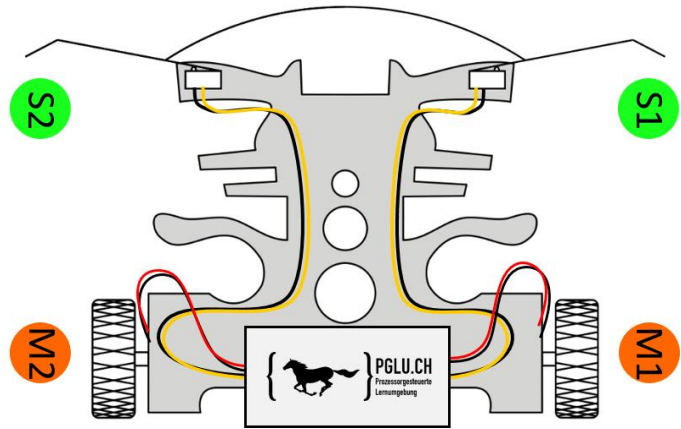


# 1



**Lösung** Beschreibe das Auto und das was es tut!

Lese und verstehe diesen Text, damit es dir leichter fällt, das Auto und seine Funktionen zu programmieren!

Weiterführende Google Keywords:

- Robotik
- Robot Competition/Contest
- Line follower

**Komponenten:** Das Selbstfahrende Auto besitzt 2 Motoren und 2 Sensoren.

**Mechanik:** Die Sensoren bestehen aus einem federnden Bügel an der Front sowie zwei seitlichen Antennen, welche ein Hindernis erkennen können.

**Elektronik:** Der federnde Bügel ist an 2 Schaltern befestigt, welche jede Berührung von vorne oder den Seiten an die Platine melden.

**Steuerung:** Ein Sensor steuert jeweils seinen gegenüberliegenden Motor. Trifft z.B. S1 auf ein Hindernis, dreht M2 während einer halben Sekunde rückwärts.

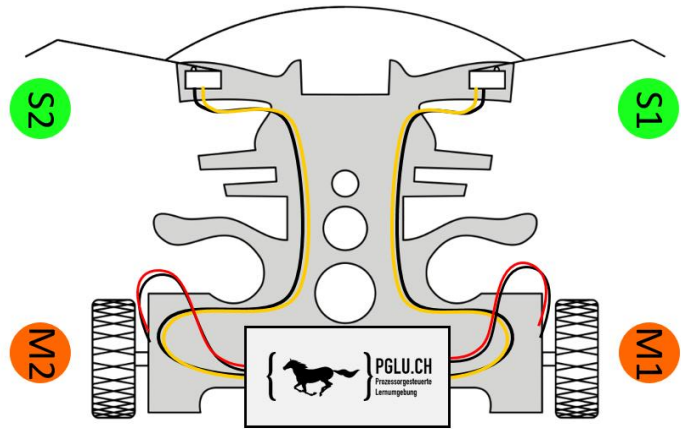
**Bewegung:** Berührt das Fahrzeug ein Hindernis, dreht es an Ort wieder davon weg.

**Timer:** Um dieses Abdrehen zu verlängern, wird nach jeder Berührung das Programm für eine kurze Zeit pausiert. Das heisst, dass die Anweisung «fahre wieder geradeaus» um eine halbe Sekunde hinausgezögert wird!

Mit der Länge dieser Zeit kann der Winkel des Abdrehens eingestellt werden.

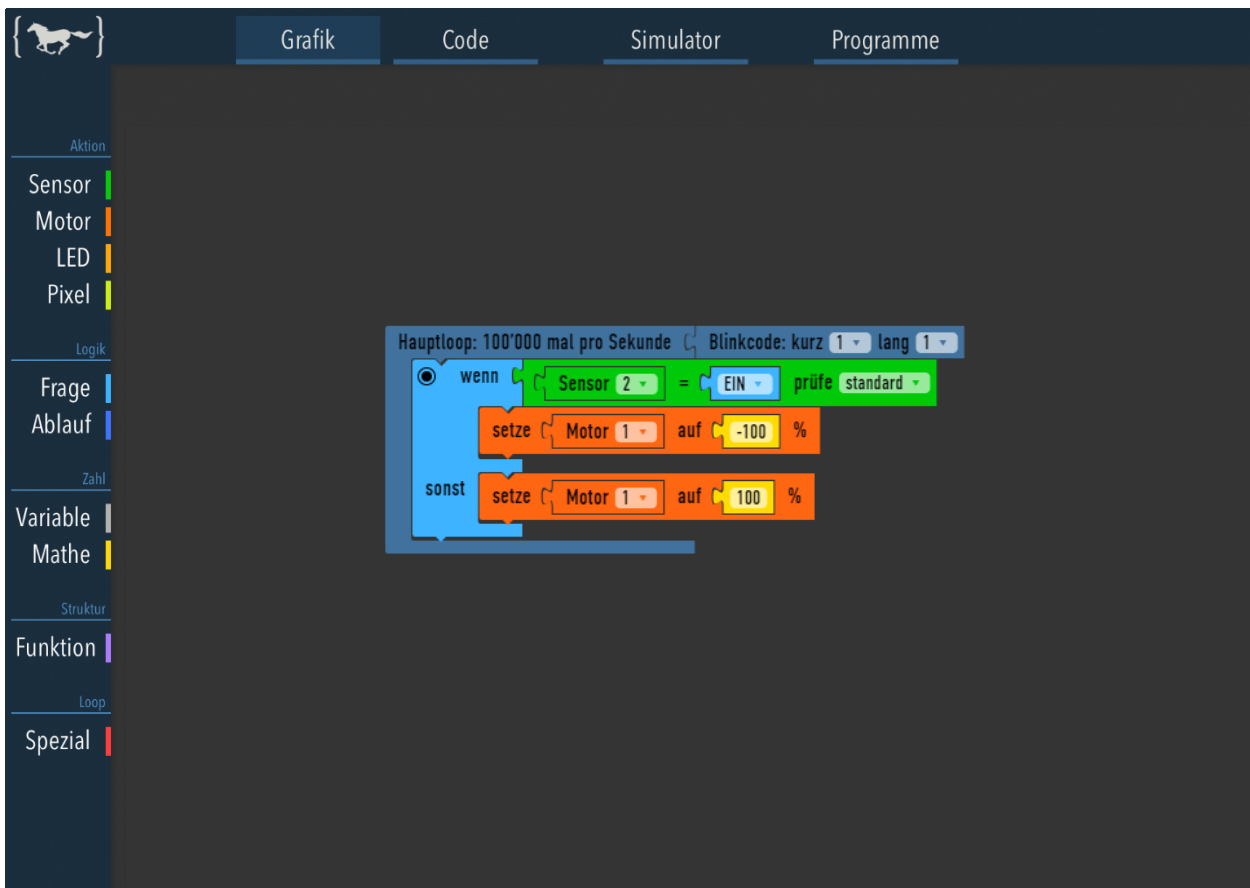
In Erweiterungsaufgabe 5 lernst du, dass diese einfache Art, die Drehung zu verlängern auch Nachteile mit sich bringt!

2

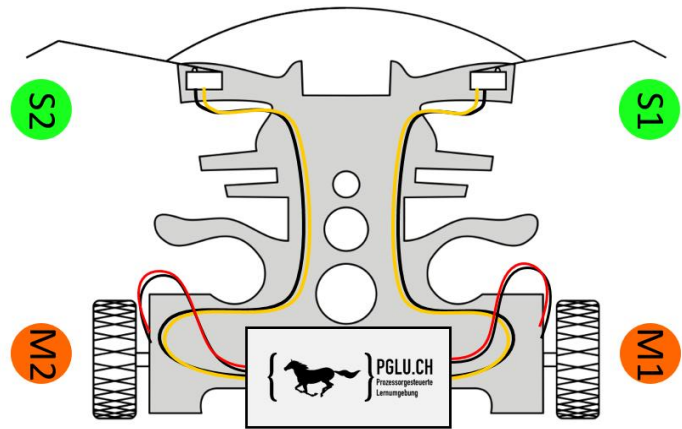


**Lösung** Ein einzelner Motor vorwärts und bei Sensorberührung rückwärts

Bei Programmieraufgaben gibt es immer viele mögliche Lösungen!  
Achte darauf, immer die eleganteste zu suchen. Das hilft dir, komplizierte Programme besser zu verstehen oder zu erweitern!



3



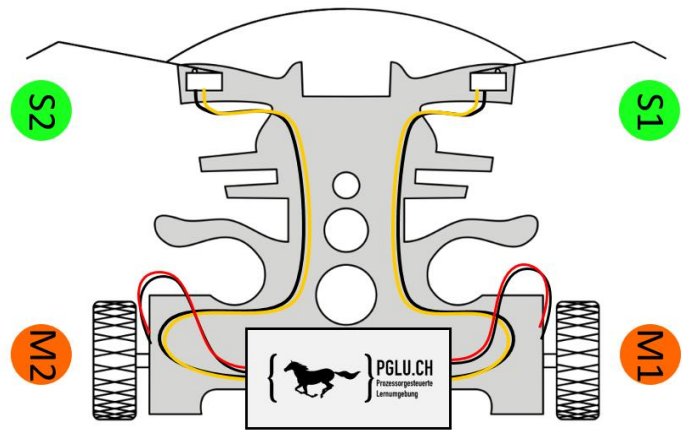
**Lösung** Ein einzelner Motor vorwärts und bei Sensorberührung eine halbe Sekunde rückwärts

Öffne den Simulator und teste dein Programm: drücke auf den Taster von Sensor 1 in der oberen Leiste und beobachte den Ausgang M1 in der unteren Leiste.

Auf diese Weise kannst du das Programm jederzeit testen. Du kannst es während laufender Simulation auch verändern!

The screenshot shows the PGLU.CH simulator interface. The top bar has tabs for 'Grafik', 'Code', 'Simulator', and 'Programme'. The 'Simulator' tab is active. On the left, there is a sidebar with various components: 'Aktion' (Action), 'Sensor', 'Motor', 'LED', 'Pixel', 'Logik' (Logic), 'Frage' (Question), 'Ablauf' (Flow), 'Zahl' (Number), 'Variable', 'Mathe' (Math), 'Struktur' (Structure), 'Funktion' (Function), 'Loop', and 'Spezial' (Special). The main area displays a block-based programming code. The code is a 'Hauptloop: 100'000 mal pro Sekunde' (Main loop: 100,000 times per second). It contains a 'wenn' (if) block with the condition 'Sensor 2 = EIN' (Sensor 2 = ON) and 'prüfe standard' (check standard). If true, it sets 'Motor 1' to '-100 %' and pauses for '500 ms'. Otherwise, it sets 'Motor 1' to '100 %'. The 'Blinkcode' is set to 'kurz 1' (short 1) and 'lang 1' (long 1). Below the code, there is a 'Simulationsgeschwindigkeit' (Simulation speed) slider. At the bottom, there is a status bar showing the status of various components: 'M1' is set to '100' (circled in yellow), 'M2' is '0', 'L1' is '0', 'L2' is '0', 'L3' is '0', and 'L4' is '0'.

# 4



**Drei Lösungen Erweiterungsaufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung eine halbe Sekunde rückwärts, mit LED Blinker

Mit diesen drei Lösungen lernst du ein Programm zu vereinfachen und zu kürzen.

Programm 1 ist die Grundversion. In der Version 2 wird das Blinken in eine «Funktion» ausgelagert und in Version 3 kommt zusätzliche eine «Wiederhole-Schleife» zum Einsatz.

## Grundversion

The screenshot shows a block-based programming interface with a sidebar on the left containing categories: Aktion, Sensor, Motor, LED, Pixel, Logik, Frage, Ablauf, Zahl, Variable, Mathe, Struktur, Funktion, Loop, and Spezial. The main workspace displays a program titled 'Hauptloop: 100'000 mal pro Sekunde' and 'Blinkcode: kurz 1 lang 1'. The program logic is as follows:

```

Hauptloop: 100'000 mal pro Sekunde
  wenn Sensor 2 = EIN prüfe standard
    setze Motor 1 auf -100 %
    setze LED 1 auf EIN
    pausiere 100 ms
    setze LED 1 auf AUS
    pausiere 100 ms
    setze LED 1 auf EIN
    pausiere 100 ms
    setze LED 1 auf AUS
    pausiere 100 ms
    setze LED 1 auf EIN
    pausiere 100 ms
    setze LED 1 auf AUS
  sonst
    setze Motor 1 auf 100 %
  
```

## Version 2 mit Funktion

The image shows the Scratch code editor interface. The left sidebar has tabs for 'Aktion', 'Logik', 'Zahl', 'Variable', 'Mathe', 'Struktur', 'Funktion', 'Loop', and 'Spezial'. The 'Funktion' tab is selected. The main workspace shows a 'Hauptloop: 100'000 mal pro Sekunde' block. Inside the loop, there is a 'wenn' block with the condition 'Sensor 2 = EIN' and 'prüfe standard'. The 'wenn' block has two branches: 'dann' and 'sonst'. The 'dann' branch contains a 'setze Motor 1 auf -100 %' block and a 'Blinke mit LED 500ms' function block. The 'sonst' branch contains a 'setze Motor 1 auf 100 %' block. The 'Blinke mit LED 500ms' function block is highlighted with a yellow dot. A yellow line connects this dot to the 'Funktion' tab in the sidebar. Another yellow line connects the 'Funktion' tab to the 'Blinke mit LED 500ms' function block definition on the right. The function definition is titled 'Name Blinke mit LED 500ms' and contains a sequence of blocks: 'setze LED 1 auf EIN', 'pausiere 100 ms', 'setze LED 1 auf AUS', 'pausiere 100 ms', 'setze LED 1 auf EIN', 'pausiere 100 ms', 'setze LED 1 auf AUS', 'pausiere 100 ms', 'setze LED 1 auf EIN', 'pausiere 100 ms', and 'setze LED 1 auf AUS'.

Hauptloop: 100'000 mal pro Sekunde Blinkcode: kurz 1 lang 1

wenn Sensor 2 = EIN prüfe standard

sonst

setze Motor 1 auf -100 %

Blinke mit LED 500ms

setze Motor 1 auf 100 %

Name Blinke mit LED 500ms

setze LED 1 auf EIN

pausiere 100 ms

setze LED 1 auf AUS

pausiere 100 ms

setze LED 1 auf EIN

pausiere 100 ms

setze LED 1 auf AUS

pausiere 100 ms

setze LED 1 auf EIN

pausiere 100 ms

setze LED 1 auf AUS

>Zuerst Funktionsklammer in Menu holen

>Funktion einen Namen geben

>Funktionsblock holen und in Hauptloop einsetzen

## Version 3 mit Wiederhole-Schleife

The image shows the Scratch code editor interface. The left sidebar has tabs for 'Aktion', 'Logik', 'Zahl', 'Variable', 'Mathe', 'Struktur', 'Funktion', 'Loop', and 'Spezial'. The 'Funktion' tab is selected. The main workspace shows a 'Hauptloop: 100'000 mal pro Sekunde' block. Inside the loop, there is a 'wenn' block with the condition 'Sensor 2 = EIN' and 'prüfe standard'. The 'wenn' block has two branches: 'dann' and 'sonst'. The 'dann' branch contains a 'setze Motor 1 auf -100 %' block and a 'Blinke LED 500ms' function block. The 'sonst' branch contains a 'setze Motor 1 auf 100 %' block. The 'Blinke LED 500ms' function block is highlighted with a yellow dot. A yellow line connects this dot to the 'Funktion' tab in the sidebar. Another yellow line connects the 'Funktion' tab to the 'Blinke LED 500ms' function block definition on the right. The function definition is titled 'Name Blinke LED 500ms' and contains a 'wiederhole 50 mal' loop block. Inside the loop, there are four blocks: 'setze LED 1 auf EIN', 'pausiere 5 ms', 'setze LED 1 auf AUS', and 'pausiere 5 ms'.

Hauptloop: 100'000 mal pro Sekunde Blinkcode: kurz 1 lang 1

wenn Sensor 2 = EIN prüfe standard

sonst

setze Motor 1 auf -100 %

Blinke LED 500ms

setze Motor 1 auf 100 %

Name Blinke LED 500ms

wiederhole 50 mal

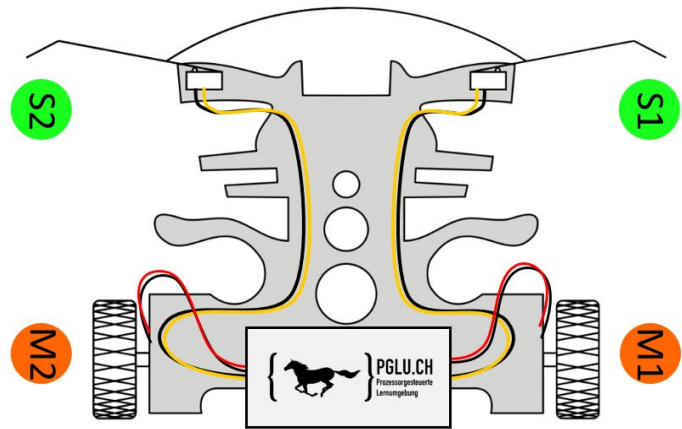
setze LED 1 auf EIN

pausiere 5 ms

setze LED 1 auf AUS

pausiere 5 ms

5



**Lösung Erweiterungsaufgabe** Ein einzelner Motor vorwärts und bei Sensorberührung eine Sekunde rückwärts ohne Programmpause: so würde ein Profi programmieren! Siehe auch Variante 3!

Eine Variable «Timer1» zählt im Millisekundentakt von 0 bis unendlich hoch. Ist «Timer1» kleiner als 500 dreht M1 rückwärts (M1=-100%).

Sobald Sensor 2 ein Hindernis spürt, wird «Timer1» auf 0 gesetzt und ist somit eine halbe Sekunde lang kleiner als 500.

**Programmcode:**

**Hauptloop: 100'000 mal pro Sekunde** (Blinkcode: kurz 1 lang 1)

```

schreibe Timer1 = Timer1 + 1
wenn Sensor 1 = EIN prüfe standard
  schreibe Timer1 = 0
  wenn Timer1 < 500
    setze Motor 2 auf -100 %
  sonst
    setze Motor 2 auf 100 %
  pausiere 1 ms

```

**Vor Hauptloop: 1x**

```

schreibe Timer1 = 500
schreibe Timer2 = 500

```

**Testblock:**

```

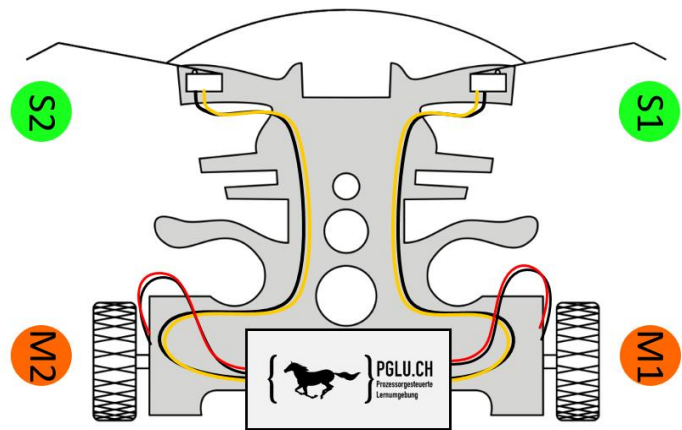
setze LED 1 auf Timer1 %

```

Für einen Test im Simulator, setze hier 20 statt 500 ein, da der Simulator langsamer arbeitet, als die PGLU-Platine

Setze diesen Block ein, wenn du das Programm im Simulator testen willst. Du kannst dann an Ausgang L1 den Wert von «Timer» beobachten solange er unter 100 ist.

# 6



## Lösung Das Hauptprogramm «Selbstfahrendes Auto»

Gratuliere! Dies ist das vollständige Programm, das dein Auto zum Fahren bringt. Es basiert auf den Programmübungen 1-3, die du gemacht hast.

Spendiere deinem Auto noch zusätzliche Funktionen, indem du die Übungen 4-5 einbaust und LEDs blinken lässt oder eine Version ohne Pausieren-Block entwickelst.

Teste auch die Varianten 1-3 im separaten Dokument.

The screenshot shows the PGLU.CH programming interface. The top bar has tabs for 'Grafik', 'Code', 'Simulator', and 'Programme'. The left sidebar contains a list of components: 'Sensor', 'Motor', 'LED', 'Pixel', 'Logik', 'Frage', 'Ablauf', 'Zahl', 'Variable', 'Mathe', 'Struktur', 'Funktion', 'Loop', and 'Spezial'. The main area displays a block-based code editor for a 'Hauptloop: 100'000 mal pro Sekunde'. The code is as follows:

```

Hauptloop: 100'000 mal pro Sekunde
  wenn Sensor 2 = EIN prüfe standard
    setze Motor 1 auf -100 %
    pausiere 500 ms
  sonst
    setze Motor 1 auf 100 %
  wenn Sensor 1 = EIN prüfe standard
    setze Motor 2 auf -100 %
    pausiere 500 ms
  sonst
    setze Motor 2 auf 100 %
  
```