

# 09\_04\_TemperaturAmpelBesser

Calliope-Kurs (Lehrer Herrengasse)

---

Jogi Künstner, Turbine Brunnen

Frühjahr 2019



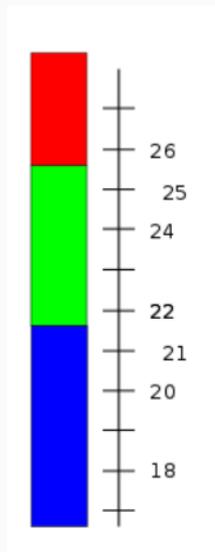
## **Die verbesserte Temperatur-Ampel**

---

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)





**Figure 1:** Neue Temperatur-Bereiche

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.

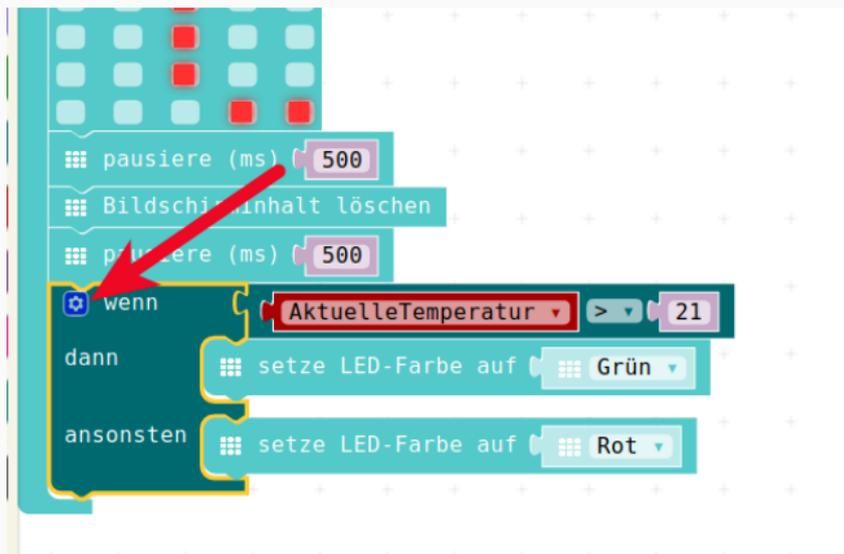


Figure 2: ToolBox



# 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**

The image shows a Scratch-style programming environment. On the left, there is a vertical palette of colored blocks. The main workspace contains a script starting with a 'wenn' (if) block. The condition is 'AktuelleTemperatur > 21'. The 'dann' (then) branch contains a 'setze LED-Farbe auf' (set LED color to) block with 'Grün' (Green) selected. The 'ansonsten' (else) branch contains a 'setze LED-Farbe auf' (set LED color to) block with 'Rot' (Red) selected. A red arrow points from the 'wenn' block in the workspace to the 'else if' block in the palette. Another red arrow points from the 'else if' block in the palette to the 'wenn' block in the workspace. The word 'itten' is visible at the bottom left of the workspace.



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

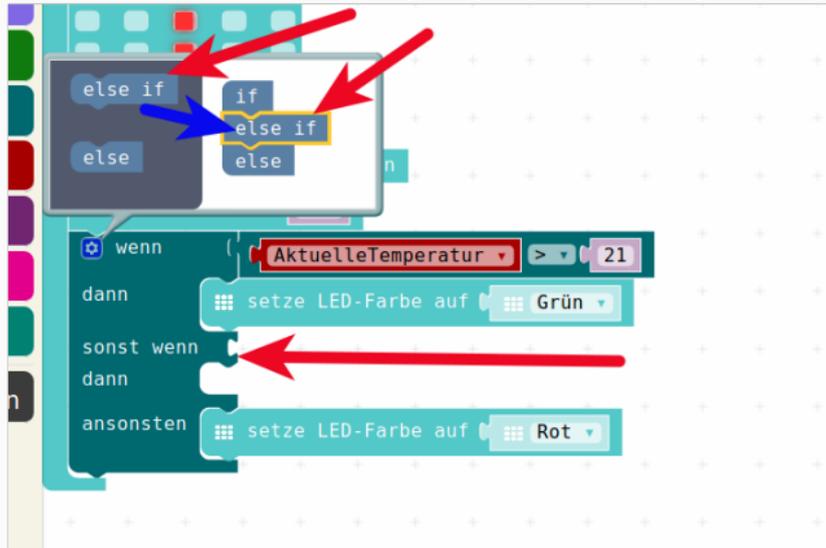


Figure 4: Toolbox Miniatur

# 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.

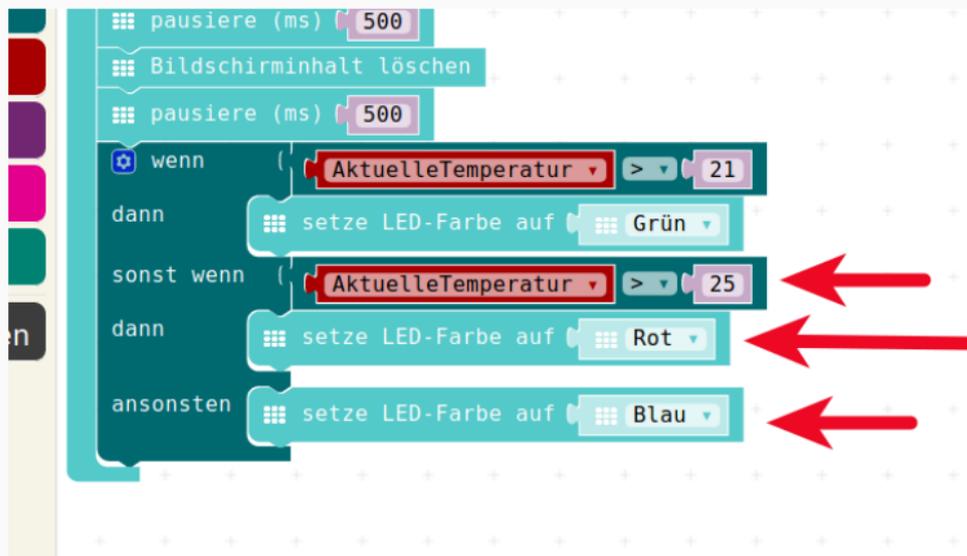


Figure 5: Neues Programm Elself

# Ein Bug ( ein Fehler) !

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

Dazu können wir mal versuchen, die Temperatur auf  $> 26^{\circ}\text{C}$ , also z.B.  $30^{\circ}\text{C}$  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) !

The image shows the Scratch environment. On the left is a Scratch sprite of a green robot on a blue circuit board. Below it is a toolbar with icons for home, undo, bug search, redo, and delete. A red arrow points to the bug search icon. In the center is a search bar with the text 'Suche...' and a magnifying glass icon. Below the search bar is a list of categories: Grundlagen, Eingabe, Musik, LED, Schleifen, Logik, Variablen, Mathematik, Funk, Motoren, and Fortgeschritten. On the right is a script area with the following code blocks:

```
ändere AktuelleTemperatur auf Temperatur (°C)
zeige Nummer AktuelleTemperatur
pausiere (ms) 500
zeige LEDs
pausiere (ms) 500
Bildschirminhalt löschen
pausiere (ms) 500
wenn AktuelleTemperatur > 21
dann setze LED-Farbe auf Grün
sonst wenn AktuelleTemperatur > 25
dann setze LED-Farbe auf Rot
ansonsten setze LED-Farbe auf Blau
```

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 zuehörige Aktion durchgeführt: Setzen der Farbe auf grün
- Das **Sonst Wenn** wird gar nicht erreicht und darum dann auch die Überprüfung auf  $> 25^{\circ}\text{C}$  erst gar nicht durchgeführt!



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

- als erstes Vergleich auf  $> 25 \text{ °C} \Rightarrow \text{ROT}$
- als zweites Vergleich auf  $> 21 \text{ °C} \Rightarrow \text{GRÜN}$
- ansonsten  $\Rightarrow \text{BLAU}$

Vor dem Umbau spielen wir das hier einmal durch :



# Vergleich auf "Papier"

gemessener Wert : **30**

- als erstes Vergleich auf  $> 25 \text{ }^\circ\text{C}$  : **WAHR**  $\Rightarrow$  ROT und Ende
- Ergebnis : **ROT**



gemessener Wert : **24**

- als erstes Vergleich auf  $> 25 \text{ }^\circ\text{C}$  : **FALSCH**  $\Rightarrow$  Weiter
- als zweites Vergleich auf  $> 21 \text{ }^\circ\text{C}$  : **WAHR**  $\Rightarrow$  GRÜN und Ende
- Ergebnis : **GRÜN**



# Vergleich auf "Papier"

gemessener Wert : **19**

- als erstes Vergleich auf  $> 25 \text{ }^{\circ}\text{C}$  : **FALSCH**  $\Rightarrow$  Weiter
- als zweites Vergleich auf  $> 21 \text{ }^{\circ}\text{C}$  : **FALSCH**  $\Rightarrow$  Weiter
- ansonsten  $\Rightarrow$  BLAU
- Ergebnis : **BLAU**



# Neu zusammensetzen

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

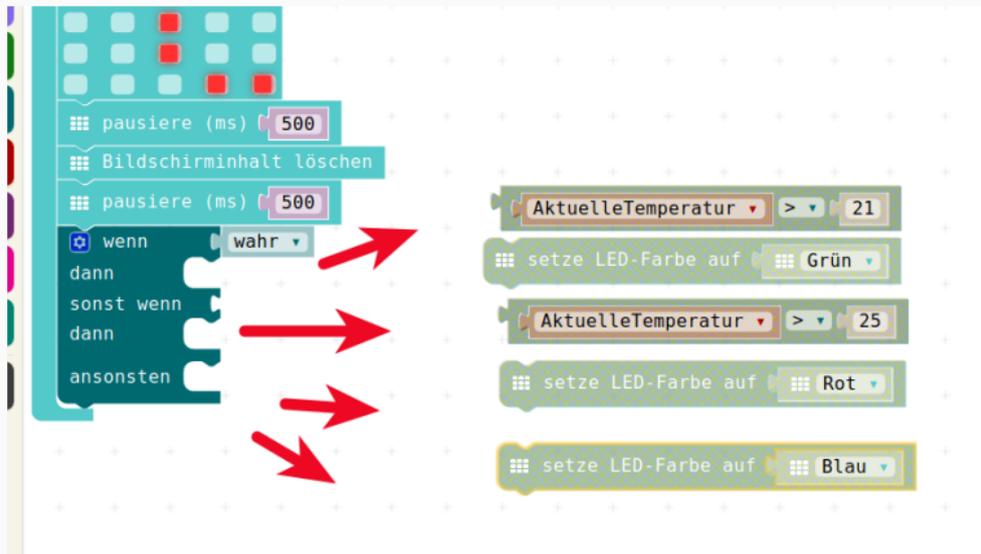


Figure 7: If Else Frei Geraeumt



# Neu zusammensetzen

und setzen es wie angedacht wieder zusammen.

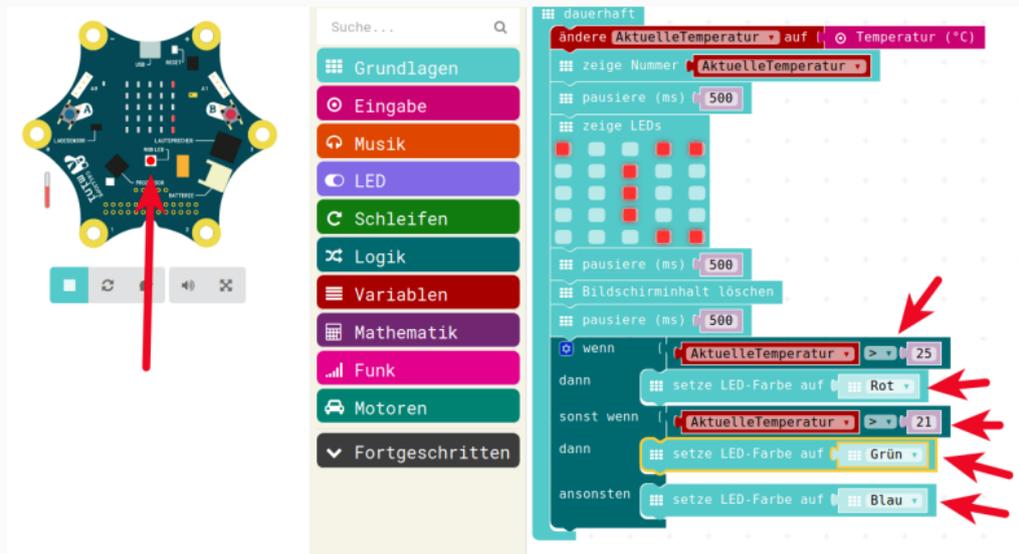


Figure 8: Neue Struktur Und Geht

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



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



## 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) {
    basic.setLedColor(Colors.Red)
  }
})
```



Für alle Texte und Bilder auf dieser Seite gilt:

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

