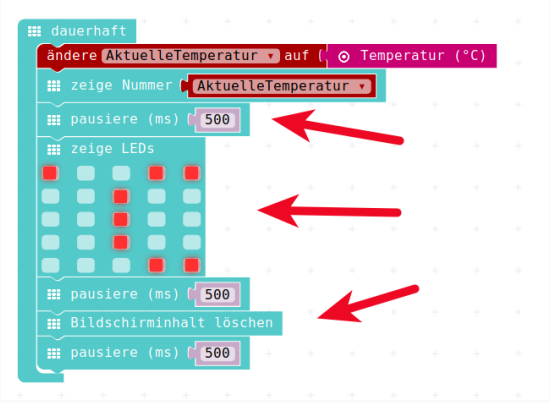


Figure 8: Verbesserte Anzeige

Download in der Calliope

Jetzt ist der Programm-Code eigentlich gut genug, um eine echte Messung in unserem echten Calliope durchzuführen.

Wir laden das Programm dazu auf den Calliope:

- Dem Ganzen unten einen sinnvollen Namen geben (z.B. Temperatur-Messer_01)
- Den Speichern-Knopf (Diskette) drücken
- Das erzeigte und geladene HEX-File im **Download** - Ordner finden
- Das HEX-File kopieren , “STRG-C”
- Den Calliope-Mini anschliessen
- Das HEX-File auf dem Calliope “fallen lassen”, “Einfügen”, “STRG-V”
- Der Calliope sollte anfangen das neue Programm in seinen Speicher zu übertragen.

JavaScript-Code

Java-Script-Code

```
let AktuelleTemperatur = 0
basic.forever(() => {
    AktuelleTemperatur = input.temperature()
    basic.showNumber(AktuelleTemperatur)
    basic.pause(500)
    basic.showLeds(`
        # . . # #
        . . # . .
        . . # . .
        . . # . .
        . . . # #
        `)
    basic.pause(500)
    basic.clearScreen()
    basic.pause(500)
})
```


wnload Hex-Code

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

04_05_TemperaturAmpel

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Die Temperatur-Ampel

Wiedereinbau Wenn-Dann

Nun können wir aus der Wenn-Dann Übung von vorher und dem kleinen Temperatur-Programm von eben eine kleine Temperatur-“**Ampel**” machen.

Dazu bauen wir das vorher “beiseite” gelegte **Wenn-Dann** wieder ein.

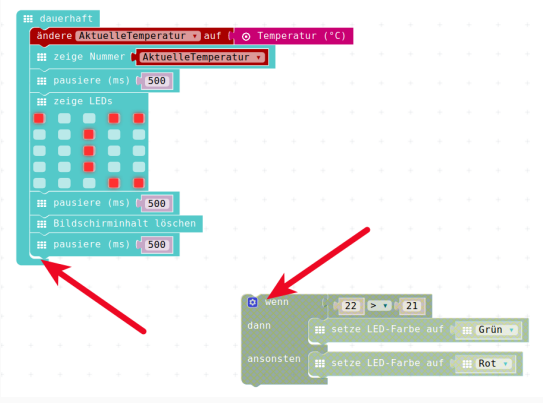


Figure 1: Wieder Einbau

Wiedereinbau Wenn-Dann

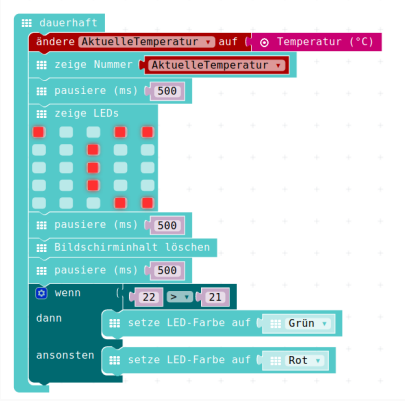


Figure 2: Eingebaut

Verwendung sinnvoller Werte

Wir möchten im ersten Schritt, "**Alles im Grünen Bereich**" anzeigen, wenn die Temperatur eine **gute** Temperatur hat.

Dazu sagen wir : Alles was grösser 21 °C ist, ist gut.

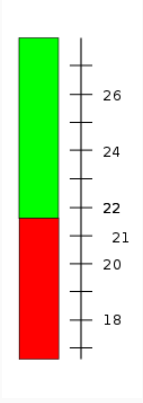


Figure 3: Thermometer

Verwendung sinnvoller Werte

Das heisst: Wir müssen unsere Wenn-Dann-Konstruktion so umbauen, dass folgender Satz/Aussage abgebildet wird:

- Wenn die Temperatur grösser als 21 °C ist,
  - **dann** soll die RGB-LED grün leuchten,
  - **ansonsten** soll sie rot leuchten.

Verwendung sinnvoller Werte

- Nun ergeben auch die vorhin “ganz willkürlich gewählten” Werte 22 und 21 etwas Sinn...
- Wir ersetzen die konstante 22 in der Wenn-Dann-Abfrage durch die jetztige Temeperatur.
- Diese befindet sich in der Variable **AktuelleTemperatur**
- Dazu holen wir uns die Variable **AktuelleTemperatur** aus dem Menu Variablen

Verwendung sinnvoller Werte

Figure 4: Variable holen

Verwendung sinnvoller Werte

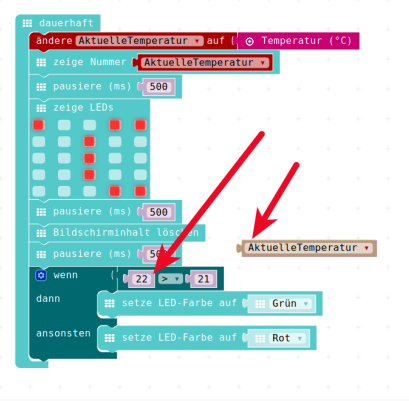


Figure 5: Variable liegt da

Verwendung sinnvoller Werte

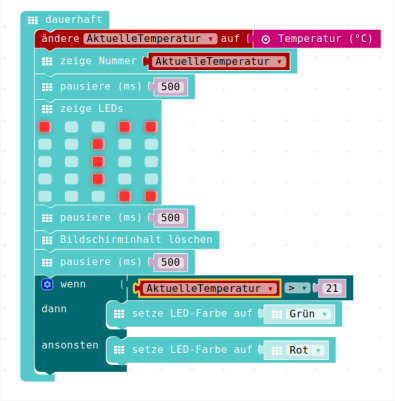


Figure 6: Variable wird benutzt

Damit lässt sich im Simulator schon mal ausprobieren, wie unsere Temperatur-Ampel reagiert.

Benutzung im Simulator

Beim Starten ist im Simulator die Temperatur immer 21 °C, das ist nach unseren Wünschen genau die Grenze. Erst wenn die **AktuelleTemperatur** grösser als 21 °C ist, dann wird die Anzeige grün. Das können wir im Simulator ausprobieren und dann natürlich auch wieder in den Calliope laden um es in der richtigen Hardware mit echten Werten zu testen.

Download in der Calliope

Jetzt ist der Programm-Code eigentlich gut genug, um eine echte Messung in unserem echten Calliope durchzuführen.

Wir laden das Programm dazu auf den Calliope:

- Dem Ganzen unten einen sinnvollen Namen geben (z.B. Temperatur-Messer_01)
- Den Speichern-Knopf (Diskette) drücken
- Das erzeigte und geladene HEX-File im **Download** - Ordner finden
- Das HEX-File kopieren
- Den Calliope-Mini anschliessen
- Das HEX-File auf dem Calliope “fallen lassen”, “Einfügen”, “CTRL-V” * Der Calliope sollte anfangen das neue Programm in seinen Speicher zu übertragen.

(So, das war jetzt aber das letzte Mal eine detaillierte Anleitung zum Laden des Programms in den Calliope, in Zukunft kommt nur noch der Hinweis, dass wir das Programm in den Calliope laden...)

JavaScript-Code

Java-Script-Code

```
let AktuelleTemperatur = 0
basic.forever(() => {
    AktuelleTemperatur = input.temperature()
    basic.showNumber(AktuelleTemperatur)
    basic.pause(500)
    basic.showLeds(`
        # . . # #
        . . # . .
        . . # . .
        . . # . .
        . . . # #
        `)
    basic.pause(500)
    basic.clearScreen()
    basic.pause(500)
    if (AktuelleTemperatur > 21) {
        basic.setLedColor(Colors.Green)
    } else {
```


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

04_06_TemperaturAmpelBesser

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Die verbesserte Temperatur-Ampel

Echter grüner Bereich

Nun wollen wir die Temperatur-Ampel etwas verbessern.

- Wir legen einen grünen Bereich zwischen grösser als 21 °C und bis 25 °C fest
- Wenn die Temperatur kleiner/gleich 21 °C ist, soll das Licht blau sein, es ist uns zu kalt (blau wie am Wasserhahn)
- Wenn die Temperatur in unserem "grünen" Bereich ist, dass soll natürlich auch die LED grün leuchten
- Wenn die Temperatur grösser als 25 °C ist, dann soll die LED rot leuchten (rot wie am Wasserhahn)

Echter grüner Bereich

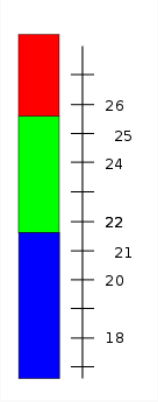


Figure 1: Neue Temperatur-Bereiche

Wenn-Dann erweitert

Ebenso wie im “echten Leben” kann man auch das Wenn-Dann - Programmier-Konstrukt erweitern.

- **Wenn** xyz **Dann** macheDas
- **Ansonsten Wenn** abc **Dann** macheJenes
- **Ansonsten Wenn** def **Dann** machedochnochwasanderes
- **Ansonsten** MacheEinfachIrgendwas

Mit solch einem Konstrukt können wir nun unserer Temperatur-Abfrage erweitern um eine zusätzliche Abfrage grösser 25° C
Und die Farben müssen wir natürlich auch noch anpassen.

Erweiterung in der Programmier-Oberfläche

Um das Wenn-Dann - Konstrukt in der Programmier-Oberfläche zu erweitern, muss im "Wenn-Dann-Puzzle-Stück" das Zahnrädchen benutzt werden.
Das öffnet die Tool (= Werkzeug)-Box des Wenn-Dann-Puzzleteils.

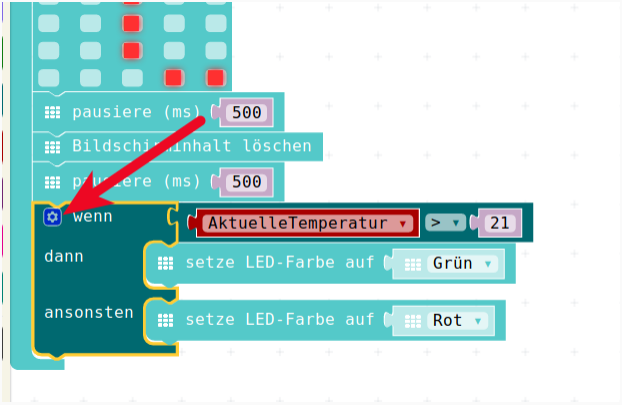
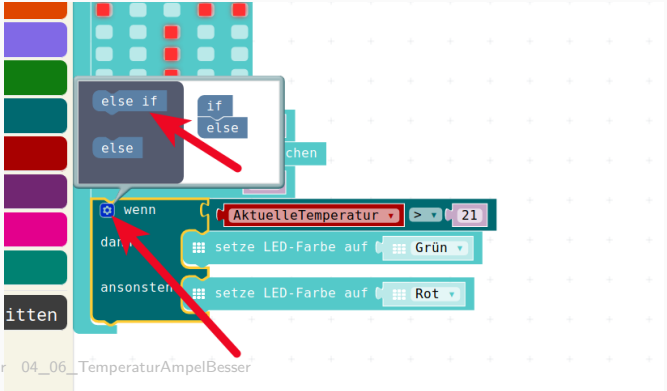


Figure 2: ToolBox

Erweiterung in der Programmier-Oberfläche

Leider merkt man bei diesen Spezialitäten, dass die Programmier-Oberfläche noch nicht an allen Stellen vom Englischen ins Deutsche übersetzt wurde, dann hier kommen plötzlich:

- **if** anstatt **wenn**
- **else if** anstatt **sonst wenn**
- **else** anstatt **sonst**



Erweiterung in der Programmier-Oberfläche

Die Benutzung ist hier auch etwas gewöhnungs-bedürftig:

- Um unser “Wenn-Dann”-Konstrukt um ein zusätzliches **sonst wenn** zu erweitern zieht man das **else if** oben in der Toolbox von der linken Hälfte auf die rechte Hälfte rüber, zwischen das **if** und das **else**.
- Dies führt unten zu Erweiterung der Wenn-Dann-Abfrage um eine **Ansonsten Wenn** - Konstruktion.
- Mann kann auch durchaus noch mehrere dieser **else if** einbauen, wenn man noch mehr Fälle unterscheiden will.
- Für unsere Zwecke reicht allerdings dieses eine.

Erweiterung in der Programmier-Oberfläche

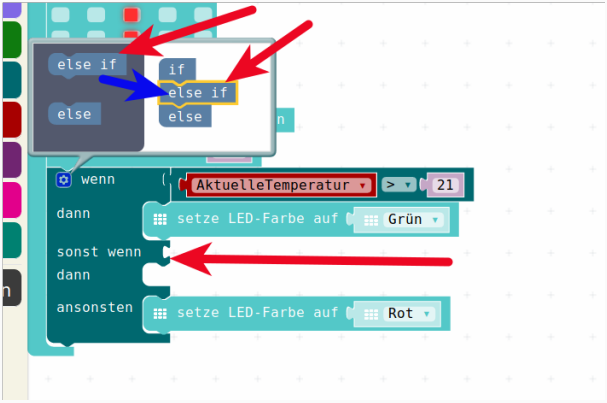


Figure 4: Toolbox Miniatur

Einbau der zusätzlichen Abfragen

Nun können wir also in die zusätzlichen Abfragen unsere weiteren Überprüfungen auf Temperatur $>$ 25 einklicken (am Besten die Überprüfung per rechter Maustaste von oben kopieren) und die LED-Farben-Setzen befehle einklicken und die Farben entsprechend ändern.

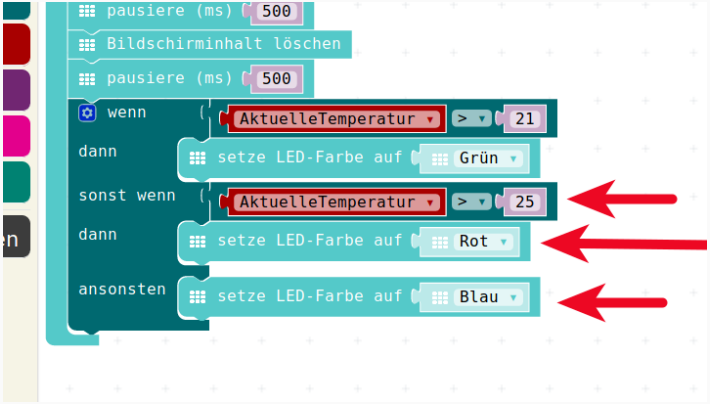


Figure 5: Neues Programm Elseif

Ein Bug (ein Fehler) !

- Es geht nicht!
- Wir bekommen kein **rot** zu sehen!
- Was ist falsch?

Dazu können wir mal versuchen, die Temperatur auf $> 26°C$, also z.B. 30°C einzustellen und dann das Programm anschauen / beobachten.

Dazu eignet sich die **Schnecke**.
Die lässt das Programm im Simulator im Schneckentempo ablaufen und zeigt jeweils durch Hervorheben an, welcher Schritt gerade ausgeführt wird.

Ein Bug (ein Fehler) !

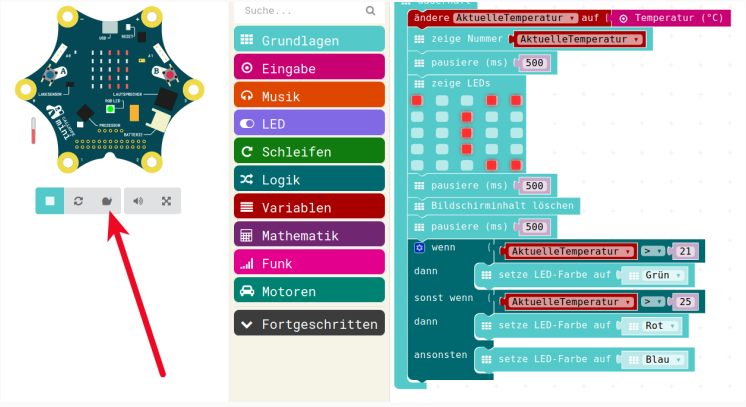


Figure 6: Bugsuche Schnecke

Das ist schonmal eine grosse Hilfe und könnte uns bei der Fehlersuche unter die Arme greifen.

Ein Bug (ein Fehler) !

Was passiert?
Auch eine Temperatur von z.B. 30°C, die ja als rot angezeigt werden soll, geht durch das ganze Wenn-Dann-Konstrukt durch. Wenn irgendeine Bedingung erfüllt ist, dann wird die zugehörige Aktion durchgeführt und dann das Konstrukt verlassen.

- Als erstes wird die gemessene aktuelle Temperatur von 30°C überprüft, ob sie grösser ist als 21 °C.
- 30°C **IST** grösser als 21°C
- Der Vergleich liefert das Ergebnis **WAHR**
- Also wird die zueghörige Aktion durchgeführt: Setzen der Farbe auf grün
- Das **Sonst Wenn** wird gar nicht erreicht und darum dann auch die Überprüfung auf > 25°C erst gar nicht durchgeführt!

Umbau des Wenn-Dann-Konstrukts

Nachdem wir diesen Fehler gefunden haben, müssen wir unser **“Wenn-Dann”** - Konstrukt umbauen:

- als erstes Vergleich auf > 25 °C => ROT
- als zweites Vergleich auf > 21 °C => GRÜN
- ansonsten => BLAU

Vor dem Umbau spielen wir das hier einmal durch :


Vergleich auf “Papier”

gemessener Wert : **30**

- als erstes Vergleich auf $>$ 25 °C : **WAHR** $\Rightarrow$ ROT und Ende
- Ergebnis : **ROT**

Vergleich auf “Papier”

gemessener Wert : **24**

- als erstes Vergleich auf $>$ 25 °C : **FALSCH** $=>$ Weiter
- als zweites Vergleich auf $>$ 21 °C : **WAHR** $=>$ GRÜN und Ende
- Ergebnis : **GRÜN**

Vergleich auf “Papier”

gemessener Wert : **19**

- als erstes Vergleich auf $>$ 25 °C : **FALSCH** $=>$ Weiter
- als zweites Vergleich auf $>$ 21 °C : **FALSCH** $=>$ Weiter
- ansonsten $=>$ BLAU
- Ergebnis : **BLAU**

Neu zusammensetzen

Nun ziehen wir also unsere Vergleichs-Puzzle-Teile und unsere RGB-LED-Farben-Setz-Puzzle-Teile raus:

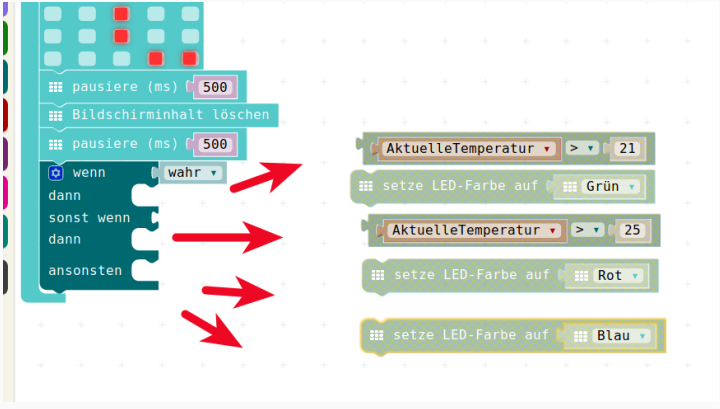


Figure 7: If Else Frei Geraeumt

Neu zusammensetzen

und setzen es wie angedacht wieder zusammen.

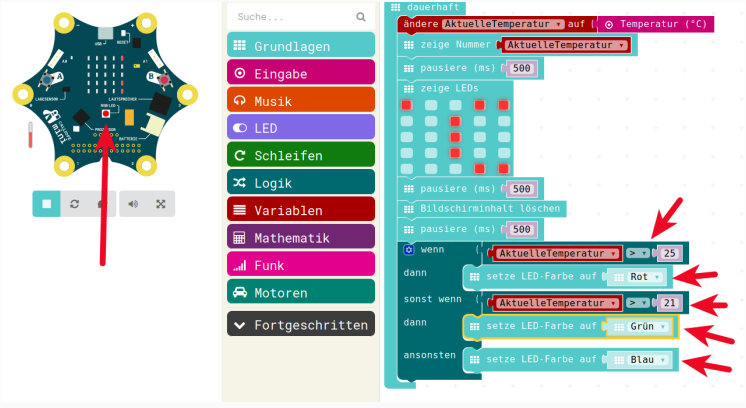


Figure 8: Neue Struktur Und Geht

Wenn wir nun die Temperatur mit der Maus im Simulator ändern, dann sehen wir, dass Farb-Anzeige unsere LED wir gewünscht funktioniert.

Download in der Calliope

Jetzt ist der Programm-Code eigentlich gut genug, um eine echte Messung in unserem echten Calliope durchzuführen.

Wir laden das Programm dazu auf den Calliope

JavaScript-Code

Java-Script-Code

```
let AktuelleTemperatur = 0
basic.forever(() => {
    AktuelleTemperatur = input.temperature()
    basic.showNumber(AktuelleTemperatur)
    basic.pause(500)
    basic.showLeds(`
        # . . # #
        . . # . .
        . . # . .
        . . # . .
        . . . # #
        `)
    basic.pause(500)
    basic.clearScreen()
    basic.pause(500)
    if (AktuelleTemperatur > 25) {
        basic.setLedColor(Colors.Red)
    } else if (AktuelleTemperatur > 21) {
```


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

05_01_LichtSensor

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Der Licht-Sensor

Sensor/Eingang/Input => Licht

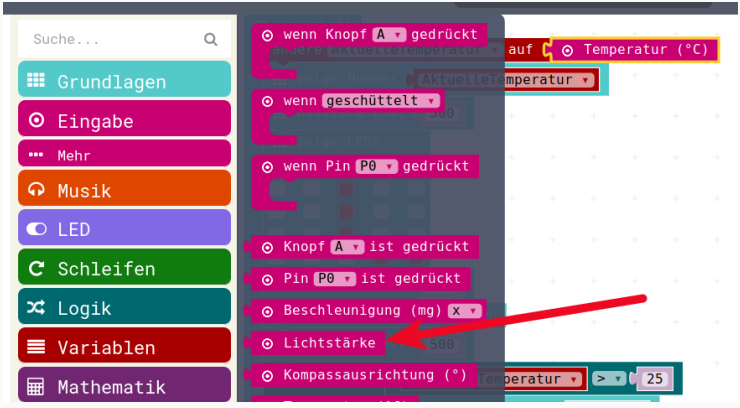
Nun wollen wir etwas mit dem ebenso im Calliope vorhandenen Licht-Sensor experimentieren.

Achtung : Weil wir faul sind, fangen wir nicht ein ganz neues Programm an, sondern wir verwenden unseren Temperatur-Sensor-Programm als Start-Punkt.

Wenn wir das Programm dann später aber sichern um es in den Calliope zu laden, dann sollten wir ihm einen sinnvollen neuen Namen geben (z.B. LichtSensor01) um nicht unser Temperatur-Sensor Programm zu überschreiben!

Nun begeben wir uns also auf die Suche nach dem Licht-Sensor.
Auch diesen finden wir im Menu Eingabe.

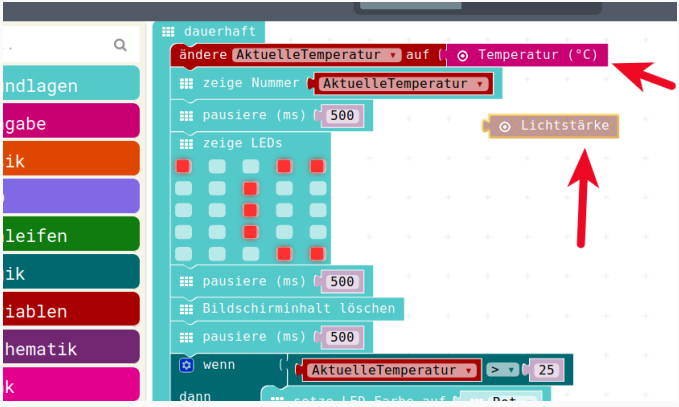
Sensor/Eingang/Input => Licht



Diese Input-Variable ziehen wir uns einfach mal auf die Arbeitsfläche, damit wir sie zur Verfügung haben.

Austausch Temperatur gegen Licht

Nun haben wir die **momentan unbenutzte** Input-Variable **Lichtstärke** auf dem Arbeitsbereich rumliegen.



Austausch Temperatur gegen Licht

Diese **tauschen** wir nun in unserem Programm gegen die Temperatur-Input-Variable aus. Wir ziehen die **Temperatur** aus dem Programm raus in die Arbeitsfläche und klicken dafür die **Lichtstärke** in das Programm ein

Variable umbenennen

- Nun haben wir schon ein theoretisch funktionierendes Programm.
- Das können wir sogar schon im Simulator anschauen.
- Aber vorher wollen wir nun doch etwas “sauber machen”

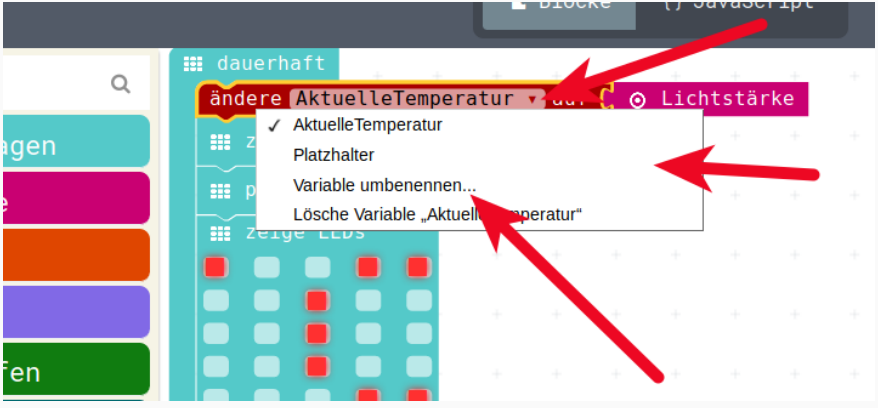
Bei der Programmierung ist fast nichts schlimmer, als Variablen-Namen, die falsch benannt sind.

- Das haben wir jetzt aber gerade gemacht, wir haben ein Programm geschrieben,
- welches die **Lichtstärke** misst u
- nd diese in einer Variable namens **AktuelleTemperatur** ablegt
- und anschliessend damit weiterarbeitet!


Variable umbenennen

Also benennen wir die Variable **AktuelleTemperatur** in **AktuelleLichtstaerke** um!

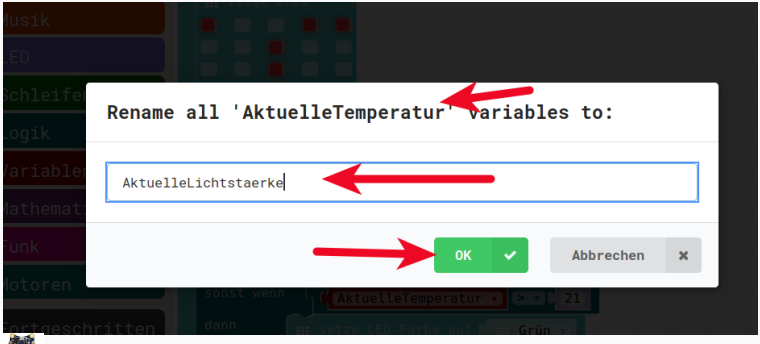
Dazu klicken wir auf das kleine Dreieck neben **AktuelleTemperatur** mitten in unserem Programm



Variable umbenennen

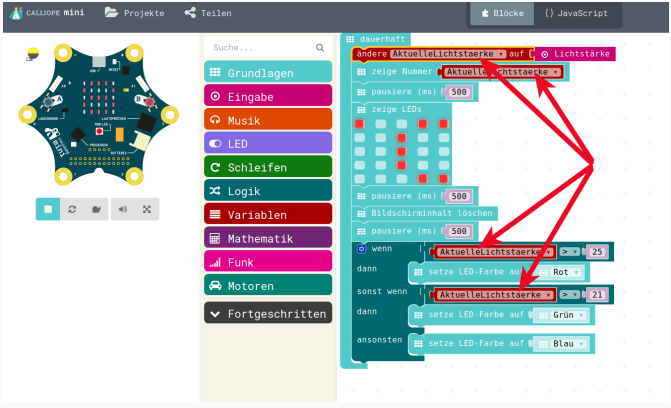
Da öffnet sich dann ein kleines Menu, in dem wir eine andere Variable benutzen können, was wir aber gar nicht wollen, sondern wir wollen wirklich die Variable **AktuelleTemperatur** umbenennen.
Dazu klicken wir also im Menu auf **Variable umbenennen**

Nun öffnet sich ein Fenster, in dem wir die Variable umbenennen können:
AktuelleLichtstaerke



Alle Verwendungen umbenannt

Hier sehen wir jetzt auch gleich den Vorteil:
Dadurch dass wir die Variable umbenannt haben, wurden **ALLE** Vorkommen dieser Variable im ganzen Programm ersetzt



Programm noch aufräumen

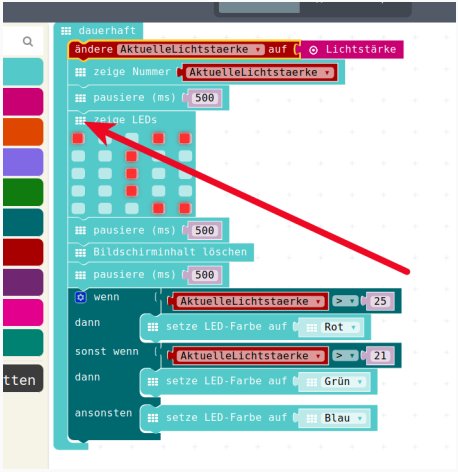
Wir bleiben weiterhin faul:

- Wir wollen später noch das **Wenn-Dann** Konstrukt benutzen.
- Momentan interessieren uns nur die Messwerte,
- die wir in der echten Welt
- mit unserem echten Calliope
- hier in userem Raum bekommen.

Darum räumen wir den ganzen Rest zur Seite.

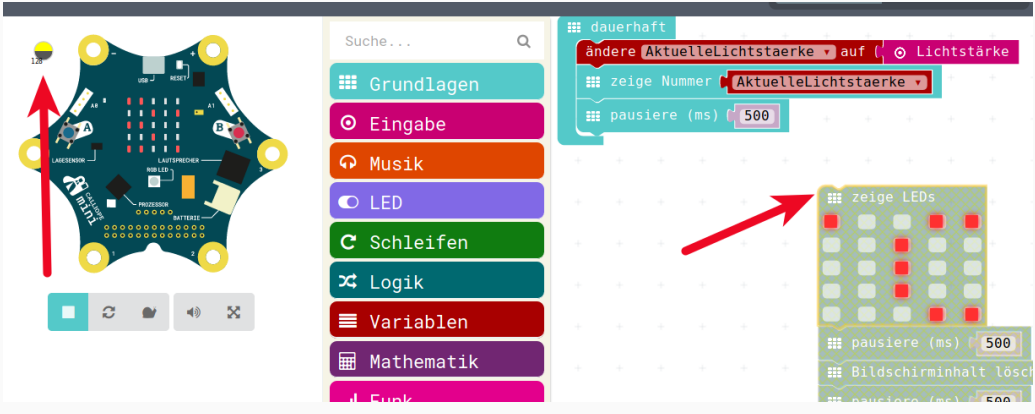

Programm noch aufräumen

ACHTUNG: Nicht löschen, nur zur Seite ziehen



Sinnvolles Programm

Sinnvolles Programm => im Simulator testen (mit der Maus auf den kleinen Licht-Regler und dort das Licht ändern)



Wenn das funktioniert, dann in den Calliope hochladen.

ACHTUNG: Bitte einen sinnvolleren Namen geben, z.B. **LichtMesser01**

JavaScript-Code

Java-Script-Code

```
let AktuelleLichtstaerke = 0
basic.forever(() => {
    AktuelleLichtstaerke = input.lightLevel()
    basic.showNumber(AktuelleLichtstaerke)
    basic.pause(500)
})
```

Download Hex-Code

Hex-code

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

05_02_SchubladenAlarm

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Die Schubladen-Alarm-Anlage

Messungen, Messwert-Ermittlung

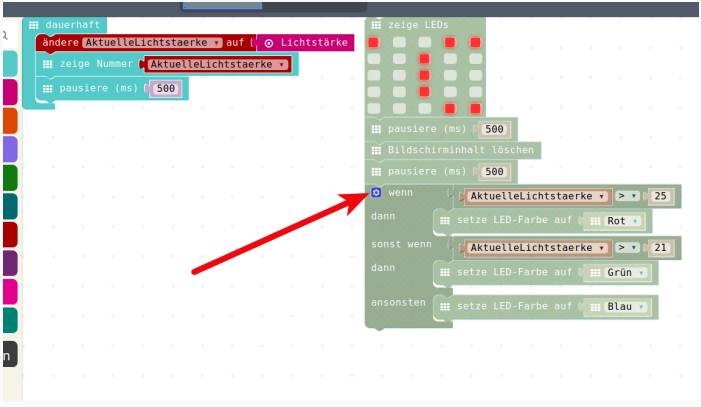
Nachdem wir das Mess-Programm in den Calliope runtergeladen haben, können wir einige Messwerte mit heller und dunkler Umgebung ermitteln.
Bei mir kamen bei den letzten Messungen in mittlerer Umgebung Messwerte um die 90 raus. Beim Abdecken des Calliope mit der Hand fallen die Werte dann sehr schnell auf ca 60 und kleiner.

Ziel unserer Mini-Alarm-Anlage

- Nun wollen wir also eine einfache Schubladen-Alarm-Anlage bauen.
- Diese soll auf der gemessenen Helligkeit basieren.
- In der geschlossenen Schublade ist es dunkel
- Das heisst niedere Messwerte
- Wenn die Schublade geöffnet wird, dann wird (ausser in einem dunklen Zimmer...) in der Schublade hell
- Das heisst hohe Messwerte
- Wir wollen also ab einem bestimmten Messwert, den Ihr Euch basierend auf den gerade gemachten Messungen selbst ausdenken könnt, etwas tun
- Wir wollen einen kleinen Alarmton auslösen

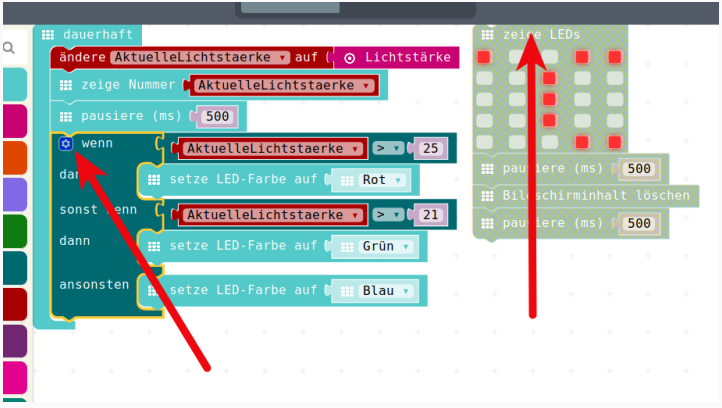

Wiedereinbau Wenn-Dann

Nun bauen wir das vorher “zur Seite geschobene” Wenn-Dann-Konstrukt wieder ein und hängen es unten in unser Programm rein.



Wiedereinbau Wenn-Dann

Der Rest (zeige LEDs und pausieren etc) ist “Müll”, das brauchen wir wirklich nicht und können es zurück ins Menu schieben, was ja gleichzeitig der Müll-Eimer ist, wenn man irgendwelche Puzzle-Teile aus dem Arbeits-Bereich rauszieht.



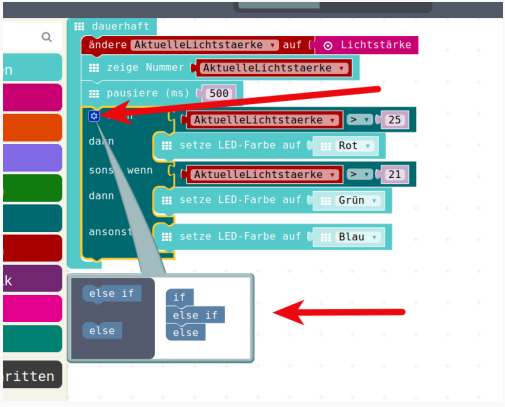
Tool/Werkzeug-Box bei Wenn-Dann

Das Wenn-Dann-Konstrukt ist noch ein Überbleibsel aus unserem Temperatur-Messer und ist so noch nicht sinnvoll einsetzbar.
Wir wollen

- gar nicht so viele Entscheidungen haben
- die Vergleiche sind wahrscheinlich noch falsch, 25 und 21 waren Temperatur-Vergleichswerte
- Wir wollen die Reaktion ändern, wir wollen keine LED-Farbe ändern, sondern einen Ton produzieren.

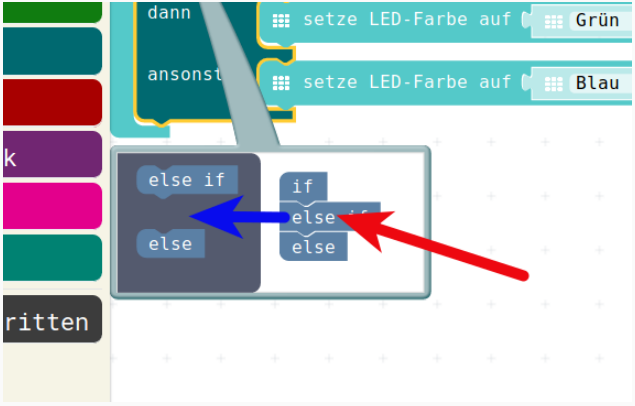
Tool/Werkzeug-Box bei Wenn-Dann

Zur Änderung des Wenn-Dann-Konstrukts öffnen wir wieder die Werkzeugbox durch das kleine Zahnrädchen.



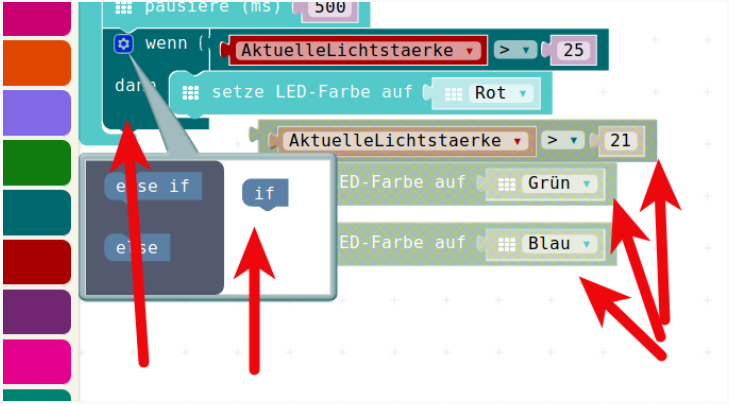
Tool/Werkzeug-Box bei Wenn-Dann

- Dort schieben wir wieder in der Miniatur-Version.
- Diesmal schieben wir von rechts nach links, das heisst wir machen unser Programm-Konstrukt viel kleiner
- Wir brauchen weder das **SonstWenn** noch das **Sonst**



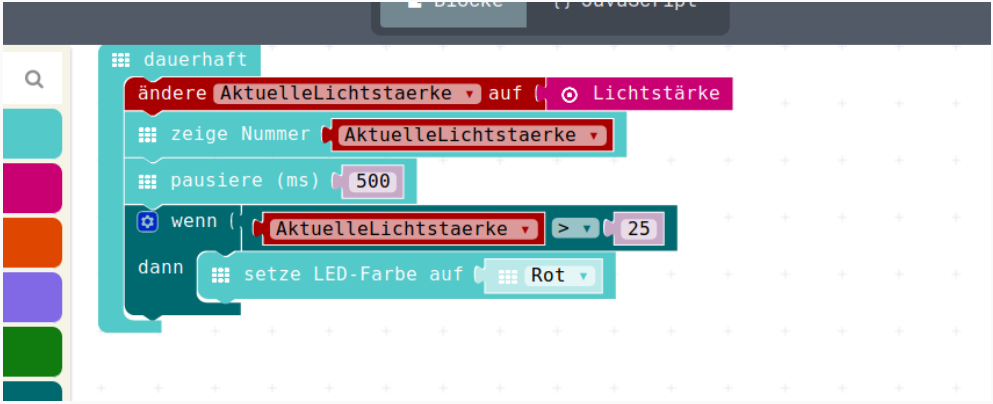
Tool/Werkzeug-Box bei Wenn-Dann

Wenn wir das alles in unserer kleinen Werkzeugbox geschoben haben, sieht unser Ergebnis in gross so aus:



Tool/Werkzeug-Box bei Wenn-Dann

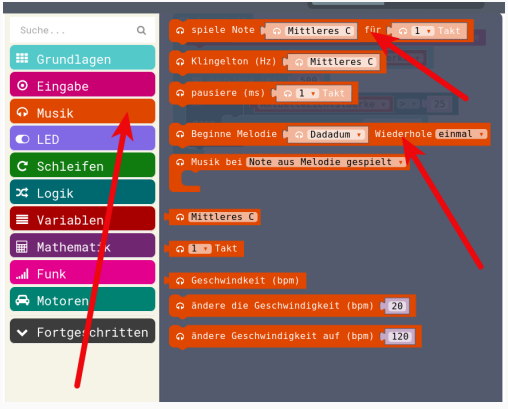
Wenn wir nun noch die jetzt überflüssig gewordenen Teile “entsorgen”, dann haben wir ein kompaktes Zwischenprogramm:



Musik/Alarmton

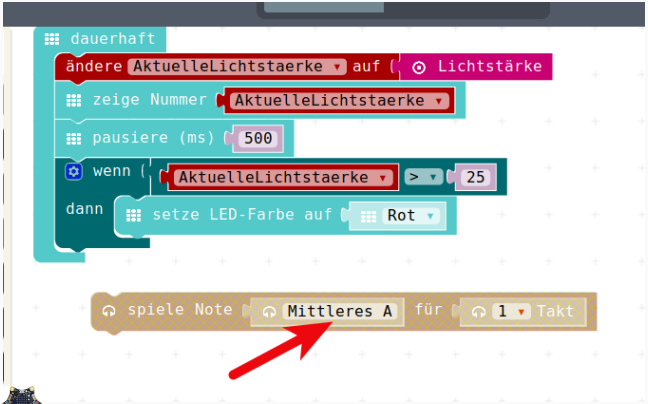
Das Menu Musik habt Ihr sicher alle schon gefunden, trotzdem hier nochmal der Hinweis:

Töne und Melodien befinden sich im Menu Musik:



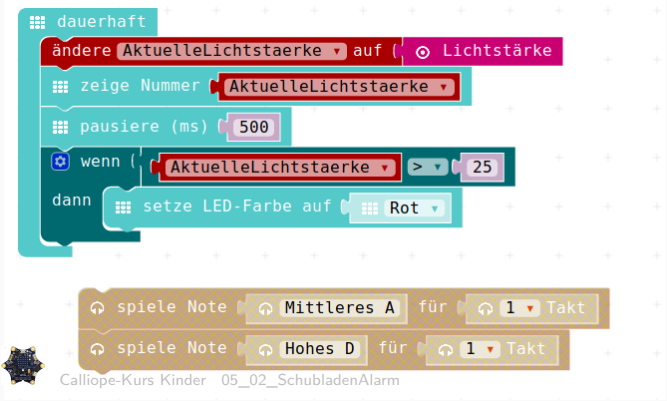
Musik/Alarmton

Wir können uns unseren Alarmton entweder aus einer Melodie direkt holen, oder aber wir bauen etwas aus Einzeltönen zusammen.
Für unsere Alarm-Anlage wollen wir das beispielhaft aus einzelnen Tönen zusammenbauen:
wir wollen ein **Martinshorn**, eine Polizeisirene nachbilden.



Musik/Alarmton

Dafür wird laut Wikipedia eine Quint empfohlen, zum Beispiel ein A und das nächsthöhere D.
Im Beispiel nehmen wir hier das mittlere A und das hohe D, aber da kann natürlich jeder für sich experimentieren.
BITTE : Macht die Lautsprecher an Euren Laptops leise, wenn jetzt alle gleichzeitig eine Sirene bauen...



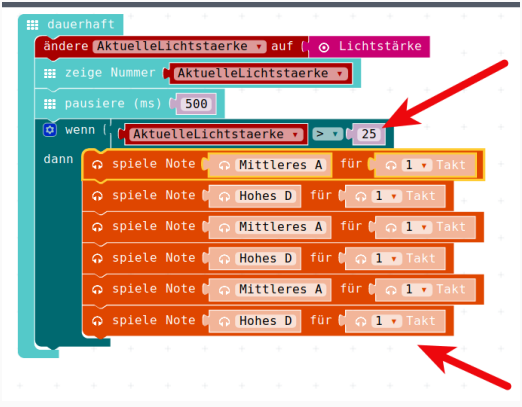
Musik/Alarmton

Wenn man die Tonfolge aus den beiden Tönen dreimal kopiert ergibt sich schon ein schöner Martinshorn-Klang:

Musik/Alarmton

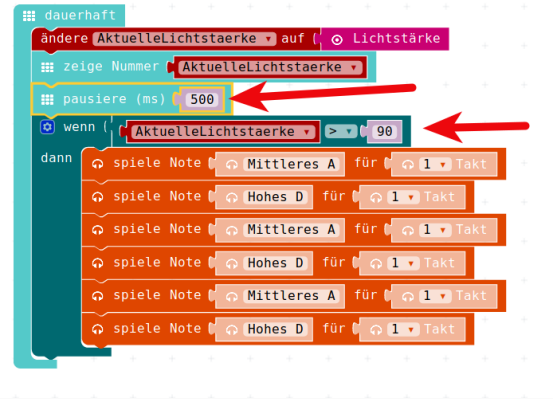
Das könne wir nun in den aktiven Bereich des Programms reinschieben und dann müssen wir noch die Lichtstärke auf einen sinnvollen Wert anpassen:

Bei meinen Messungen ergaben sich - wie gesagt - sinnvolle Werte um die 90



Musik/Alarmton

Hier ist jetzt also der Wert auf **meinen** sinnvollen Wert abgeändert.
Nun können wir noch die inzwischen unnötige Pause rausmachen:
(die brauchen wir nicht mehr, die war darin, damit das Display nicht ständig anzeigt, aber unser Programm läuft ja erst nach dem Töne machen weiter)



Das finale Programm

So sieht nun unser finales Programm aus:

Das finale Programm

Das können wir nun im Simulator ausprobieren, indem wir verschiedene Helligkeits-Stufen ausprobieren und dann wollen wir es natürlich auch wieder im echten Calliope laufen lassen.

ACHTUNG : Sinnvoellen, neuen Namen vergeben, z.B. **LichtMesser02** oder **SchubladenAlarm** oder ähnliches.

JavaScript-Code

Java-Script-Code

```
let AktuelleLichtstaerke = 0
basic.forever(() => {
    AktuelleLichtstaerke = input.lightLevel()
    basic.showNumber(AktuelleLichtstaerke)
    if (AktuelleLichtstaerke > 90) {
        music.playTone(440, music.beat(BeatFraction.Whole))
        music.playTone(587, music.beat(BeatFraction.Whole))
        music.playTone(440, music.beat(BeatFraction.Whole))
        music.playTone(587, music.beat(BeatFraction.Whole))
        music.playTone(440, music.beat(BeatFraction.Whole))
        music.playTone(587, music.beat(BeatFraction.Whole))
    }
})
```

Download Hex-Code

Hex-code


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seite gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

06_01_Auffrischen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Auffrischen

Variablen

- Variablen genauer angeschaut,
- vor allem den Unterschied zwischen Belegung einer Variablen und Benutzung einer Variablen,
- sowohl vom Verständnis als auch in unserer Programmier-Oberfläche
- für Details bitte hier schauen


LED-Ampel

- Wir haben uns auch nochmals das Vorgehen zur LED-Ampel genauer angeschaut,
- vor allem die Umwandlung von einer Ampel-Schaltung mit drei Schaltern
- in eine Ampel-Schaltung mit Calliope und Software
- für Details bitte hier schauen

Wenn-Dann

- Das alles sind sogenannte **logische** Entscheidungen.
- Dazu gibt es den Begriff des Wahrheitswertes :
  - Wahr
  - Falsch
- Weiss/Nicht oder Vielleicht, gibt es da nicht, denn dann weiss der Computer/Calliope nicht, was er tun soll
- **Logik** : **Wenn** etwas eintreffen wird, **Dann** wird etwas bestimmtes gemacht
- **Ansonsten** kann auch etwas anderes gemacht werden


Wenn-Dann

- **Wenn-Dann** kann erweitert werden:
  - **wenn dann**
  - **ansonsten wenn**
  - **ansonsten wenn**
  - **ansonsten wenn**
  - **ansonsten**
- für Details bitte hier schauen


Temperatur-Sensor

- Auslesen des Temperatur-Sensors, Speichern des aktuellen Temperatur-Wertes in einer Variablen
- Entscheidung (siehe oben) etwas zu tun, wenn eine bestimmte Temperatur überschritten wird.
- für Details bitte hier schauen


RGB-LED

- Ansteuern der RGB-LED mit einfachen Farben
- Ansteuern geht auch die einzelnen Farbanteile selbst mischen
- Dazu kann man auch die Schleifen verwenden, die lernen wir heute kennen.
- für Details bitte hier schauen, erst in der zweiten Hälfte


Hausaufgabe Licht-Sensor

- Umwandeln der Temperatur-Mess-Anlage mit LED
- in eine Licht-Mess-Anlage mit Ton-Ausgabe als Schubladen-Alarm-Anlage
- für Details bitte hier schauen

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

06_02_Schleifen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Schleifen-Programmierung

Was sind Schleifen

- **Frage** : Wofür braucht man Schleifen?
- **Antwort** : Immer dann, wenn man etwas gleiches wiederholen will!

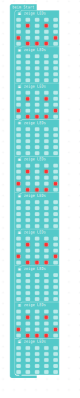
Beispiel : Man möchte beim Einschalten 5 mal ein Gesicht blinken lassen

Mit den uns bekannten Möglichkeiten:

- **beim Start**-Block holen
- Gesicht malen
- LED-Bildschirm löschen
- Gesicht malen
- LED-Bildschirm löschen
- usw usw . . . (alles in **Grundlagen**)


Beispiel 1 : Ohne Schleife

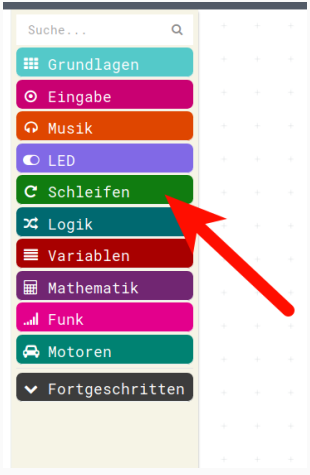
Beim Starten 5 mal ein Gesicht blinken lassen



Beispiel 1 : Mit Schleife (1)

Wo finden wir Schleifen ?

Hauptmenu:



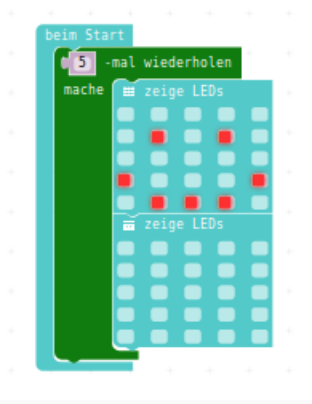
Beispiel 1 : Mit Schleife (2)

Hier finden wir verschiedene Schleifen.

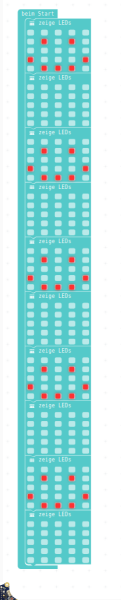
Wir interessieren uns zuerstmal für die erste Variante

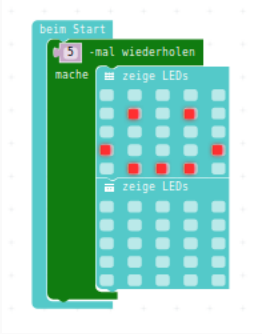
Beispiel 1 : Mit Schleife (3)

- Wir ziehen diese **4 mal wiederholen** - Schleife in den **beim Start**-Block
- dann überschreiben wir die 4 mit einer 5
- und ziehen ein Gesicht und ein Löschen in den **mache**-Teil der Schleife
- Fertig. . .



Beispiel 1 : Vergleich





Beispiel 1 : Auswertung

- Welches Programm sieht kompakter aus?
- Welches Programm ist einfacher zu verstehen?
- Welches Programm ist einfacher zu warten, zu ändern?

Damit kommen wir zu Beispiel 2:

Beispiel 2 : Beim Starten 5 mal ein Herz anzeigen

Nun kommt unser **Auftrag-Geber** von **Programm 1** und sagt:

- Ach, ich wollte doch lieber ein Herz blinken haben,
- und das wenn möglich 6 mal.

Nun wollen wir die beiden Varianten aus Beispiel 1 nehmen und entsprechend verändern.


Wieviele Änderungen müssen wir machen, wie oft mit der Maus klicken, um aus Programm 1 das Programm 2 zu machen.

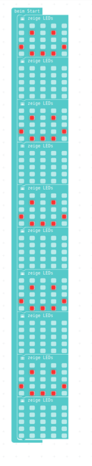


Figure 1: Von hier

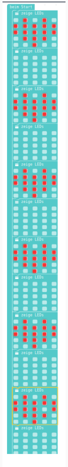


Figure 2: nach hier

Beispiel 2 : Mit Schleife

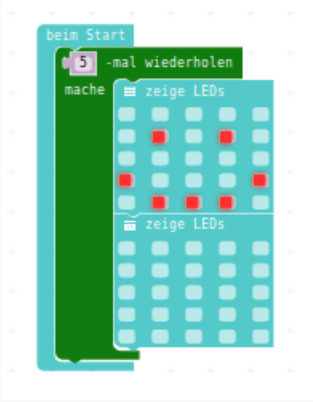


Figure 3: von hier

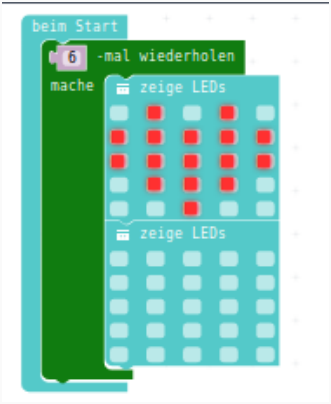


Figure 4: nach hier

Beispiel 2 : Auswertung

- Welches Programm sieht kompakter aus?
- Welches Programm ist einfacher zu verstehen?
- Welches Programm ist einfacher zu warten, zu ändern?
- Welches Programm ist fehler-anfälliger?

MERKE : Sobald man anfängt, beim Software-Programmieren etwas zu kopieren, muss man darüber nachdenken, ob man das mit einer Schleife den Computer erledigen lassen könnte.


Ursprungs-Frage

- **Frage** : Wofür braucht man Schleifen?
- **Antwort 2** : Immer dann, wenn man etwas sehr ähnliches wiederholen will, wobei sich dabei bestimmte Dinge ändern können, die vom Schleifendurchlauf abhängen.
  - Also beim **ersten** Schleifendurchlauf wird etwas mit einer **1** gemacht
  - Beim **zweiten** Durchlauf wird etwas mit einer **2** gemacht
  - usw. usw.
- Wir wollen nun einen Zähler bauen.


Beispiel 3 : Schleife mit Zähler

Nun wollen wir innerhalb des sogenannten “Schleifenkörpers” die Anzahl der Schleifen-Durchgänge anzeigen.

- Dazu benutzen wir die gerade vorhandene Schleife,
- legen **VOR** der Schleife eine Variable namens **SchleifenZaehler** an,
- diese belegen wir mit 0.


Beispiel 3 : Schleife mit Zähler

Im Schleifenkörper lassen wir uns den Wert dieser Variable anzeigen (mit “Zeige Nummer”) und erhöhen anschliessend die Variable/den Zähler.

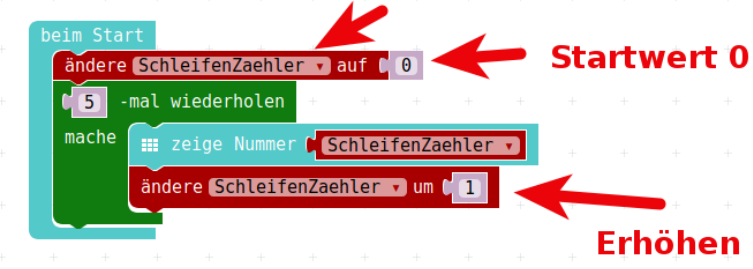


Figure 5: Schleife mit manuellem Zähler

Da wir den Zähler mit 0 vorbelegen und die Schleife 5 mal läuft, bekommen wir durch dieses Programm die Zahlen 0 bis 4 angezeigt.

Beispiel 4 : Schleife mit eingebautem Zähler

Diese Art der Schleife wird sehr oft gebraucht:
eine Schleife, die eine bestimmte Anzahl von Durchläufen erlaubt und bei der man die Schleifendurchläufe mitzählt.

Darum gibt es dafür ein extra Programmier-Konstrukt.

Das ist die sogenannte **Index-For-Schleife** .


Beispiel 4 : Schleife mit eingebautem Zähler

Die Index-For-Schleife finden wir ebenso im Menu Schleifen:

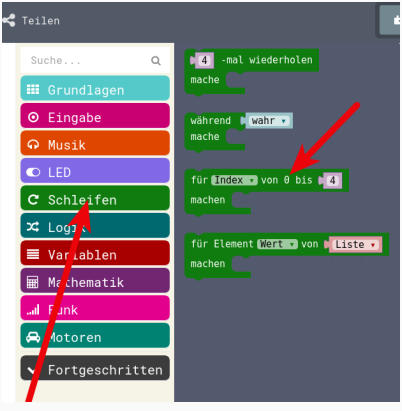


Figure 6: Schleifen-Menu

Beispiel 4 : Schleife mit eingebautem Zähler

Wenn wir diese Schleife benutzen und unser Programm entsprechend umgestalten, sieht es nochmal um einiges einfacher aus:

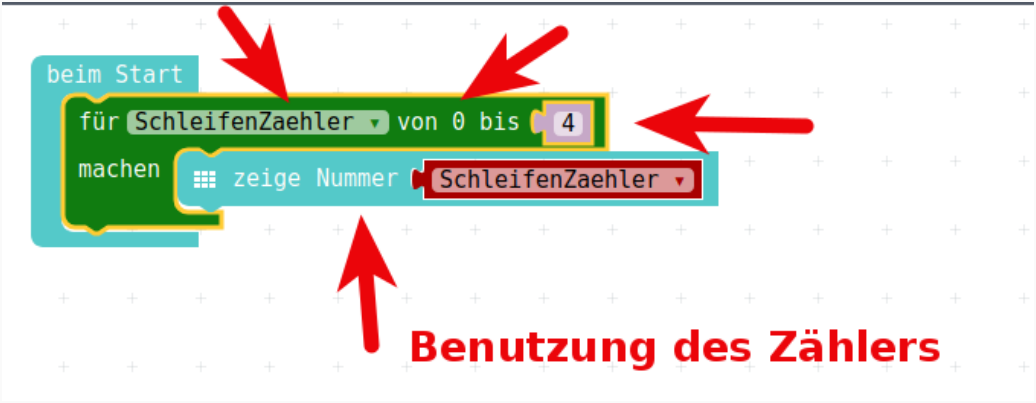
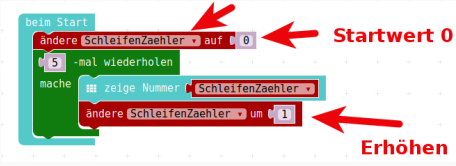
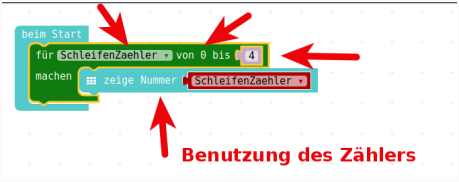


Figure 7: Schleife mit IndexZähler

Vergleich der beiden Schleifen





Lizenz/Copyright-Info


06_03_Motoren

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Elektro-Motoren

Verschiedene Motoren

Es gibt sehr viele verschiedene Motoren:

- Dieselmotoren
- Benzin-Motoren
- Düsen-Antriebs-Motoren
- ...

Um die alle wollen wir uns heute sicher **NICHT** kümmern...

Elektro Motoren

Sondern heute wollen wir - wie oben geschrieben - Elektro-Motoren anschauen. Aber auch bei den Elektro-Motoren gibt es sehr grosse Unterschiede, sowohl was deren elektrische Spannung als auch was deren Funktionsweise anbelangt.

Elektro Motoren

Folgende Arten von Elektro-Motoren fallen mir auf Anhieb ein:

- DC-Motoren
- Schritt-Motoren
- Servo-Motoren
- DC-Motoren mit Getriebe
- und noch mehr
- ...

Grundprinzip

Grundprinzip ist im Allgemeinen immer ein Elektro-Magnet, der einen Permanent-Magneten anzieht und in dem Moment, wenn der Permanent-Magnet “angekommen” ist, wird der Elektro-Magnet ausgeschaltet und dann der nächste Elektro-Magnet ein Stück weiter in der Drehung eingeschaltet.

(Achtung: Das ist nur das Prinzip. Im Detail sieht das meist etwas anders aus, aber so kann man es sich am ehesten bildlich vortsellen)


Grundprinzip

So kann man sich das vorstellen:

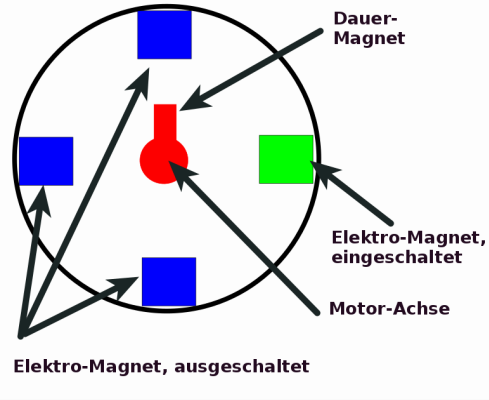


Figure 1: Prinzip-Bild Motor

Grundprinzip

- Der eingeschaltete Elektro-Magnet zieht den Dauer-Magneten an
- Der Dauer-Magnet bewegt sich in Richtung Elektro-Magnet
- Die Achse dreht sich mit
- Wenn der Dauer-Magnet den Elektro-Magnet erreicht hat, schaltet sich der Elektro-Magnet ab
- Der Elektro-Magnet eine viertels Umdrehung (90°) weiter schaltet sich ein
- Er zieht den Dauer-Magneten an
- usw. . . . usw.


Animierter Motor Langsam

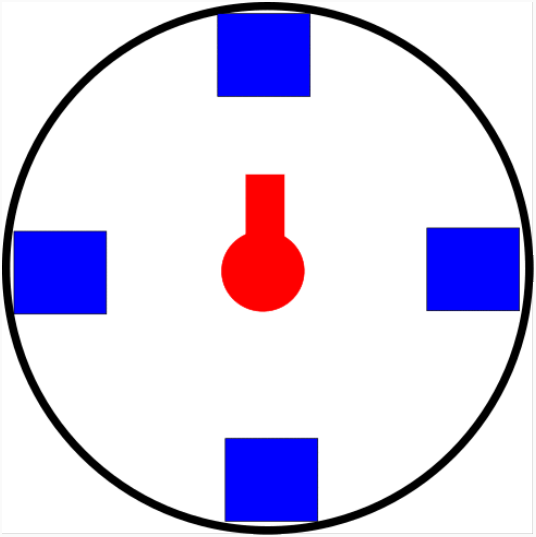


Figure 2: Animierter Motor langsam

Animierter Motor Schnell

Und wenn das ganze etwas schneller abläuft, dann sieht das so aus:

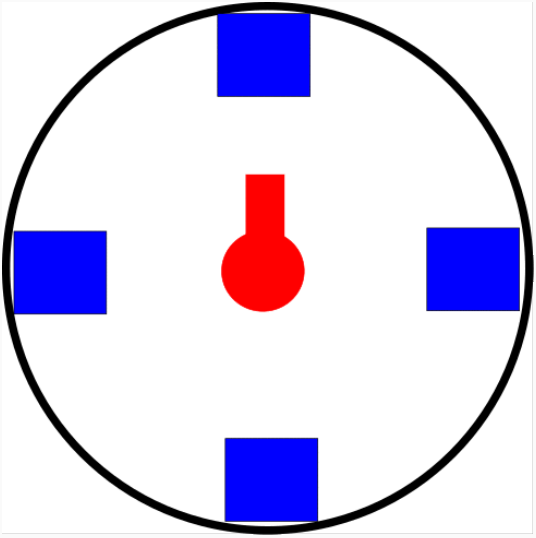
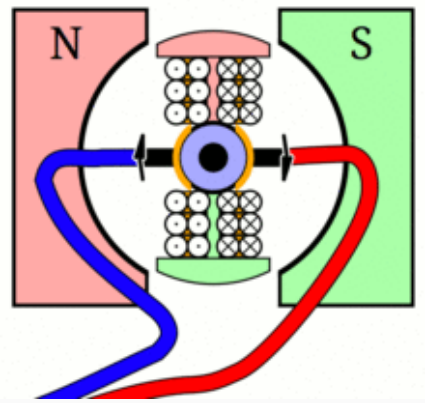


Figure 3: Animierter Motor schnell

Realistischere Animation

Wie geschrieben, das zeigt nur das Prinzip.



Dieses Model kommt der Wirklichkeit näher:

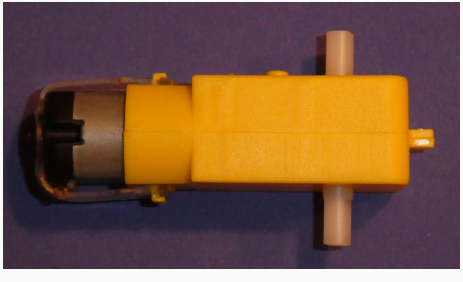
(https://de.wikipedia.org/wiki/Elektromotor , MichaelFrey, CC BY-SA 3.0)

DC-Motoren

DC-Motoren benutzen dieses Prinzip um einen Motor einfach ständig komplett um seine Achse drehen zu lassen. Dabei wird durch einen sehr einfachen Mechanismus dafür gesorgt, dass die jeweiligen Magneten ständig weiter-geschaltet werden. Im Allgemeinen kann man (im Bereich wie es für diesen Motor festgelegt ist) sagen: Höhere Spannung => Schnellere Drehung. Wie weit sich der Motor dreht, kann normalerweise nicht haargenau vorherbestimmt werden.

Robotik-Motor

Das hier ist ein “Standard”-Robotik-Motor mit eingebautem Getriebe, der Motor selbst ist ein DC-Motor.



Lüfter-Motor

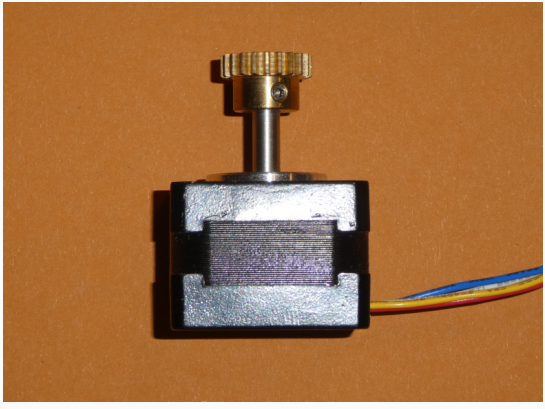
Das hier ist ein Lüfter-Motor, wie er im Computerbau verwendet wird, auch das ist ein normaler DC-Motor.

Schritt-Motoren

Bei **Schritt-Motoren** hat man das normalerweise selbst haargenau im Griff, man kann entweder durch Elektronik oder durch genaue Programmierung genau einzelne Schritte bestimmen. Da wird zum Beispiel ein Schritt-Motor mit 36 Schritten pro Umdrehung hergestellt, d.h. jeder Schritt dreht den Motor um 10° weiter und nach 36 Schritten hat er genau eine komplette Umdrehung gemacht. Die einzelnen Schritte müssen gesteuert werden.

Schritt-Motoren

Das hier ist ein Schritt-Motor, er hat vier Anschlüsse:



Wenn man den Schritt-Motor von Hand dreht, kann man die einzelnen Schritte auch spüren.

Servo-Motoren

Servo-Motoren hingegen haben eine ganz andere Funktion:
Sie können sich normalerweise nur ca eine halbe Umdrehung umdrehen. Sie werden üblicherweise mit Gradzahlen angesteuert.
So kann man einen Servomotor von 0 - 180 Grad (Eine volle Umdreheung sind immer 360 Grad) ansteuern, normalerweise ist die Null-Stellung bei 90 Grad, man kann also einen Servomotor um eine viertels Drehung nach links auf 0 und um eine viertels Drehung nach rechts auf 180 Grad bewegen.
Im Modellbau werden Servos sehr oft benutzt, z.B. um zu lenken, um ein Höhenruder beim Flugzeug zu verstellen.
Unser Calliope-Männchen hier in der Turbine nutzt einen Servo-Motor um zu winken.

Servo-Motoren

Das hier ist ein Servo-Motor, er hat drei Anschlüsse,

Figure 4: 05_ServoMotor

- einmal Plus,
- einmal Minus
- und einmal den Ansteuer-Anschluss

Was wir uns anschauen

Wir schauen uns heute die Ansteuerung von normalen DC-Motoren (mit Getriebe) an. Und beim nächsten Mal schauen wir uns evt. die Ansteuerung von Servo-Motoren (Servos) an.

Lizenz/Copyright-Info

06_04_DC_Motoren

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

DC-Motoren

Einführung

DC steht für **D** irect **C** urrent und heisst Gleichstrom. (AC steht für **A** lternate **C** urrent und heisst Wechselstrom. Die australische Musikgruppe AC/DC heisst also eigentlich Wechselstrom/Gleichstrom)

Ein **DC**-Motor funktioniert prinzipiell wie gerade beschrieben :

- Ein elektrischer Magnet zieht einen Permanent-Magnet an.
- Sobald der Permanent-Magnet in die Nähe des elektrischen Magneten kommt, schaltet dieser ab und der etwas weiter entfernte Elektro-Magnet wird eingeschaltet.
- So geht es in einem fort, immer im Kreis herum.

Der DC-Motor läuft einfach mit einer Batterie, je grösser die Spannung um so schneller dreht sich der Motor.
Er darf aber nicht mehr Spannung bekommen, als er verträgt!

Motor ausprobieren

Nun probieren wir einfach mal direkt den Motor an eine Batterie anzuschliessen.

So sieht das dann als Schaltbild aus:

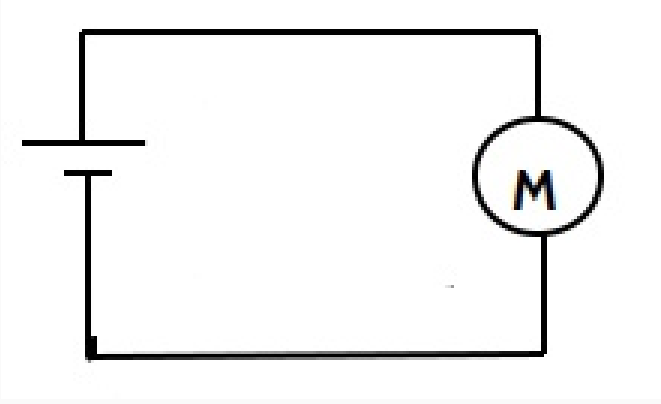
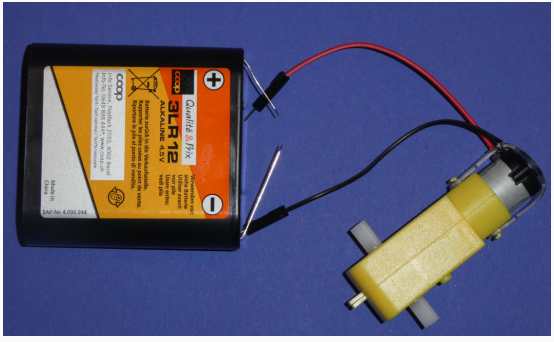


Figure 1: Motor-Schaltbild

Batterie ranhalten

Wir halten hier die Kabel nur an die Batterie ran, wir wollen nur wissen ob das so funktioniert. **Achtung** : Wie wir am zweiten Nachmittag gelernt haben, ist elektrischer Strom gefährlich. Wir haben aber auch gelernt, dass normalerweise eine Batterie mit 4.5 V uns nicht gefährlich werden kann. Darum können wir das einfach zusammenhalten!



Motor-Anschluss an den Calliope

Der Calliope hat eine zusätzliche elektrische Schaltung mit auf der Platine, die den Anschluss eines Motors überhaupt erlaubt.
Ein Motor braucht so viel Leistung, dass man den normalerweise nicht direkt an den Mikro-Prozessoren (das sind die "Gehirne" bei unseren Calliopes und anderen Bastel-Platinen) anschliessen darf.
Der Calliope hat aber die notwendigen "Motor-Treiber" gleich mit eingebaut. Die Schaltung erlaubt sogar den zusätzlichen Anschluss einer stärkeren Batterie (Achtung: bis zu **9V**) so dass man mit dem Calliope sogar Motoren mit bis zu 9V anschliessen und ansteuern kann.

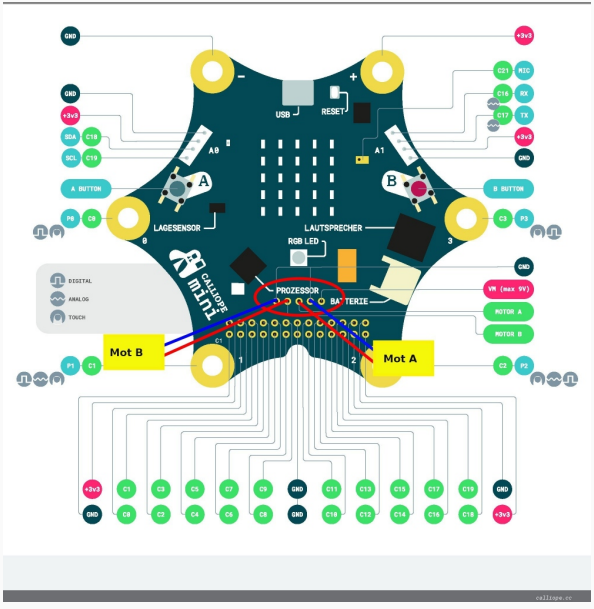
Ein oder Zwei Motoren

Je nach Verwendungszweck kann man an den Calliope entweder

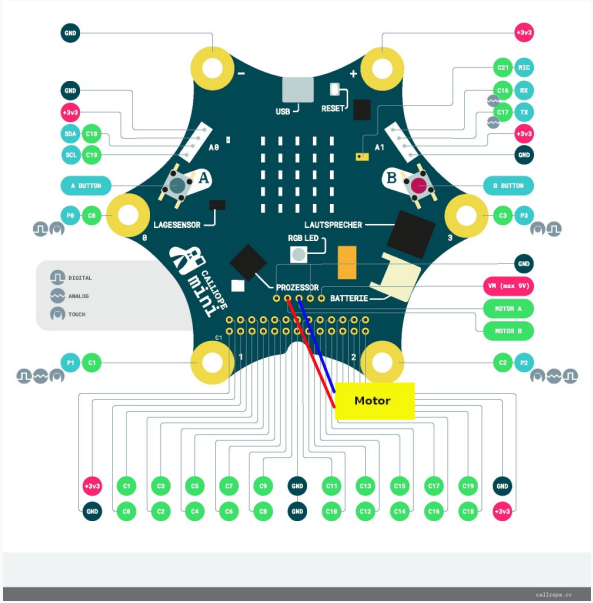
- 1 Motor anschliessen, der kann dann vorwärts und rückwärts drehen
- 2 Motoren anschliessen, die können dann nur einzeln vorwärts drehen

So sehen die beiden Möglichkeiten zum Anschluss von zwei oder einem Motor aus:

Zwei Motoren



Ein Motor


GND
+3v3
USB
RESET
C21
MIC
C16
RX
C17
TX
SDA
C18
SCL
C19
A0
A1
A
B
A BUTTON
B BUTTON
P0
C0
C3
P3
LAGESENSOR
LAUTSPRECHER
RGB LED
DIGITAL
ANALOG
TOUCH
PROZESSOR
BATTERIE
VM (max 9V)
MOTOR A
MOTOR B
P1
C1
C2
P2
Motor
C9
C11
C13
C15
C10
C12
C14
C16
C18

Pfostenstecker

Um dies zu ermöglichen, müssen wir nun zuerst einmal Pfostenstecker oder Pfostenbuchsen an den Calliope löten.

So sieht das aus:

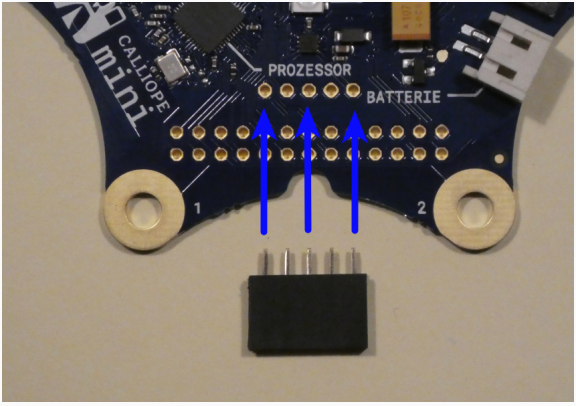


Figure 2: Pfostenstecker

Pfostenstecker aufgelötet

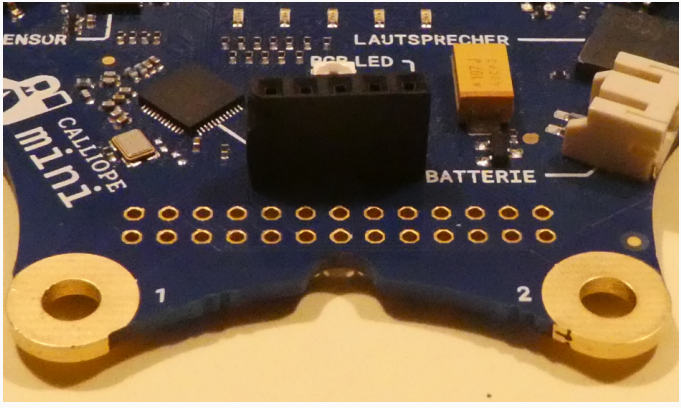


Figure 3: Pfostenstecker aufgelötet

Motor-Programmierung für Calliope

Nachdem wir nun wissen, wie wir einen einzelnen Motor an den Calliope elektrisch anschliessen, wollen wir den Ausgang für den Motor auch mit Software programmieren.

Im ersten Schritt wollen wir nur ganz einfach den Motor ein- und ausschalten können. Dazu wollen wir mit dem linken Knopf ein und mit dem rechten Knopf ausschalten.

Motor-Menu

Die Motor-Ansteuerung findet sich im Menu Motoren:

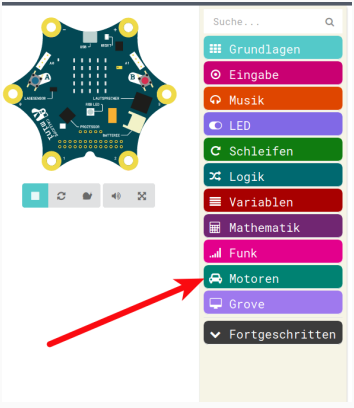


Figure 4: Menu Motor

Motor Befehle

Es gibt nicht viele Befehle zum Steuern von Motoren:

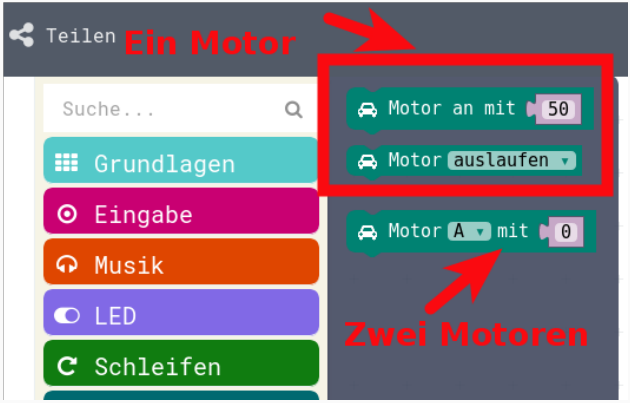


Figure 5: Motor Befehle

Verfügbare Befehle

Wie oben beschrieben, gibt es die Möglichkeit entweder zwei Motoren anszuschliessen, bei zwei Motoren kann man dann

- Den Motor A oder B auswählen
- Die Geschwindigkeit von 0 - 100 angeben.

Bei Anschluss von nur einem Motor kann man

- Eine Geschwindigkeit von -100 bis +100 eingeben
- Bremsen oder auf Leerlauf schalten

Knopfdruck einschalten

Wir wollen nun - wie beschrieben - nur den einen Motor entweder mit 100 einschalten oder beim Druck des rechten Knopfes mit 0 einschalten, was so viel wie Ausschalten heisst. . .

Mit den Befehlen zum Auswerten der Knöpfe :

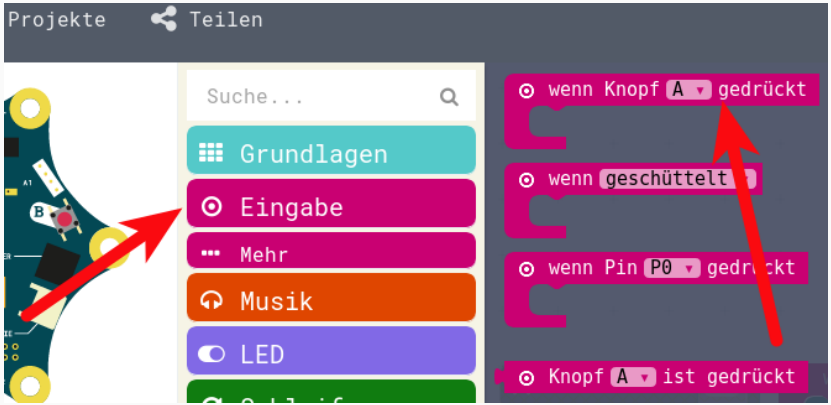


Figure 6: Knopf Befehle

Motor-Ansteuerung

Und den Befehlen zum Ansteuern des Motors :

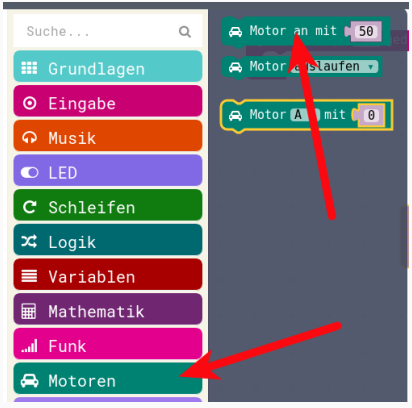


Figure 7: Motor Befehle

Erstes Motor-Programm

Ergibt das unser erstes Motor-Programm

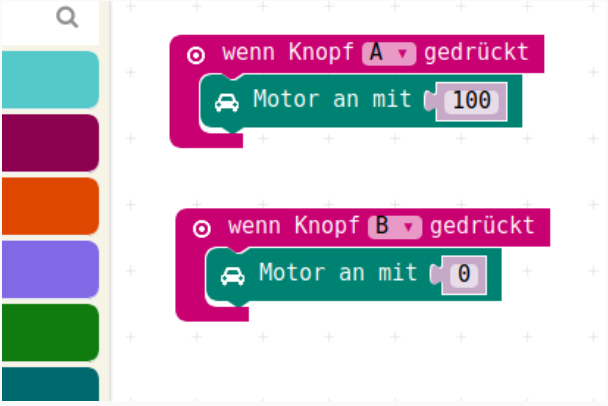


Figure 8: Motor Programm 1

Diese Programm können wir leider im Simulator gar nicht nutzen.

Dazu müssen wir das Programm auf den Calliope runterladen und dort ausprobieren.

JavaScript-Code

Java-Script-Code

```
input.onButtonPressed(Button.A, () => {
    motors.motorPower(100)
})
input.onButtonPressed(Button.B, () => {
    motors.motorPower(0)
})
```

Download Hex-Code

Hex-code

Motor rückwärts

Da wir nun einen Motor angeschlossen haben, können wir nun auch versuchen, den Motor in die andere Richtung laufen zu lassen.

Dazu brauchen wir einen dritten Informations-Eingang, neben Knopf A und Knopf B.

Wir können ja zum Beispiel das Ereignis Knopf A **UND** Knopf B gleichzeitig gedrückt auswerten.

Dazu bauen wir das Programm um:

- Knopf **A** => vorwärts
- Knopf **B** => rückwärts
- Knopf **A UND B** => Stop

Mit Motor rückwärts

Wenn wir nun unser Programm im Arbeits-Bereich entsprechend abändern, sieht das nun so aus:

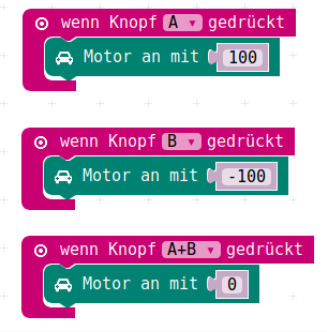


Figure 9: Motor Programm Vor / Rück

Auch dieses Programm können wir leider im Simulator nicht nutzen und müssen wir das Programm auf den Calliope runterladen und dort ausprobieren.

JavaScript-Code

Java-Script-Code

```
input.onButtonPressed(Button.A, () => {
    motors.motorPower(100)
})
input.onButtonPressed(Button.B, () => {
    motors.motorPower(-100)
})
input.onButtonPressed(Button.AB, () => {
    motors.motorPower(0)
})
```

Download Hex-Code

Hex-code


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

06_05_LageSensor

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Motorsteuerung mit Lage-Sensor

Der Lagesensor

Das funktioniert ja schon mal ganz gut.
Nun möchten wir mit diesem einfachen Motor-Steuerungs-Programm auch noch eine andere Eingangs-Möglichkeit ausprobieren:

Den Lage-Sensor!

Der Calliope hat einen Lage-Sensor eingebaut, der in allen Raumrichtungen funktioniert.

Also :

- Oben / Unten
- Links / Rechts
- Vorne / Hinten

Abfragen des Lagesensors

Die Abfragen, um den Lage-Sensor genau auszuwerten, sind recht kompliziert. Man muss Koordinaten-Systeme verstehen und man sollte Winkelrechnung verstehen. Beides ist in Euerem Alter wahrscheinlich noch nicht der Fall.

Zusätzlich zu den genauen Abfrage-Möglichkeiten, die schwierig zu verwenden sind, hat der Calliope aber auch die Möglichkeit, sehr einfach den Lage-Sensor abzufragen.

Das wollen wir nun tun:

- Beim Gerade halten des Calliope soll der Motor aus sein.
- Beim Kippen nach links soll er sich nach vorne drehen
- Beim Kippen nach rechts soll er sich nach hinten drehen.

Eingabe-Menu

Sowohl die genauen, schwierigeren Befehle als auch die Einfachen befinden sich im Menu Eingabe:

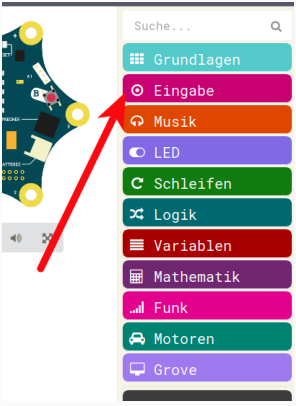


Figure 1: Menu Eingabe

Inhalte Eingabe-Menu

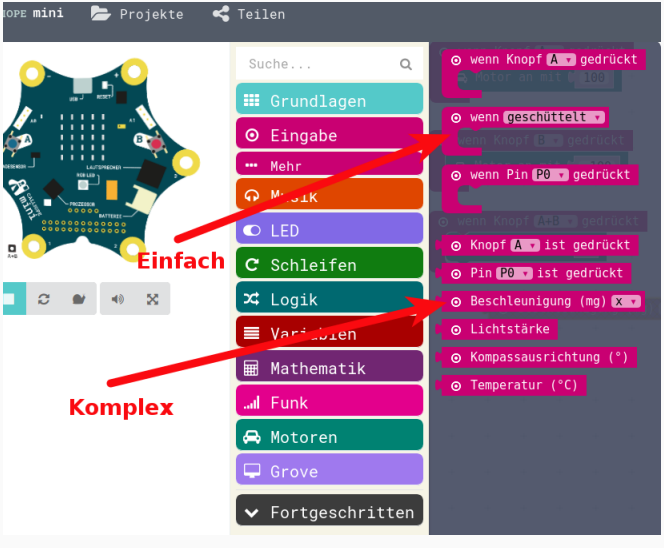


Figure 2: Menu Eingabe Inhalt

Wenn geschüttelt

Nun ziehen wir drei mal das **wenn geschüttelt** in unseren Arbeits-Bereich:

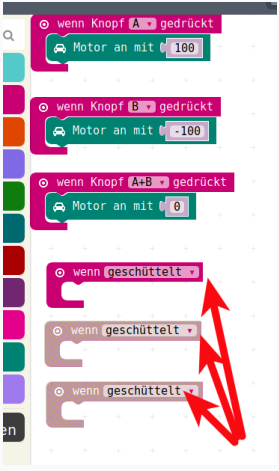


Figure 3: Dreimal Geschuettelt

Umwandeln

Diese wandeln wir nun alle durch Druck auf das Dreieick:

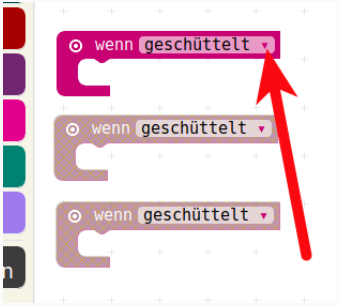


Figure 4: Dreieck

um in drei verschiedene Reaktionen:

- “nach links neigen”
- “Display nach oben”
- “nach rechts neigen”

Möglichkeiten

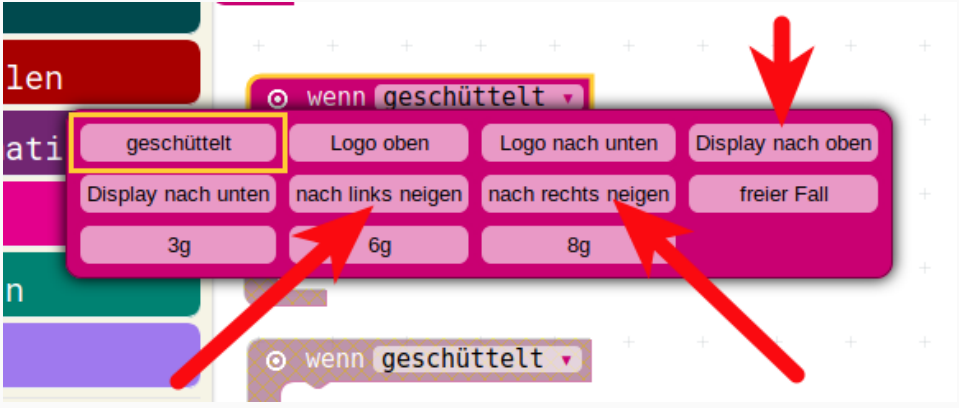


Figure 5: Lage-Sensor

Unser Programm

Damit sieht unser Programm nun so aus:

Figure 6: Lage-Sensor drin

Fertiges Programm

und wenn wir nun die entsprechenden Befehle von oben nach unten schieben, dann können wir unseren Motor durch kippen steuern.

Figure 7: Lage-Sensor benutzt

Runterladen in Calliope

Dieses Programm können wir nun auch in den Calliope laden.

Achtung: Zumindest bei manchen Kombinationen von Calliope und Computer (und vermutlich angeschlossenem Computer-Ladegerät) hat der Lage-Sensor **NICHT** richtig funktioniert.

==>

Bitte steckt in diesem Fall das USB - Kabel aus und betreibt Euren Calliope nur über Batterie.

JavaScript-Code

Java-Script-Code

```
input.onGesture(Gesture.TiltLeft, () => {
    motors.motorPower(100)
})
input.onGesture(Gesture.TiltRight, () => {
    motors.motorPower(-100)
})
input.onGesture(Gesture.ScreenUp, () => {
    motors.motorPower(0)
})
```

Download Hex-Code

Hex-code

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

07_01_Nachlese

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Nachmittag 4, “Hausaufgaben”

Aufgabe 1: RGB-LED ansteuern

RGB-Led mit Schleife 1

Die RGB-LED kann nicht nur die vorgefertigten Farben anzeigen, sondern sie kann die Farben beliebig mischen.

RGB steht für **R** ot **G** elb **B** lau, mit diesen 3 Grundfarben kann man jede Farbe zusammenmischen.

Die Anteile von Rot Gelb und Blau lassen sich von 0 bis 255 verändern.

Da kann man eine Schleife nutzen und verschiedene Farben produzieren.

RGB-Led mit Schleife 2

Spielt doch mal etwas damit rum, im Menu findet sich das hier:

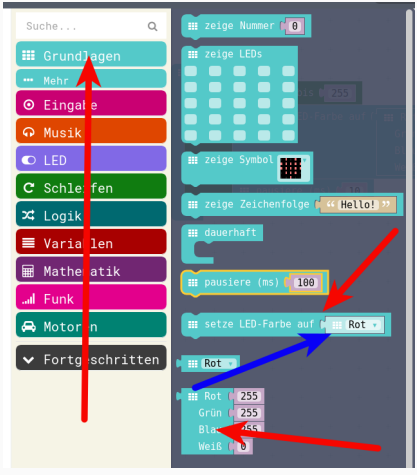


Figure 1: Menu RGB-Led

RGB-Led mit Schleife 3

So kann man zum Beispiel damit verschiedene Rot-Töne machen:

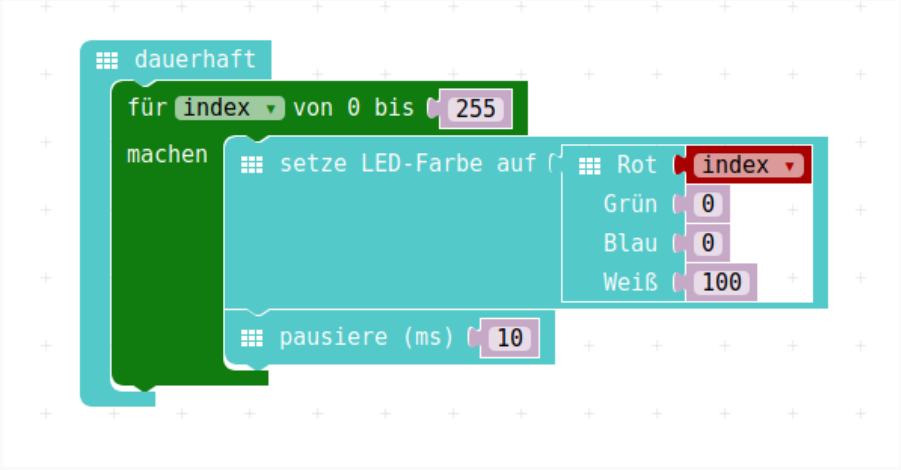


Figure 2: Rot-Töne

RGB-Led mit Schleife 4

Eure Aufgabe: **Spielt** doch damit mal rum,

- findet schöne Farbverläufe,
- macht mehrere Schleifen,
- was passiert wenn Ihr eine Schleife **innerhalb** einer anderen Schleife macht?

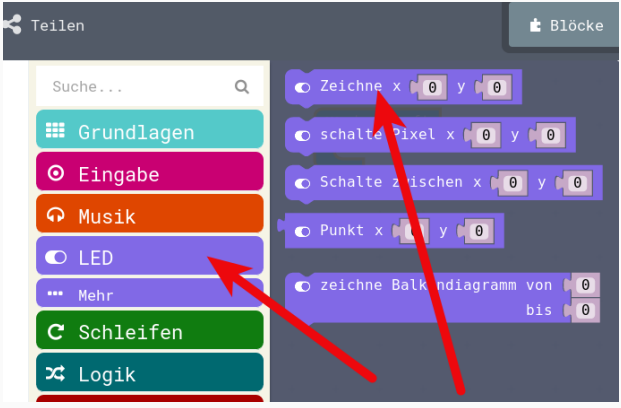
Aufgabe 2: LED-Display ansteuern

Einzelne LEDs ansteuern 1

Nun schauen wir uns aus dem Menu **LED** einmal einen Befehl etwas genauer an:

Zeichne X Wert Y Wert

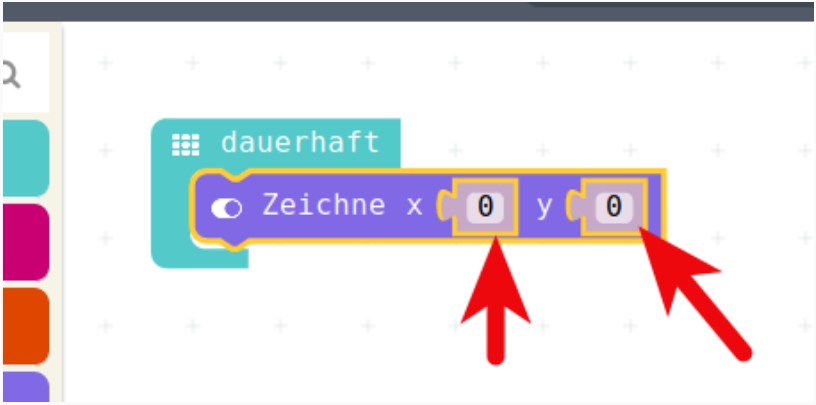
Damit kann man einzelne LEDs auf unserem 5 x 5 LED-Bildschirm einschalten.



Einzelne LEDs ansteuern 2

- Das probieren wir gleich mal im Simulator aus,
- Wir klicken diesen einen Befehl in die **Dauerhaft**-Schleife ein
- Wir spielen mit den X und Y-Werten
- Wir schauen, was im Simulator passiert.

Einzelne LEDs ansteuern 3



- Wir können durch Verändern der Werte für X und Y zwischen 0 und 4 jede beliebige LED auf unserem 5 x 5 - Display einschalten.

Aufgabe

Aufgabe 1:

- Ersetzt eine der beiden Zahlen durch eine Variable
- Baut den Setzen-Befehle in eine index-For-Schleife ein
- Könnt Ihr einzelne Zeilen oder Spalten LED für LED einschalten ?
- Damit man was sieht : **pausieren** nicht vergessen

Aufgabe 2:

- Ersetzt die andere der beiden Zahlen auch noch durch eine Variable
- Baut eine zweite Schleife um die andere Variable zu ändern
- Spielt mit den Schleifen rum
- Könnt Ihr eine Schleife in einer Schleife programmieren ?
- Schafft Ihr es, den ganzen 5x5 - LED-Bildschirm mit diesen Befehlen und Schleifen einzuschalten?

Viel Spass beim Tüfteln !


Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

08_01_Auffrischen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Auffrischen / Hausaufgabe

Auffrischen Schleifen

Schleifen-Programmierung

- Frage : Wofür braucht man Schleifen?
- Antwort 1 : Immer dann, wenn man etwas gleiches wiederholen will!
- Antwort 2 : Immer dann, wenn man etwas sehr ähnliches wiederholen will, wobei sich dabei bestimmte Dinge ändern können, die vom Schleifendurchlauf abhängen.
  - Also beim **ersten** Schleifendurchlauf wird etwas mit einer **1** gemacht
  - Beim **zweiten** Durchlauf wird etwas mit einer **2** gemacht
  - usw. usw.


Beispiel 1 : Ohne Schleife

Beim Starten 5 mal ein Gesicht blinken lassen

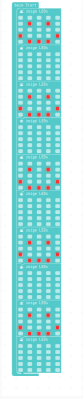


Figure 1: Ohne Schleife

Beispiel 1 : Mit Schleife

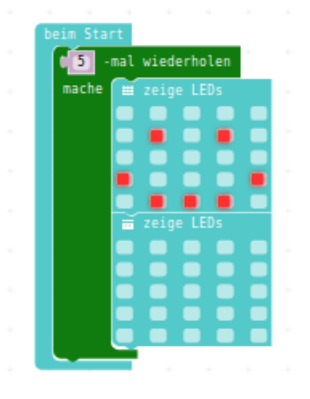
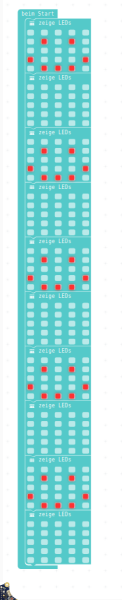
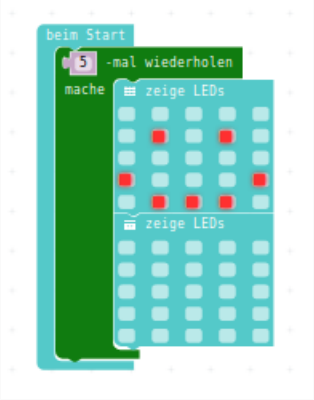


Figure 2: Mit Schleife

Beispiel 1 : Vergleich





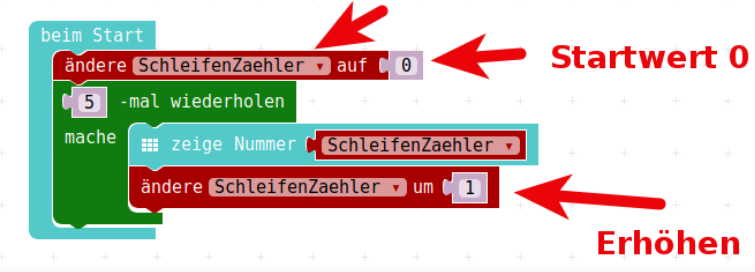
Beispiel 2 : Schleife mit Zähler

Nun wollen wir innerhalb des sogenannten “Schleifenkörpers” die Anzahl der Schleifen-Durchgänge anzeigen.

- Dazu benutzen wir die gerade vorhandene Schleife,
- legen **VOR** der Schleife eine Variable namens **SchleifenZaehler** an,
- diese belegen wir mit 0.


Beispiel 3 : Schleife mit Zähler

Im Schleifenkörper lassen wir uns den Wert dieser Variable anzeigen (mit “Zeige Nummer”) und erhöhen anschliessend die Variable/den Zähler.



Da wir den Zähler mit 0 vorbelegen und die Schleife 5 mal läuft, bekommen wir durch dieses Programm die Zahlen 0 bis 4 angezeigt.

Beispiel 3 : Schleife mit eingebautem Zähler

Diese Art der Schleife wird sehr oft gebraucht:
eine Schleife, die eine bestimmte Anzahl von Durchläufen erlaubt und bei der man die Schleifendurchläufe mitzählt.

Darum gibt es dafür ein extra Programmier-Konstrukt.

Das ist die Index-For-Schleife, die wir am letzten Nachmittag schon kennengelernt haben.


Beispiel 4 : Schleife mit eingebautem Zähler

Diese finden wir ebenso im Menu Schleifen:

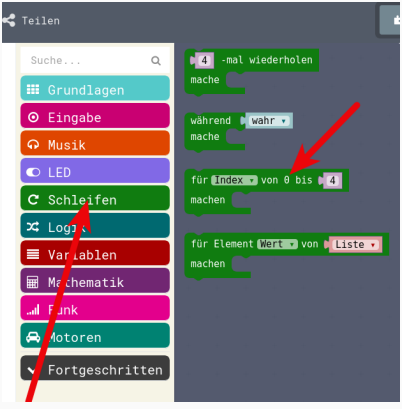
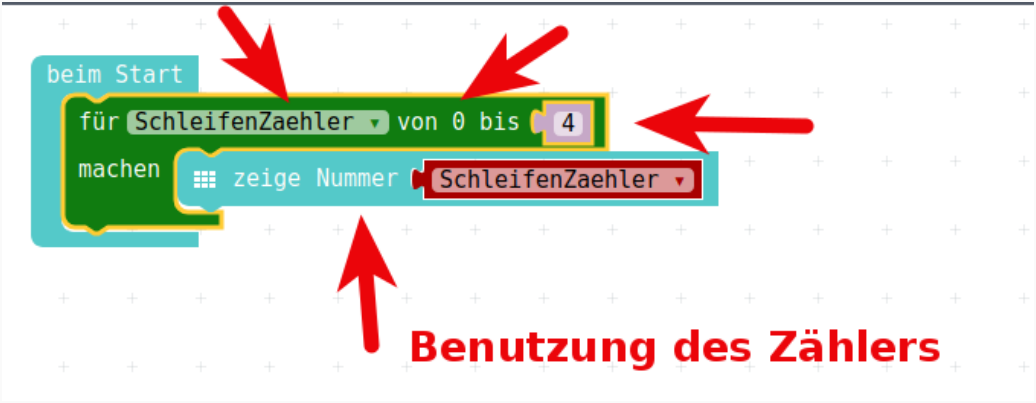


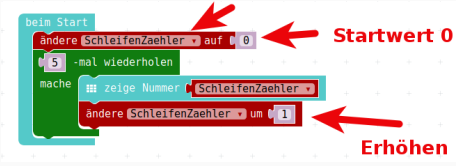
Figure 3: Schleifen-Menu

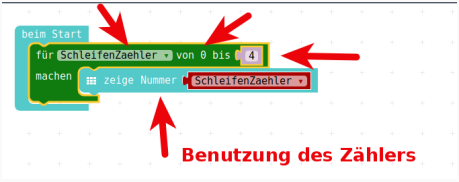
Beispiel 4 : Schleife mit eingebautem Zähler

Wenn wir diese Schleife benutzen und unser Programm entsprechend umgestalten, sieht es nochmal um einiges einfacher aus:



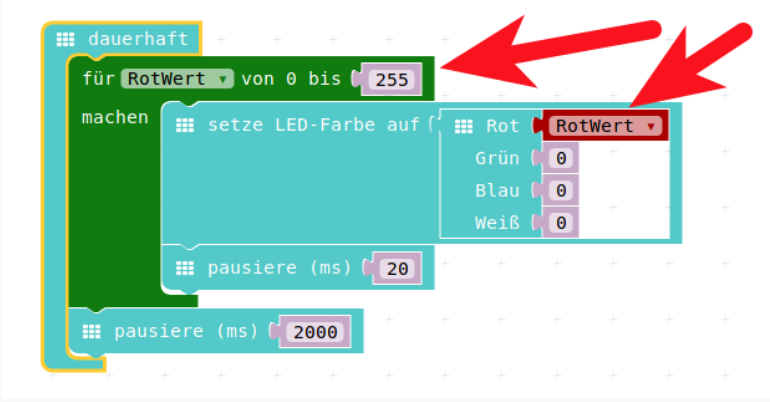
Vergleich der beiden Schleifen





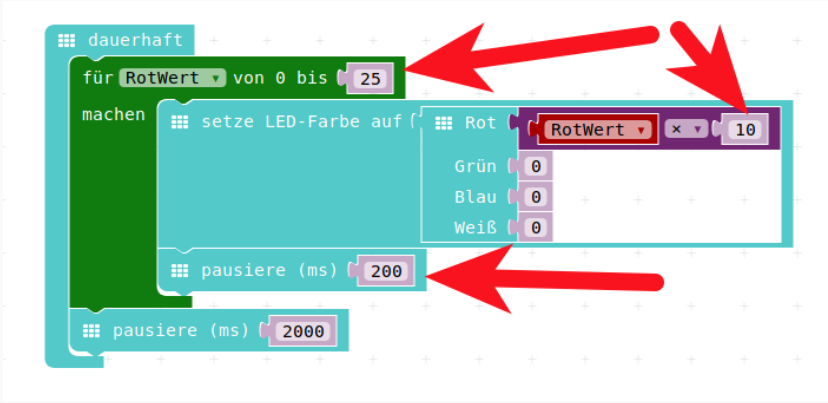
Hausaufgabe 1 RGB-Led

Schleife mit Rot-Anteil



- Eine Schleife um den Rot-Anteil zu verändern
- Wieviele Schleifendurchläufe sind das?
- Wie lange läuft dieses Programm einmal (inklusive pausieren!) ?

Zweite Rot-Anteil-Schleife

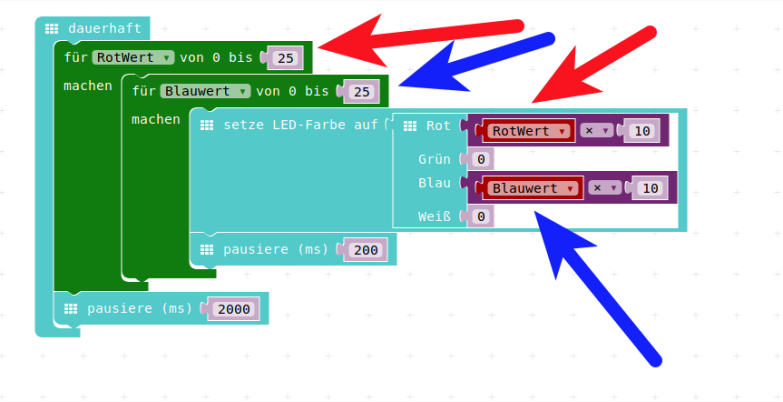


- Nochmal eine Schleife um den Rot-Anteil zu verändern
- Wieviele Schleifendurchläufe sind das?
- Wie lange läuft dieses Programm einmal (inklusive pausieren!) ?
- Seht Ihr einen Unterschied?

Warum diese Änderung

- Wir sehen kaum einen Unterschied
- Aber wir haben nur 25 anstatt 255 Schleifendurchläufe
- Nun wollen wir Schleifen “verschachteln”
- Wenn wir da auch nur 25 anstatt 255 Durchläufe machen, sehen wir etwas
- und das in einigermassen sinnvoller Zeit. . .
- Siehe nächste Seite


Verschachtelte Schleife



- Diesmal zwei verschachtelte Schleifen
- Die äussere ändert Rot, die innere Blau
- Wieviele Schleifendurchläufe sind das aussen?
- Wieviele Schleifendurchläufe sind das innen?
- Wie lange läuft dieses Programm einmal (inklusive pausieren!) ?

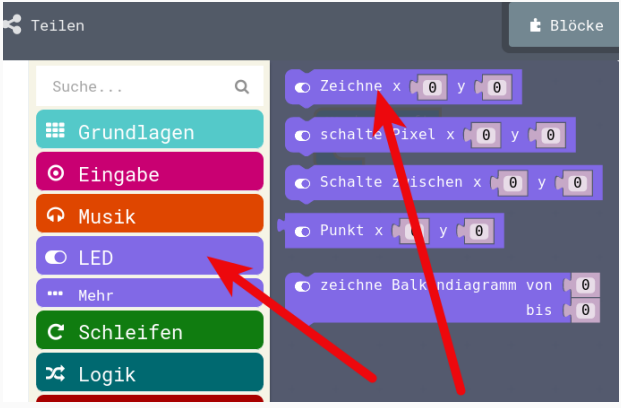
Hausaufgabe 2 : Display füllen

Zeichen X Wert Y Wert

Dazu schauen wir uns aus dem Menu **LED** einmal einen Befehl etwas genauer an:

Zeichne X Wert Y Wert

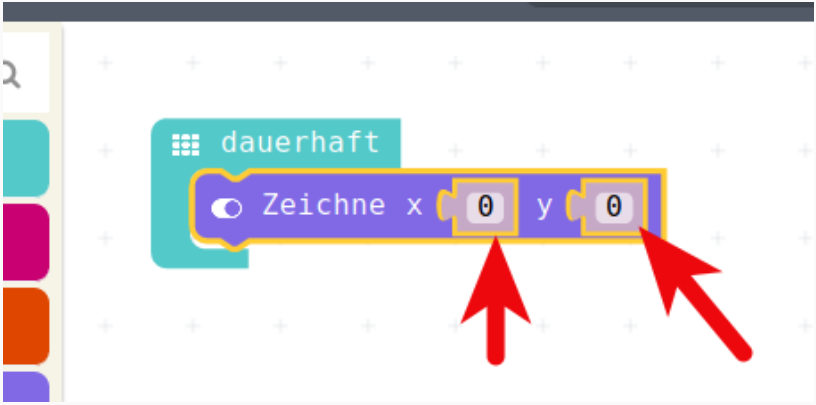
Damit kann man einzelne LEDs auf unserem 5 x 5 LED-Bildschirm einschalten.



Simulator ausprobieren

- Das probieren wir gleich mal im Simulator aus,
- Wir klicken diesen einen Befehl in die **Dauerhaft**-Schleife ein
- Wir spielen mit den X und Y-Werten
- Wir schauen, was im Simulator passiert.

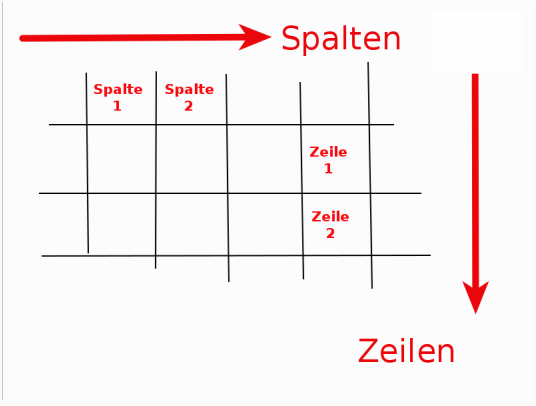

Jede LED einschaltbar



- Wir können durch Verändern der Werte für X und Y zwischen 0 und 4 jede beliebige LED auf unserem 5 x 5 - Display einschalten.

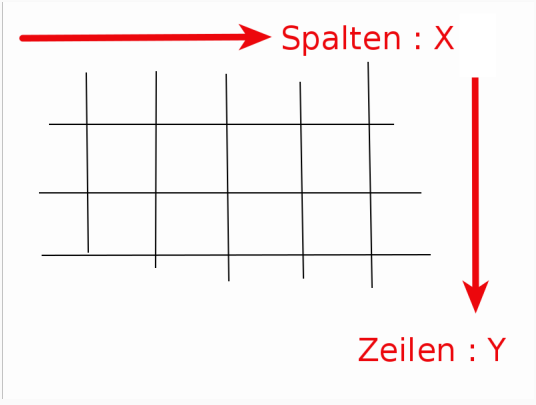
Zeilen und Spalten

- Das kommt aus der Mathematik, man bezeichnet im Allgemeinen die Spalten-Richtung mit X
- und man bezeichnet die Zeilen mit Y



Zeilen und Spalten

- Spalten, horizontale Richtung, Links-Rechts : **X**
- Zeilen, vertikale Richtung, Oben-Unten : **Y**



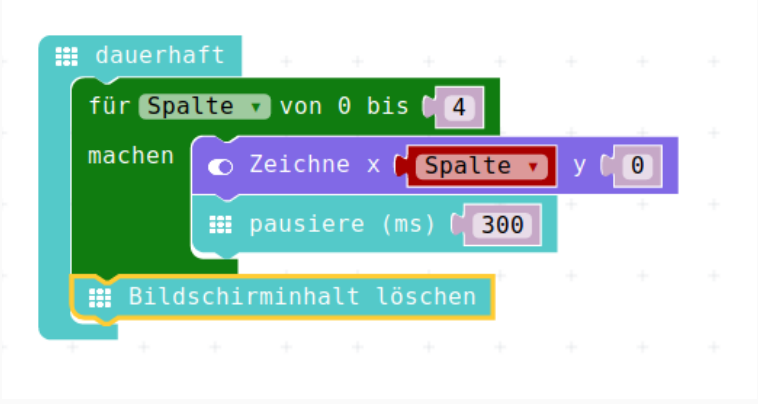
- Auch hier wieder - wie oft beim Programmieren : es geht bei 0 los !

Eine Zeile füllen

- Nun beschränken wir uns also auf eine Zeile in unserer LED-Anzeige
- Wir zeichnen einen Punkt nach dem anderen
- Dazu erhöhen wir jeweils den Spalten-Wert
- Also **X** !


Schleife mit Variablen

- Mit unserem Schleifen-Wissen können wir das mit Schleifen programmieren
- Das Ergebnis schauen wir zuerst im Simulator an
- Wenn alles passt, können wir das auch in den Calliope programmieren.

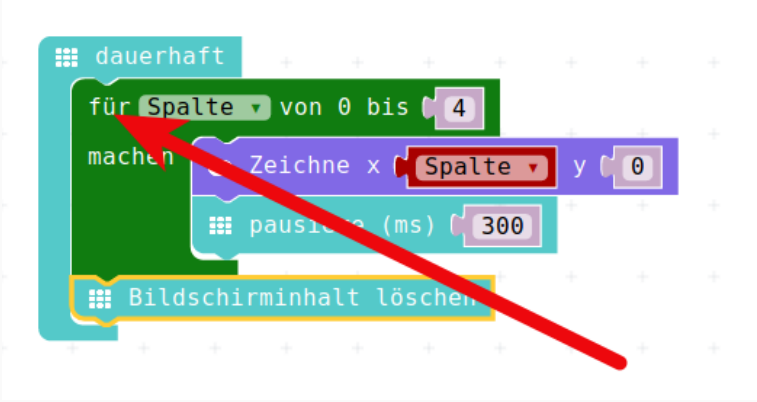


Finale Schleife mit Variablen

- Diese Schleife läuft nun **5 mal**!
  - Zuerst mit Variable **Spalte = 0**, dann wird die LED in Spalte (X) **0** Zeile (Y) 0 gesetzt
  - Dann mit Variable **Spalte = 1**, dann wird die LED in Spalte (X) **1** Zeile (Y) 0 gesetzt
  - Anschliessend mit Variable **Spalte = 2**, dann wird die LED in Spalte (X) **2** Zeile (Y) 0 gesetzt
  - Daraufhin mit Variable **Spalte = 3**, dann wird die LED in Spalte (X) **3** Zeile (Y) 0 gesetzt
  - Schliesslich ein letztes Mal mit Variable **Spalte = 4**, dann wird die LED in Spalte (X) **4** Zeile (Y) 0 gesetzt

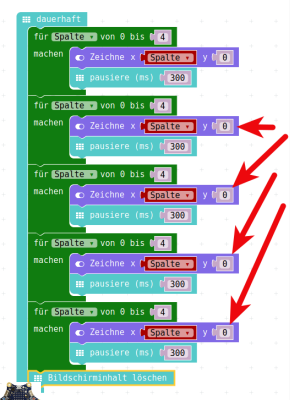
Nächster Versuch, Ganzer Bildschirm

- Nun können wir also mit einer Schleife eine Zeile füllen, durch die Index-Schleife wird die Zeile **spaltenweise** gefüllt
- Wir wollen aber immnoch den ganzen Bildschirm einzeln mit LEDs füllen.
- Dazu kopieren wir nun die Index-Schleife 4 mal



4 mal kopiert

- Wir hängen sie viermal untereinander
- Das Bildschirm-Löschen schieben wir dabei nach unten, das wollen wir nur einmal ganz am Schluss haben
- Nun haben wir 5 identischen Zeilenfüller, verwirklicht durch eine Schleife
- Wir müssen nun noch die Zeilen-Nummern anpassen.



Fertiges Programm, Ganzer Bildschirm

- Wenn wir das gemacht haben, haben wir ein Programm, das den Bildschirm LED für LED füllt
- Und das ganze durch die schlaue Verwendung von Schleifen.

Fünf fast identische Codestücke

Was wir aber immer noch haben, sind 5 fast gleich aussehende Schleifen. Und diese 5-fache Wiederholung wollen wir nun durch eine zweite, verschachtelte Schleife ersetzen.

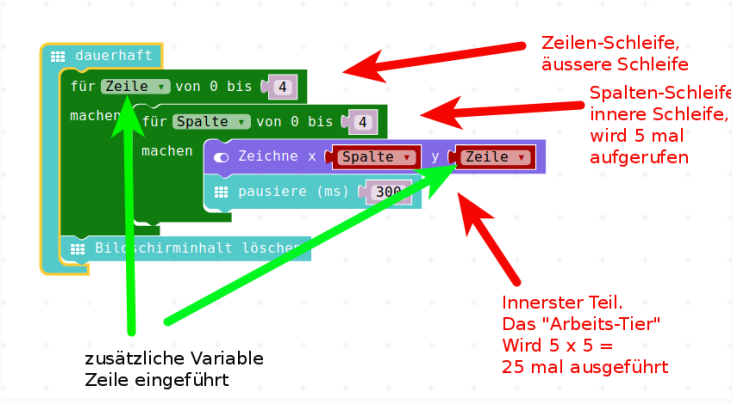
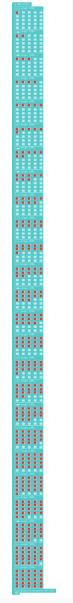


Figure 4: Zwei verschachtelte Schleifen

“Böser” Auftrag-Geber

Nehmt an, Ihr hättet die Schleife noch nicht kennengelernt, und hättet die Aufgabe mit einzelnen zeige LED-Befehlen programmiert:



“Böser” Auftrag-Geber 2

Nun kommt der Auftrag-Geber und möchte nun doch lieber anstatt zeilenweise den Bildschirm zu füllen, diesen spaltenweise gefüllt haben.

In dem gezeigten Beispiel müsst Ihr alles neu programmieren.
23 mal andere Bildschirm-Inhalte von Hand malen.
(Der erste und der letzte Bildschirm-Inhalt passen)
Wollt Ihr es versuchen?

Ich würde lieber die Variante mit den Schleifen nehmen und dort nur die zwei Schleifen bzw die beiden Index-Variable vertauschen!

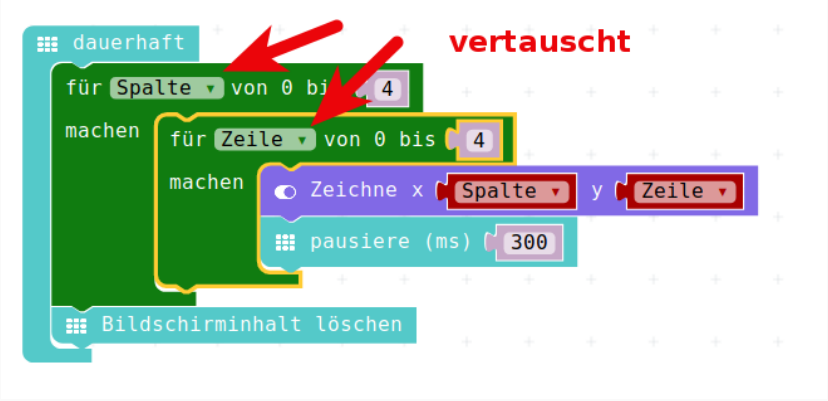



Figure 5: Zwei verschachtelte Schleifen

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

08_02_Motoren_Auffrischen

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019


DC-Motoren Auffrischen

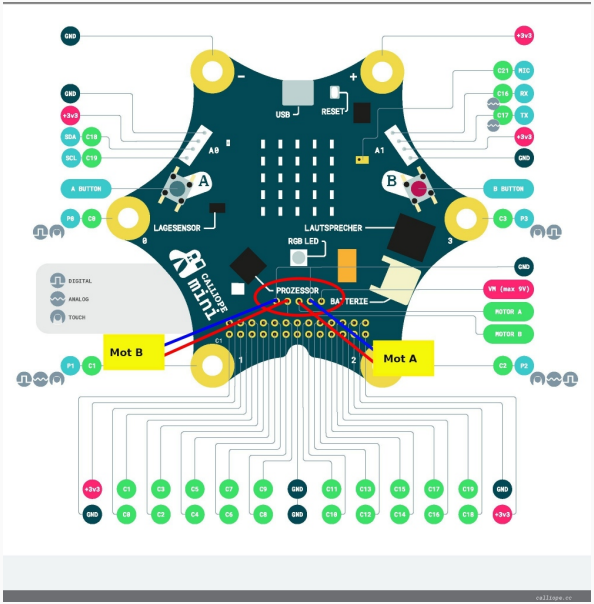
Ein oder Zwei Motoren

Je nach Verwendungszweck kann man an den Calliope entweder

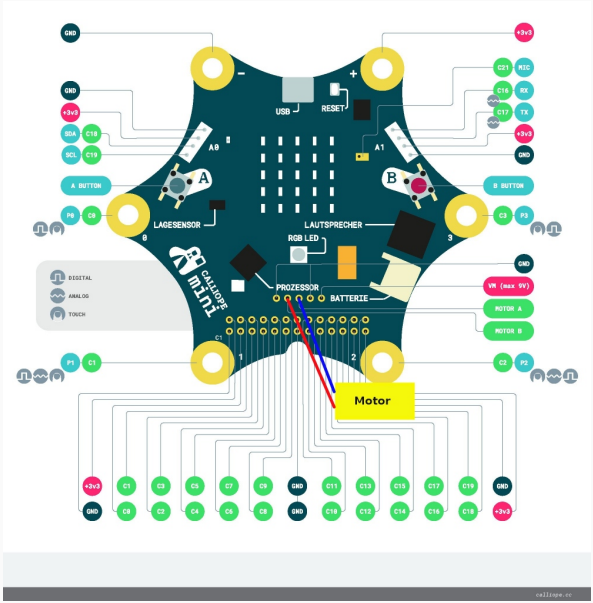
- 1 Motor anschliessen, der kann dann vorwärts und rückwärts drehen
- 2 Motoren anschliessen, die können dann nur einzeln vorwärts drehen

So sehen die beiden Möglichkeiten zum Anschluss von zwei oder einem Motor aus:

Zwei Motoren



Ein Motor


GND
+3v3
USB
RESET
C21
MIC
C16
RX
C17
TX
SDA
C18
SCL
C19
A0
A1
A
B
A BUTTON
B BUTTON
P0
C0
C3
P3
LAGESENSOR
LAUTSPRECHER
RGB LED
DIGITAL
ANALOG
TOUCH
CALLIOPE mini
PROZESSOR
BATTERIE
VM (max 9V)
MOTOR A
MOTOR B
P1
C1
C2
P2
Motor
C1
C3
C5
C7
C9
C11
C13
C15
C17
C19
C0
C2
C4
C6
C8
C10
C12
C14
C16
C18
calliope.cc

Motor-Programmierung für Calliope

Nachdem wir nun wissen, wie wir einen einzelnen Motor an den Calliope elektrisch anschliessen, wollen wir den Ausgang für den Motor auch mit Software programmieren.

Im ersten Schritt wollen wir nur ganz einfach den Motor ein- und ausschalten können. Dazu wollen wir mit dem linken Knopf ein und mit dem rechten Knopf ausschalten.

Motor-Menu

Die Motor-Ansteuerung findet sich im Menu Motoren:

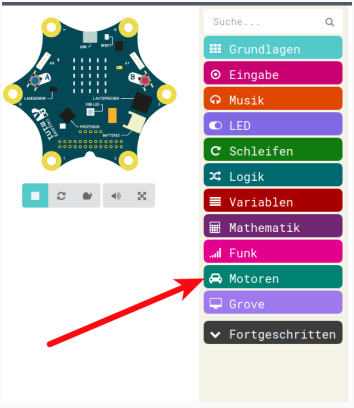


Figure 1: Menu Motor

Motor Befehle

Es gibt nicht viele Befehle zum Steuern von Motoren:

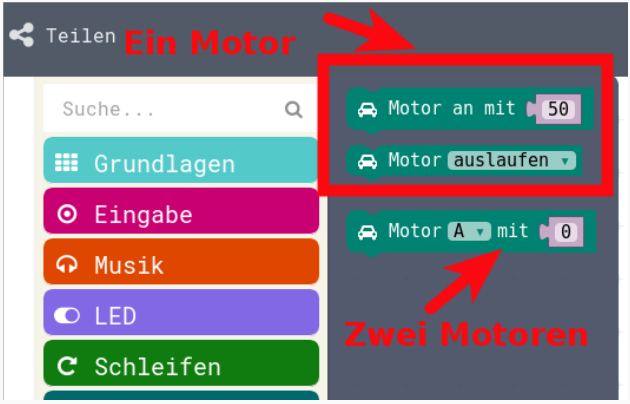


Figure 2: Motor Befehle

Erstes Motor-Programm

Das war unser erstes Motor-Programm

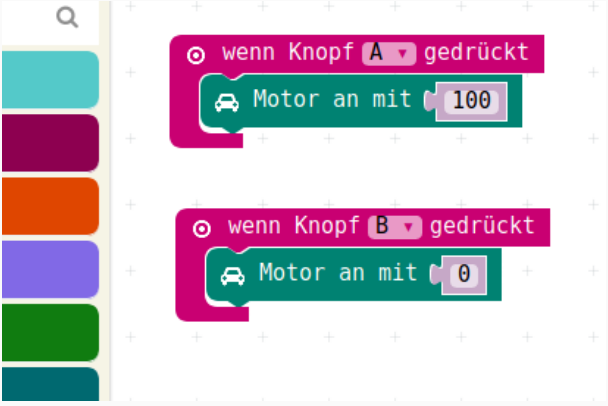


Figure 3: Motor Programm 1

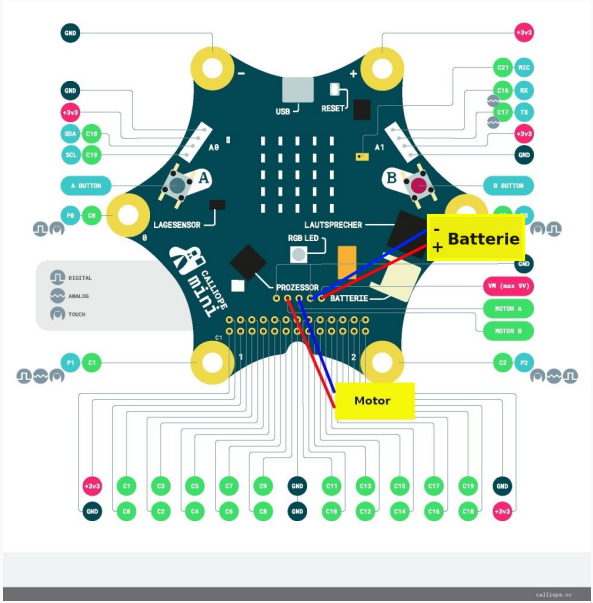
Dieses Programm können wir leider im Simulator gar nicht nutzen.

Also müssen wir das Programm auf den Calliope runterladen und dort ausprobieren.

Anschluss von zusätzlicher Batterie

- Der Calliope erlaubt **für den Motor** den Anschluss einer zusätzlichen Batterie.
- Deren Spannung wird **NUR** zum Ansteuern des Motors verwendet
- Die Batterie darf nicht mehr als *9V* haben
- Sie muss zum Motor passen

Zusätzliche Batterie



Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

08_03_LageSensor

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Motorsteuerung mit Lage-Sensor

Der Lagesensor

Das funktioniert ja schon mal ganz gut.
Nun möchten wir mit diesem einfachen Motor-Steuerungs-Programm auch noch eine andere Eingangs-Möglichkeit ausprobieren:

Den Lage-Sensor!

Der Calliope hat einen Lage-Sensor eingebaut, der in allen Raumrichtungen funktioniert.

Also :

- Oben / Unten
- Links / Rechts
- Vorne / Hinten

Abfragen des Lagesensors

Die Abfragen, um den Lage-Sensor genau auszuwerten, sind recht kompliziert. Man muss Koordinaten-Systeme verstehen und man sollte Winkelrechnung verstehen. Beides ist in Euerem Alter wahrscheinlich noch nicht der Fall.

Zusätzlich zu den genauen Abfrage-Möglichkeiten, die schwierig zu verwenden sind, hat der Calliope aber auch die Möglichkeit, sehr einfach den Lage-Sensor abzufragen.

Das wollen wir nun tun:

- Beim Gerade halten des Calliope soll der Motor aus sein.
- Beim Kippen nach links soll er sich nach vorne drehen
- Beim Kippen nach rechts soll er sich nach hinten drehen.

Eingabe-Menu

Sowohl die genauen, schwierigeren Befehle als auch die Einfachen befinden sich im Menu Eingabe:

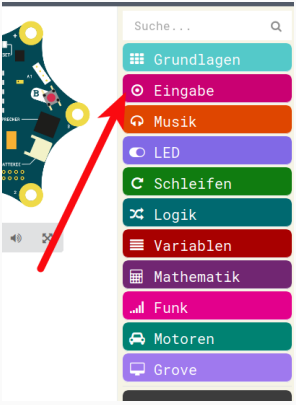


Figure 1: Menu Eingabe

Inhalte Eingabe-Menu

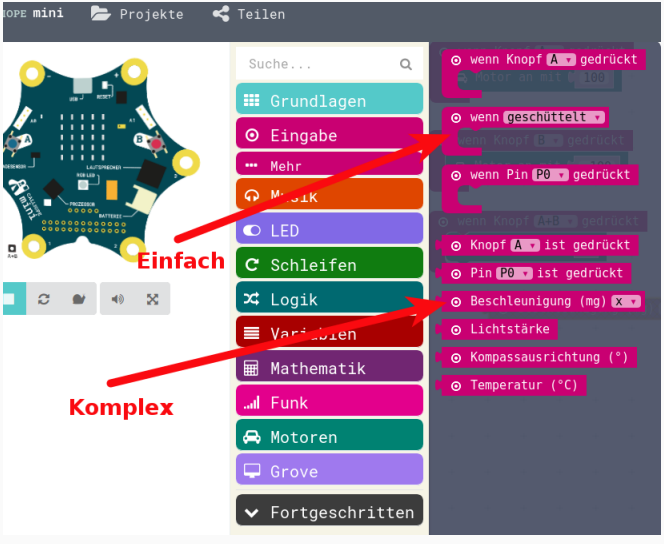


Figure 2: Menu Eingabe Inhalt

Wenn geschüttelt

Nun ziehen wir drei mal das **wenn geschüttelt** in unseren Arbeits-Bereich:

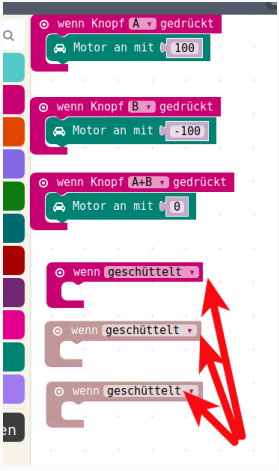


Figure 3: Dreimal Geschuettelt

Umwandeln

Diese wandeln wir nun alle durch Druck auf das Dreieick:

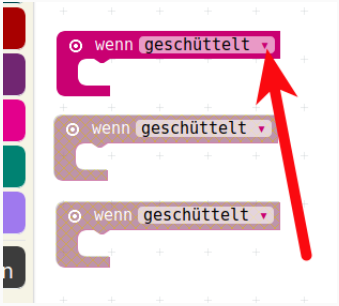


Figure 4: Dreieck

um in drei verschiedene Reaktionen:

- “nach links neigen”
- “Display nach oben”
- “nach rechts neigen”

Möglichkeiten

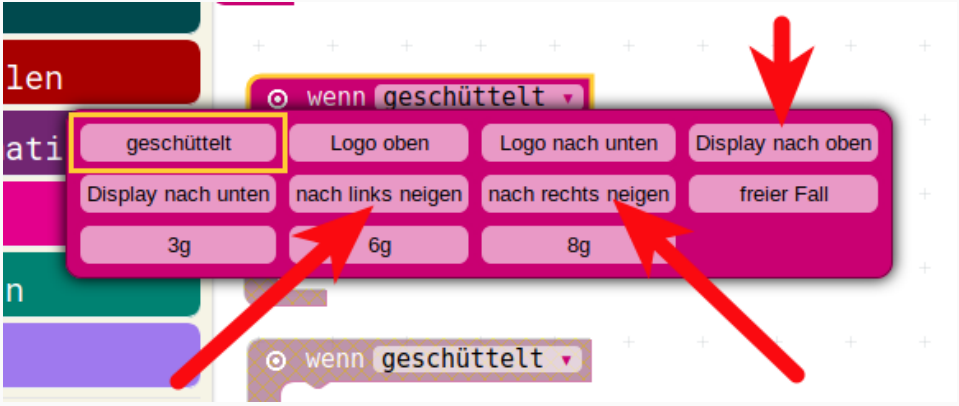


Figure 5: Lage-Sensor

Unser Programm

Damit sieht unser Programm nun so aus:

Figure 6: Lage-Sensor drin

Fertiges Programm

und wenn wir nun die entsprechenden Befehle von oben nach unten schieben, dann können wir unseren Motor durch kippen steuern.

Figure 7: Lage-Sensor benutzt

Runterladen in Calliope

Dieses Programm können wir nun auch in den Calliope laden.

Achtung: Zumindest bei manchen Kombinationen von Calliope und Computer (und vermutlich angeschlossenem Computer-Ladegerät) hat der Lage-Sensor **NICHT** richtig funktioniert.

==>

Bitte steckt in diesem Fall das USB - Kabel aus und betreibt Euren Calliope nur über Batterie.

JavaScript-Code

Java-Script-Code

```
input.onGesture(Gesture.TiltLeft, () => {
    motors.motorPower(100)
})
input.onGesture(Gesture.TiltRight, () => {
    motors.motorPower(-100)
})
input.onGesture(Gesture.ScreenUp, () => {
    motors.motorPower(0)
})
```

Download Hex-Code

Hex-code

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

08_04_Funkuebertragung

Calliope-Kurs Kinder

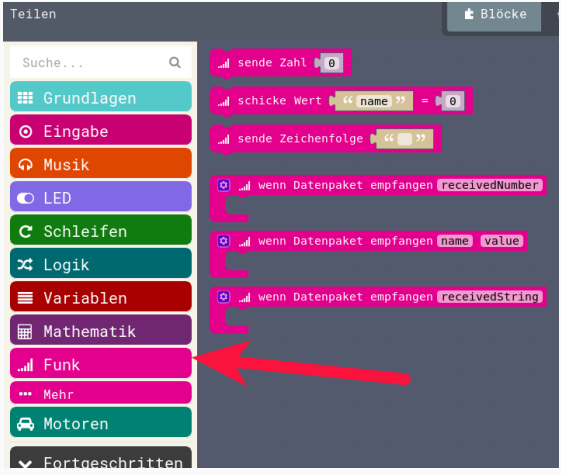
Jogi Künstner, Turbine Brunnen

Frühjahr 2019


Funk - Uebertragung

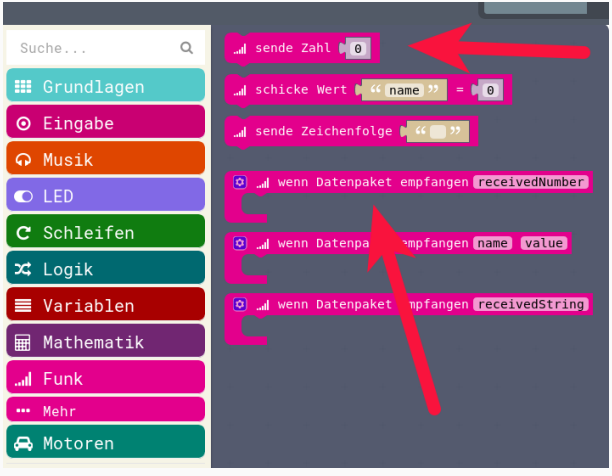
Menu Funk

Im Menu Funk findet sich:

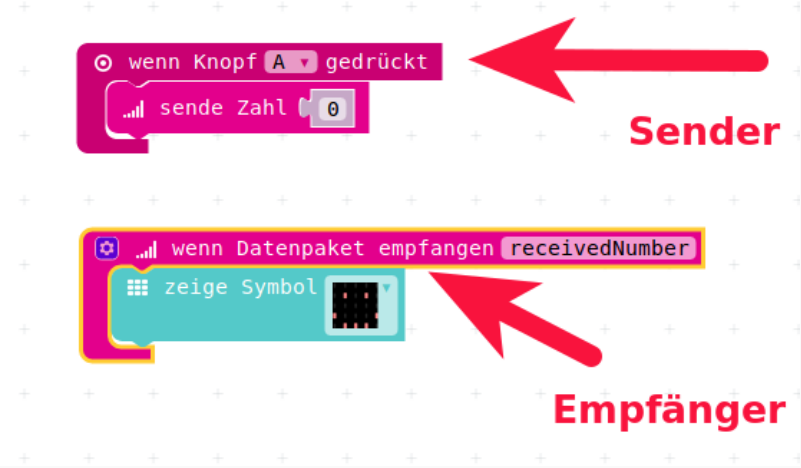


Zahlen Funken

Wir senden und empfangen zuerst mal Zahlen

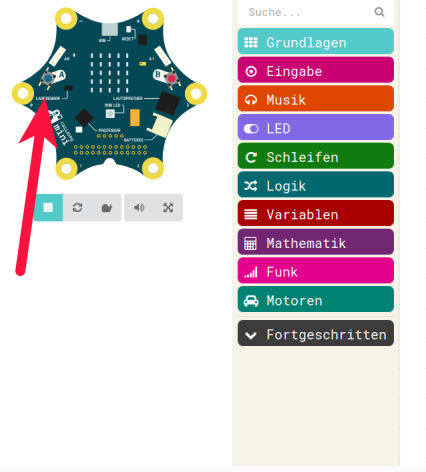


Simpler Sender/Empfänger



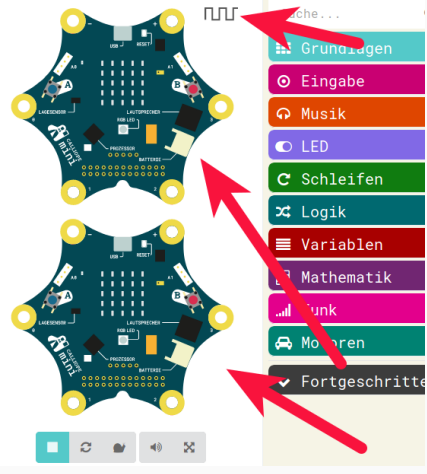
Simulator kann auch Funk

- Sobald man den Knopf A drückt
- merkt das der Simulator und blendet einen zweiten Calliope ein



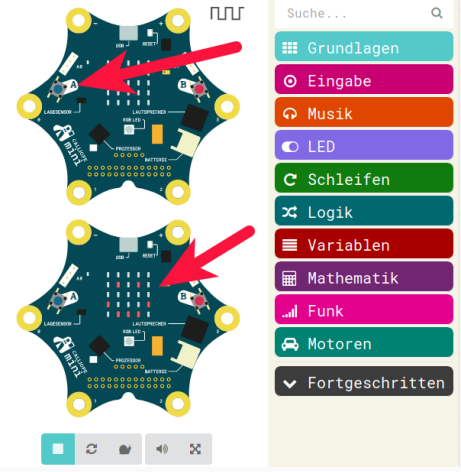
Zwei simulierte Calliopes

- Nun sind zwei simulierte Calliopes zu sehen
- Und der obere ist am Funken



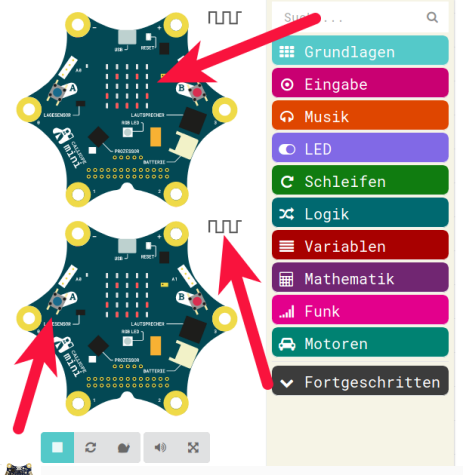
Nochmal Knopf drücken

- Nun muss man nochmal Knopf A drücken
- damit der andere Calliope das empfangen kann

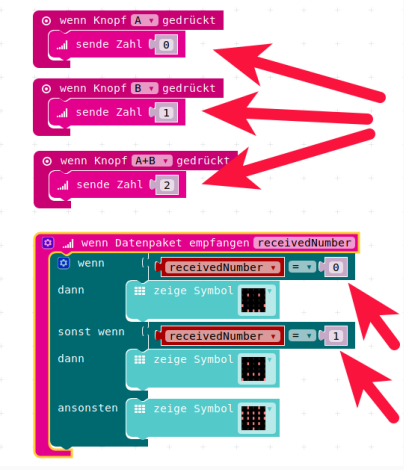


Anderer Calliope sendet auch

- Drücken auf zweitem Calliope
- bringt den auch zum Senden
- und der erste/obere empfängt



“Sinnvolle” Inhalte Senden



Mit Nachbarn funken

- Einigt euch auf 3 Zahlen, die Ihr senden wollt (z.B. 0-2)
- (am Besten verwendet Ihr das Programm aus diesem Beispiel)
- Jeder kann seinen eigenen Empfangs-Teil mit verschiedenen Symbolen machen
- Dann versucht, Euch gegenseitig was zu senden
- Klappt das ?
- . . .
- . . .
- . . .
- ????
- Im Simulator gehts doch, oder ?


Mit Nachbarn funken II

- Nun probiert mal das HEX-File von einem der beiden Computer in beide Calliopes zu programmieren.
- Jetzt gehts?
- Im Simulator gehts auch ?
- Habt Ihr nen Fehler gemacht ?
- Wer von Euch ?
-
- Antwort: Vermutlich hat keiner von Euch einen Fehler gemacht...

Mit Nachbarn funken III

- Wer von Euch hat Funkgeräte?
- Können die immer miteinander kommunizieren
- Können die direkt mit den Funkgeräten von Euren Freunden funken?
- Oder muss man da was einstellen, damit das funktioniert?
- **JA** : Der Sende und Empfangs-Kanal !
- Beim Calliope heisst das **“Gruppe”**
- und befindet sich im **Funk** -> **Mehr** - Menu

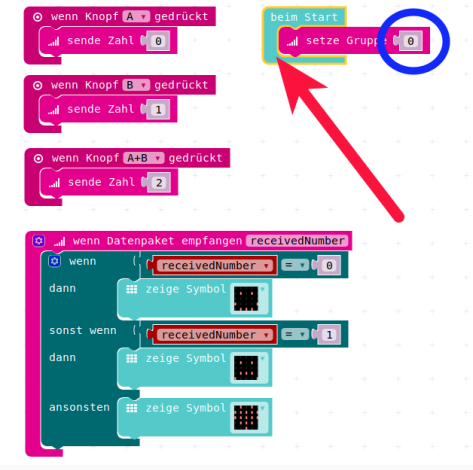

Setze Gruppe

Erklärung

- Wenn die Funk-Gruppe nicht von Euch im Programm-Lauf gesetzt wird
- dann setzt der Computer das im Hintergrund
- automatisch auf einen zufälligen Wert zwischen 0 - 255
- d.h. ein HEX-File enthält einen “ausgewürfelten” Kanal
- ein anderes HEX-File enthält einen anderen “ausgewürfelten” Kanal
- darum: Enwteder gleiches HEX-File verwenden oder Kanal setzen
- => Setze Gruppe XXX beim Start

Setze Gruppe

Nun bauen wir also das entsprechende **Setze Gruppe** - Befehl **beim Start**

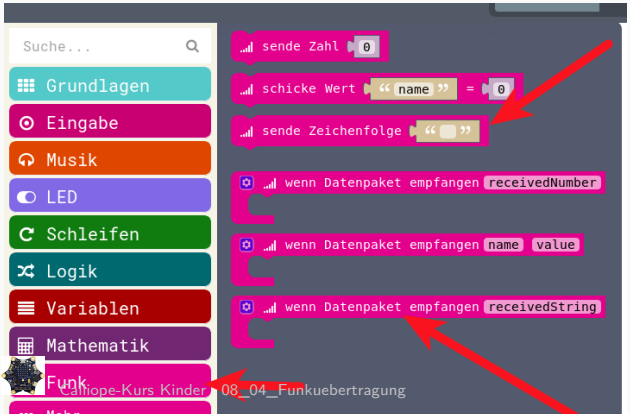


Zeichenfolgen austauschen

Ebenso kann man - anstatt einzelne Zahlen - ganze Texte von einem Calliope zum anderen schicken

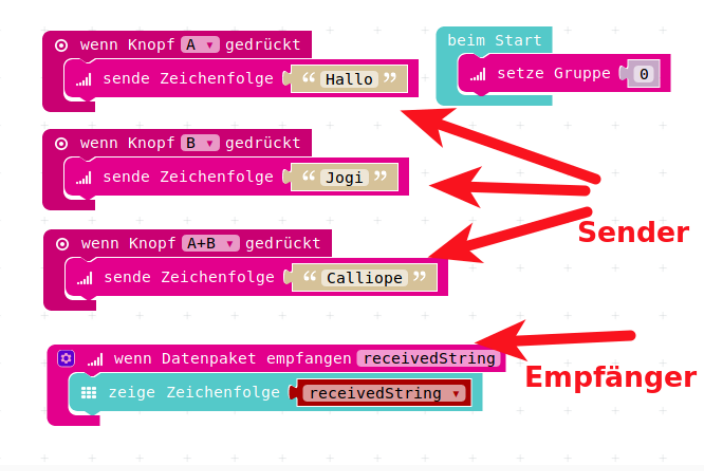
Die beide dafür verantwortlichen Befehle sind

- **sende Zeichenfolge**
- **wenn Datenpaket empfangen receivedString**



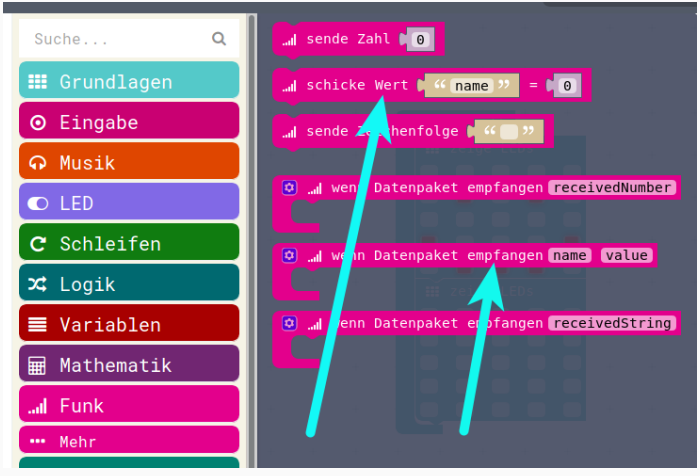
Texte senden und empfangen

So kann man sich Text-Nachrichten hin und her senden



Variablen und Werte schicken

Und dann gibt es noch die Möglichkeit, ganze Zeichenketten **UND** gleichzeitig Zahlenwerte zu verschicken und zu empfangen:



Variablen und Werte schicken II

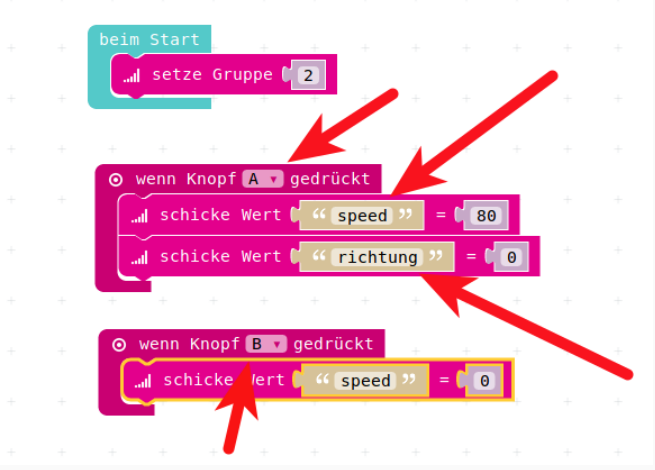
In der Praxis kann man das nutzen, um zum Beispiel verschiedene Variablen anhand der Variablen-Namen zu verschicken und gleichzeitig deren Werte zu verschicken.

Beispiel:

- Ferngesteuertes Auto
- Geschwindigkeit von 0 bis 100 (kein Rückwärts-Gang...)
- Richtung von -50 (nach links fahren) bis + 50 (rechts fahren) => 0 = gerade aus
- Das heisst, das ferngesteuerte Auto muss in der Lage sein, 2 Variablen zu empfangen
- Einfaches Beispiel :
  - Knopf A startet das Auto mit Geschwindigkeit 80 und Richtung 0
  - Knopf B stoppt das Auto mit Geschwindigkeit 0
- Bitte jedes Kind einen anderen Kanal verwenden, ab Kanal 2 im Uhrzeigersinn jeweils 1 mehr


Variablen und Werte schicken III

Die Variablen, die das Auto versteht, sind **speed** und **richtung**



Fernsteuerung für das Auto I

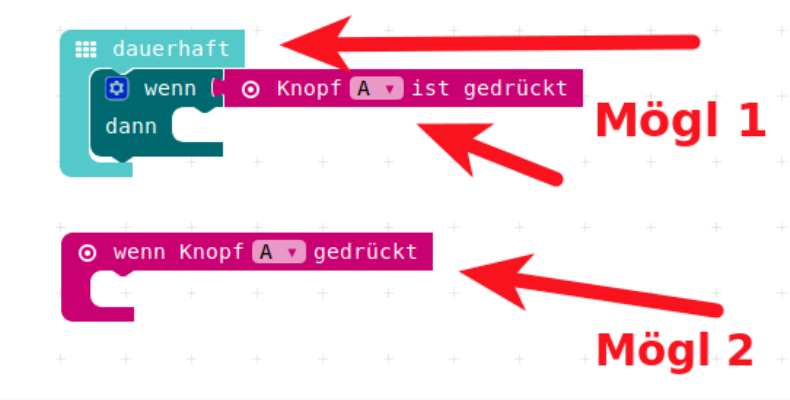
Nun wollen wir eine “richtige” Fernsteuerung für das Auto machen:

- Wenn **KEIN** Knopf gedrückt ist, dann **speed = 0**
- Wenn **Knopf A** gedrückt ist, dann **richtung = -40** und **speed = 80**
- Wenn **Knopf B** gedrückt ist, dann **richtung = 40** und **speed = 80**
- Wenn **Knopf A+B** gedrückt ist, dann **richtung = 0** und **speed = 90**

Achtung : Es gibt kein **Wenn Knopf losgelassen** Ihr müsst Euch anders behelfen.

- Dauerhaft 0 senden und **Wenn Knopf gedrückt** dann etwas anderes senden, oder
- Dauerhaft die Tasten abfragen in der Dauerhaft-Schleife und dort die Entscheidungen treffen

Fernsteuerung für das Auto II



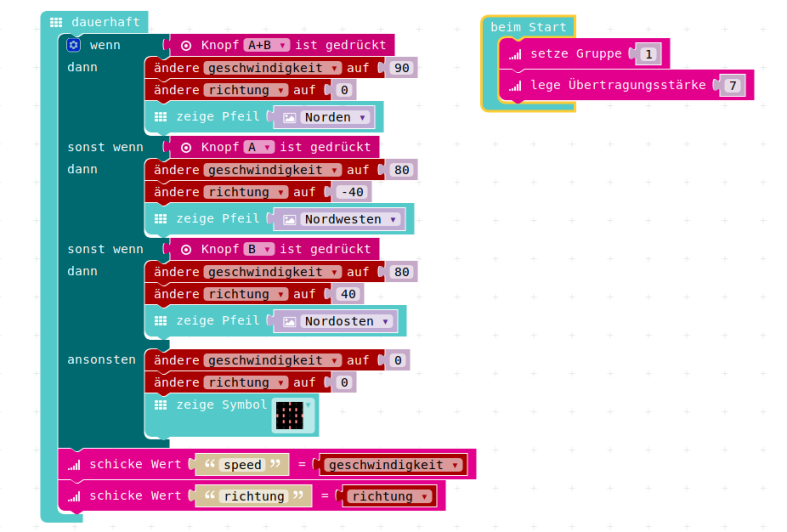
Fernsteuerung für das Auto III

Nochmal die Aufgabe:

- Wenn **KEIN** Knopf gedrückt ist, dann **speed = 0**
- Wenn **Knopf A** gedrückt ist, dann **richtung = -40** und **speed = 80**
- Wenn **Knopf B** gedrückt ist, dann **richtung = 40** und **speed = 80**
- Wenn **Knopf A+B** gedrückt ist, dann **richtung = 0** und **speed = 90**

Fernsteuerung für das Auto IV

Mögliche Lösung



Lizenz / Copyright

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0

08_05_ExternerLautsprecher

Calliope-Kurs Kinder

Jogi Künstner, Turbine Brunnen

Frühjahr 2019

Zusätzliche Batterie/Externer Lautsprecher

Zusätzliche Möglichkeiten

Nachdem wir für den Motor die 5-polige Erweiterungs-Leiste aufgelötet haben, möchte ich noch zwei zusätzliche Möglichkeiten dieser Leiste aufzeigen. Man kann daran anschliessen:

- Zusätzliche Batterie für die Motoren
- Externer Lautsprecher, für mehr Krach

Zusätzliche Batterie für Motor

Der Calliope wird - wie wir gelernt haben - normalerweise mit 2 Batterien a 1.5 Volt betrieben. Das macht zusammen 3 Volt.

Wenn man nun einen Motor anschliessen möchte, der mehr Spannung braucht, dann würde der sich damit kaum bewegen.

Darum haben die Erbauer des Calliope noch eine Möglichkeit vorgesehen, eine zusätzliche Batterie anzuschliessen!
Diese zusätzliche Batterie

- ist aber nur zum Betrieb des Motors/der Motoren gedacht
- sie muss zusätzlich zur normalen Batterie angeschlossen werden
- Sie muss zur Spannung des Motors / der zwei Motoren passen
- sie darf maximal 9V haben

Anschluss

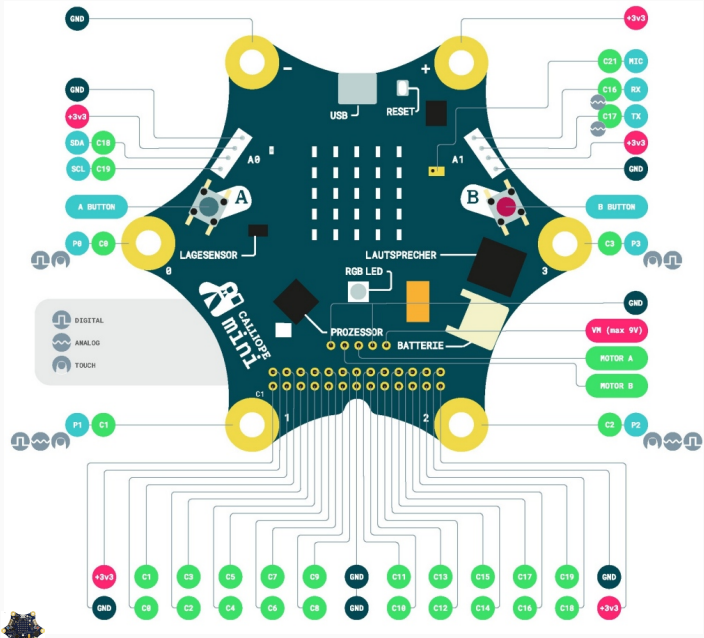
Die zusätzliche Batterie wird an den beiden rechten Pins angeschlossen, ganz aussen Plus, daneben Minus.

Auf dem offiziellen Bild des Calliope sieht man das ganz gut:

Quelle Bild nächste Seite :

https://calliope-mini.github.io/assets/v10/img/Calliope_mini_1.0_pinout_fin.jpg:

Offizielles Bild



Detail-Auschnitt

Hier nochmal ein Ausschnitt daraus, mit den Anschlüssen farbig gekennzeihnet:

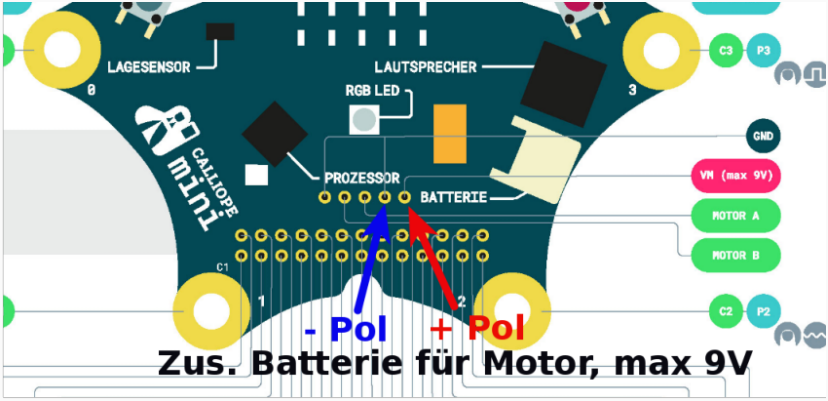


Figure 1: Ausschnitt

Und in Betrieb

In Betrieb sieht das dann so aus:

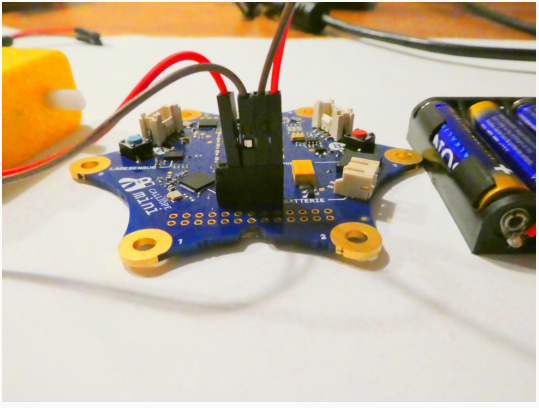


Figure 2: Anschluss Calliope

Externer Lautsprecher

Der eingebaute Lautsprecher am Calliope ist schon sehr klein und sehr leise.

An die Leiste kann man auch einen externen Lautsprecher anschliessen.

Dabei kann man entweder einen kleinen, sogenannten Passiv-Lautsprecher anschliessen, dieser braucht keine zusätzliche Stromversorgung.

Oder aber man kann einen sogenannten Aktiv-Lautsprecher anschliessen, das sind die Lautsprecher, wie man sie auch von Computern kennt, diese haben eine zusätzliche Strom-Versorgung.

Ein alter Passiv-Lautsprecher

Figure 3: Lautsprecher

Klinken-Anschluss

In Jedem Fall ist es sinnvoll, das über einen sogenannten Klinken-Stecker zu realisieren, der Anschluss sieht dann so aus:

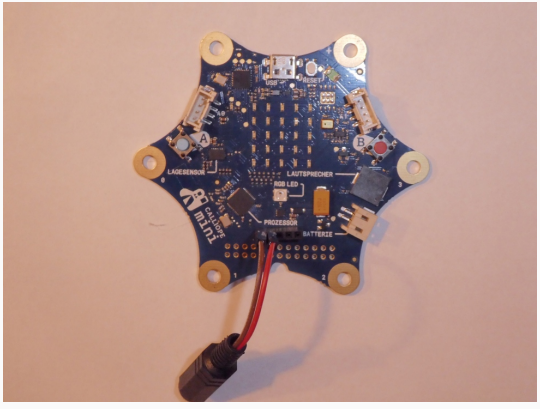


Figure 4: Klinkenanschluss

Klinkenstecker

Damit kann dann ein externer Laustsprecher oder Kopfhöhrer angeschlossen werden.

Der Klinken-Anschluss sieht so aus:

Figure 5: Klinkenstecker

Auf dem Calliope-Schaltbild

Anschluss auf dem offiziellen Calliope-Schaltbild:

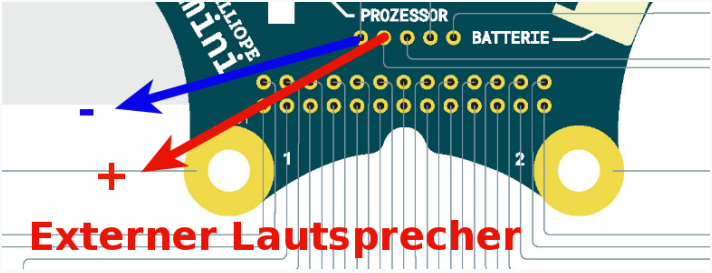


Figure 6: Lautsprecher

ACHTUNG :

Da der Lautsprecher die gleichen Anschlüsse wie die Motoren verwendet, kann man nicht beides gleichzeitig in einem Programm machen. Man muss sich also entscheiden, ob man mit seinem Calliope Musik abspielen will, oder ob man Motoren ansteuern will.

Lizenz/Copyright-Info

Für alle Bilder auf diesen Folien/Seiten gilt:

- Autor: Jörg Künstner
- Lizenz: CC BY-SA 4.0