Handreichung PhC

1. Kurzvorstellung

Bei dem Projekt zum Thema „Physical Computing“ handelt es sich um einen Zimmergärtner, der die Feuchtigkeit im Boden misst und bei Bedarf eine Pflanze gießt.

2. Einordnung in die Lehrpläne

Klassenstufe 8

Lernbereich 1: Algorithmen	**9 Ustd.**
Kennen des Algorithmusbegriffes Eigenschaften	 → Kl. 7, LB 3 → MA, Kl. 8, LB 3
Übertragen der Eigenschaften von Algorithmen auf Sachverhalte aus der Erfahrungswelt der Schüler	Soziale Medien, Smarthome, In-App-Käufe, Werbung ⇒ Medienbildung
Beherrschen der algorithmischen Lösung einfacher Problemstellungen in einer didaktisch reduzierten Programmierumgebung	Blocksprachen, visuelle Programmier-umgebungen
- Darstellungsformen von Algorithmen	Blockdarstellung, verbale Beschreibung
- Sequenz, Verzweigung und Wiederholung	Zählschleife, kopfgesteuerte oder fußgesteuerte Schleife
Kennen der Bedeutung von Algorithmen im gesellschaftlichen Kontext	

Klassenstufe 10

Lernbereich 1: Algorithmen	**11 Ustd.**
Beherrschen der Implementierung der algorithmischen Grundstrukturen	grundlegende, einfache Algorithmen → Kl. 8, LB 1 Syntax und Semantik → LB 2
- Datentypen · Zahlen · Zeichenketten · Wahrheitswerte	
- Variablenzuweisungen	
- verknüpfte Bedingungen	Verkettung durch logische Operatoren
Kennen des Prinzips der Modularisierung	Nutzen von Unterprogrammen und Bibliotheken
Übertragen der Kenntnisse zu Algorithmen auf maschinelle Entscheidungsprozesse	autonomes Fahren, Gesichtserkennung, Wahlcomputer → KL. 9, LB 2 → ETH, Kl. 10, LB 1

Sophie Burchhardt

3. Lernziele für Klasse 8/ Lernbereich 1: Algorithmen

Lernzielkategorien	**Lernziele nach *Anderson et al (2001)***
Kognitive Lernziele	Stufe 1: Erinnern Die Schüler*innen kennen den Algorithmus Begriff und dessen Eigenschaften. Stufe 3: Anwenden Die Schüler*innen übertragen die Eigenschaften von Algorithmen auf Sachverhalte aus ihrer Erfahrungswelt.
Psychomotorische Lernziele	Stufe 2: Verstehen Die Schüler*innen beherrschen algorithmische Lösungen einfacher Problemstellungen in einer didaktisch reduzierten Programmierumgebung. Die Schüler*innen beherrschen Darstellungsformen von Algorithmen, sowie die Prinzipien der Sequenz, der Verzweigung und der Wiederholung.
Affektive Lernziele	Stufe 1: Erinnern Die Schüler*innen kennen die Bedeutung von Algorithmen im gesellschaftlichen Kontext. Die Schüler*innen beurteilen den Nutzen von praktischen Informatiksystemen auf dem Gebiet der Automatisierung. Die Schüler*innen schätzen ihre Problemlösestrategie und deren Umsetzung innerhalb einer Schaltung kritisch ein.

4. Voraussetzungen

4.1 Materielle Voraussetzungen

- 1x Micro:bit inklusive Batterien und Verbindungskabel
- 1x Servo
- 1x Feuchtigkeitssensor von Funduino
- 1x Strohhalm + Alufolie
- 2x Gummibänder
- 6x Kabel (3 Kabel mit Doppelstecker und 3 Kabel mit Einfachstecker)
- 1x Steckbrett + micro:bit-Halterung
- 1x Wassergefäß
- Legosteine
- Klebeband
- Schere

Bei der Auflistung handelt es sich um die Materialien pro Ausführung. Es muss darauf geachtet werden, dass für alle Schüler genügend Material zur Verfügung steht sowie Ersatzmaterialien wie Gummibänder oder Strohalme.

4.2 Technische Voraussetzungen

- Klassensatz micro:bit
- Internetzugang für die Programmierung mit https://makecode.microbit.org/ oder die vorinstallierte MakeCode App für micro:bit
- Ausreichend Verbindungskabel zur Datenübertragung des Programms auf den micro:bit

4.3 Fachliche Voraussetzungen

- Die SuS verfügen über Grundlagen des Algorithmus-Begriffs inklusive der Eigenschaften, der Darstellungsformen und der Grenzen der Algorithmierbarkeit
- Die SuS verfügen über Grundlagen der Programmierung inklusive einfacher Datentypen, algorithmische Grundstrukturen
- Anwenden der Phasen der Problemlöseprozess inklusive Problemanalyse, Modellierung der Problemlösung, Implementierung in einer Programmierumgebung sowie Test und Fehlerbewegung
- Die SuS verfügen über Vorkenntnisse für den Umgang mit der MakeCode Programmierumgebung
- Kritische Bewertung eigener Problemlösestrategien durch die SuS
- Die SuS verfügen über Fähigkeiten zum Aufbau von technischen Schaltungen

5. Kurzdarstellung

Ziel des Projektes ist es einen Zimmergärtner zu erschaffen, welcher die Bodenfeuchtigkeit eines Pflanzentopfes überwacht. Unterschreitet der Boden eine gewisse Feuchtigkeit, soll eine Warnung ausgegeben und über eine selbstgebaute kleine Wasserschaufel der Boden etwas bewässert werden.

Grundidee:

Auf Grundlage von chemischem und physikalischem Wissen ist bekannt, dass reines Wasser selbst nichtleitend ist. Aber wenn im Wasser Nährstoffe wie Mineralien oder Salze gelöst sind, dann ist Wasser durchaus in der Lage elektrischen Strom zu leiten. Je höher der Wert von den im Wasser gelösten Nährstoffen ist, desto höher ist der Leitwert, indem der elektrische Widerstand gleichzeitig gesenkt wird. Der Boden, der mit Wasser und Mineralstoffen angereichert ist, fungiert also wie ein variabler Widerstand in einer elektronischen Schaltung. Deshalb können wir schließen, dass je mehr Bodenfeuchtigkeit vorhanden ist, desto geringer ist der Widerstand im Boden und somit, desto höher ist der Wert, den wir an einem Pinn messen können. Mit diesem Wissen können wir den Zimmergärtner mit einem Feuchtigkeitsmesser von Funduino ausstatten, der auf dieser Grundlage Feuchtigkeit misst.

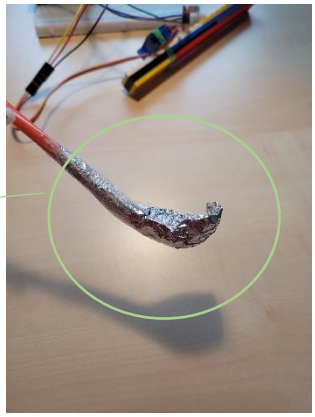
Aufbau:

a) Bau der Wasserpumpe/Wasserschaufel

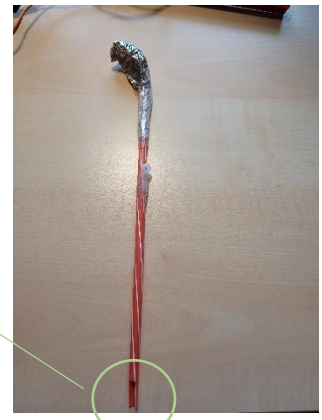
Für die Wasserpumpe wird benötigt:

- 1x Wassergefäß
- Legobausteine
- 2x Gummibänder
- 1x Servo
- 1x Plastikstrohhalm
- Klebeband
- Alufolie
- Schere

Mit Alufolie wird ein Löffel geformt und dieser mit Klebeband an einem Ende des Plastikstrohhalmes befestigt.



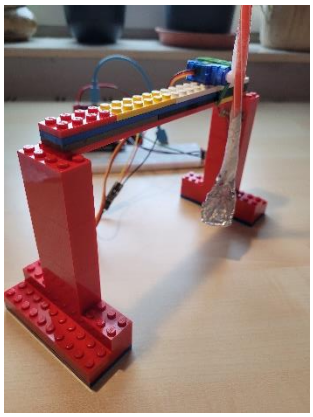
Am anderen Ende des Strohhalms wird mit der Schere eine kleine Ecke weggeschnitten, um den Wasserabfluss zu erleichtern.



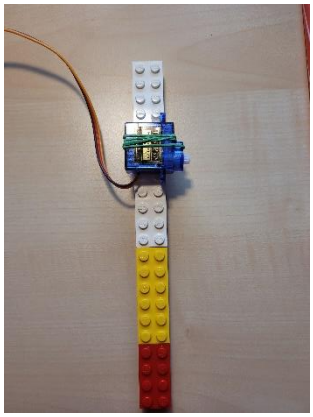
Ungefähr mittig des Strohhalms wird seitlich der bewegliche Stift des Servos mit Klebeband befestigt. Die genaue Stelle sollte allerdings im Einzelfall geprüft werden und hängt auch von der Tiefe und Größe des Wassergefäßes ab.

b) Konstruktion der Brücke und Verbindung des Servos mit der Brücke und der Wasserschaufel

Aus Lego wird eine Art „Brücke“ gebaut, die über das verwendete Wassergefäß passt.



Der Servo wird mit zwei Gummibändern auf der Brücke fixiert. Der Strohhalm kann dann am Servo angebracht werden.

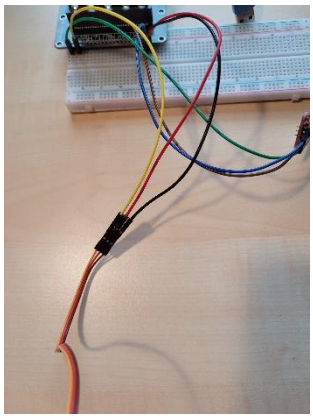


Der Servo wird wie folgt angeschlossen:

Rechtes Kabel (Servokabel: braun, Verbindungskabel: schwarz): GROUND

Mittleres Kabel (Servokabel: rot, Verbindungskabel: rot): 3V

Linkes Kabel (Servokabel: orange, Verbindungskabel: gelb): Pin P2



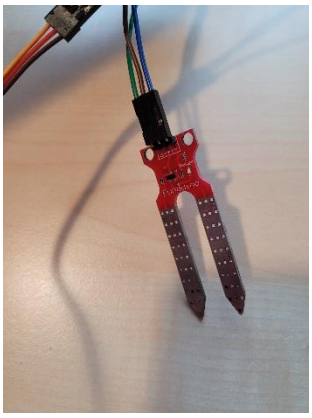
c) Anschluss des Feuchtigkeitsmessers

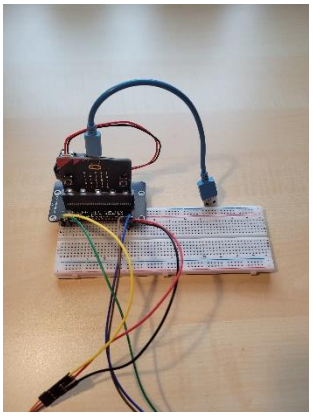
Der Feuchtigkeitssensor wird über drei Kabel mit dem micro:bit verbunden.

Rechtes Kabel (blau): Ground

Mittleres Kabel (braun): 3V

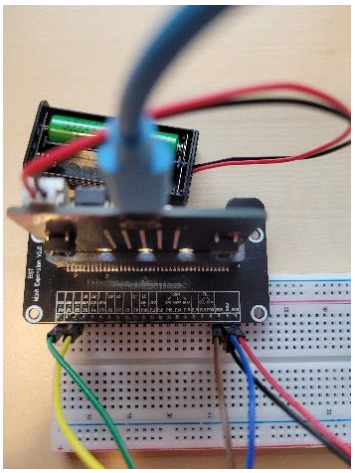
Linkes Kabel (grün): Pin P0





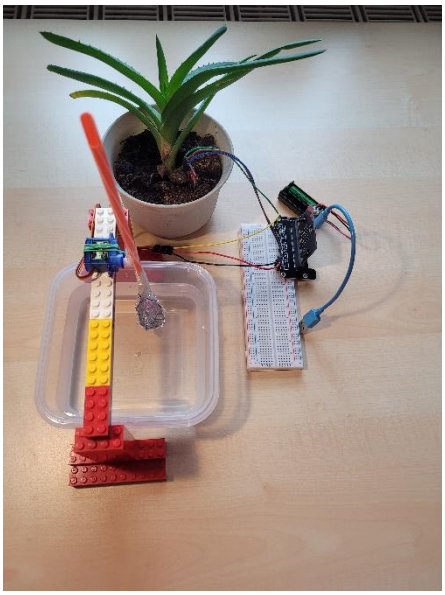
Anschließend wird der Feuchtigkeitssensor in den Topf der Zimmerpflanze gesteckt.

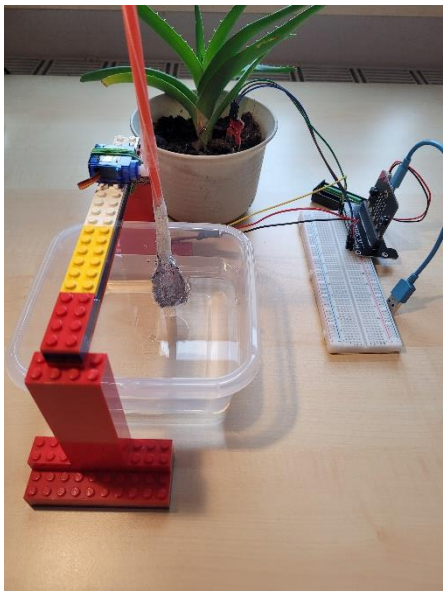
Der micro:bit wird über die Halterung auf dem Steckbrett fixiert. Alle Kabel werden wie oben beschrieben in das Steckbrett gesteckt.



d) Gesamtanordnung

Der Zimmergärtner sollte nach Ausführung folgender Schritte so aussehen:





Programmcode Teil 1: Messen der Bodenfeuchtigkeit

- Erstellung der Variable Bodenfeuchtigkeit
- Zuweisung des analogen Wertes von Pin P0 der Variable Bodenfeuchtigkeit
  - ➔ Zur besseren Vorstellung wird über den „verteile“ – Block noch der Wert von 0 auf 100 herunterskaliert
  - ➔ Normalerweise liegt der Höchstwert des Wertes ohne Skalierung bei 1023. Durch Messungen ist aber klar, dass der Maximalwert des Feuchtigkeitssensors von Funduino bei 800 liegt.
- Ausgabe des aktuellen Messwertes bei Druck der Taste A als abgerundete Zahl, da sonst im Grenzfall eher gegossen werden würde

Programmcode Teil 2: Gießen bei Trockenheit

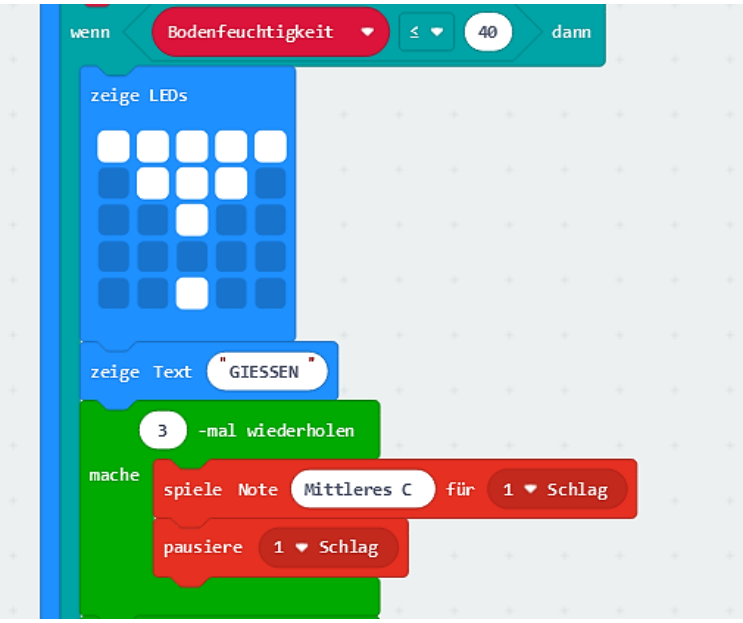
- Falls der Wert von 40 unterschritten wird, wird wenig Strom geleitet und somit befindet sich wenig Wasser mit Nährstoffen im Boden
- Warnung wird ausgegeben: Text „GIESSEN“, akustisches Signal (Piepton) sowie ein optisches Signal (!)
- Giesen der Pflanze in dem der Servo-Motor den Strohhalm bewegt, der wiederum Wasser aus einem Behälter in das Pflanzgefäß befördert

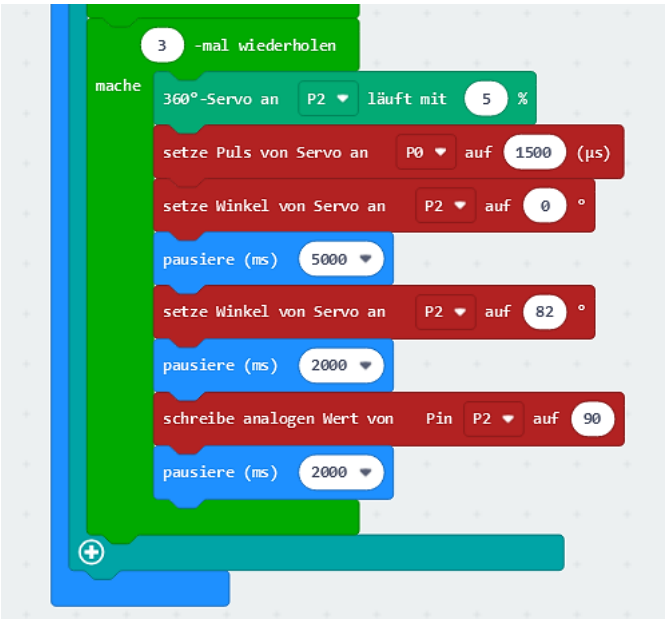
  → Änderung des Winkels des Servos und dadurch Änderung der Position des Strohhalms

  → Wiederholung dieses Vorgangs dreimal, damit der Boden ausreichend gewässert wird

  → Servo läuft lediglich mit 5%, damit zu schnelle Bewegungen der Wasserschaufel und damit ein Wasserverlust verhindert wird

  → der Puls des Servos wird auf 1500 µs gesetzt, dies ist eine Nachjustierung der Wasserschaufel, damit nach jedem Gießvorgang wieder genügend Wasser in die Schaufel laufen kann





Allgemeine Hinweise:

1. Der Feuchtigkeitssensor sollte nicht zu weit weg von der Stelle im Pflanztopf sein, an der das Wasser einfließt, um unnötiges Gießen zu vermeiden! Wenn sich der Sensor beispielsweise auf der gegenüberliegenden Seite der Gießstelle befindet, kann eine geringe Bodenfeuchtigkeit gemessen werden, obwohl schon ausreichend gegossen wurde,
2. Zum Spritzschutz sollte um die Brücke etwas Küchenpapier gelegt werden, welches kleine unvermeidbare Wasserspritzer aufsaugen kann!

Java Code des gesamten Programmes

```
let Bodenfeuchtigkeit = 0
input.onButtonPressed(Button.A, function () {
    Bodenfeuchtigkeit = Math.map(pins.analogReadPin(AnalogPin.P0), 0, 800, 0, 100)
    basic.showNumber(Math.floor(Bodenfeuchtigkeit))
    basic.clearScreen()
})
basic.forever(function () {
    basic.pause(7200)
    Bodenfeuchtigkeit = Math.map(pins.analogReadPin(AnalogPin.P0), 0, 800, 0, 100)
    if (Bodenfeuchtigkeit <= 40) {
        basic.showLeds(`
            # # # # #
            . # # # .
            . . # . .
            . . . . .
            . . # . .
            `)
        basic.showString("GIESSEN")
        for (let index = 0; index < 3; index++) {
            music.playTone(262, music.beat(BeatFraction.Whole))
            music.rest(music.beat(BeatFraction.Whole))
        }
        for (let index = 0; index < 3; index++) {
            servos.P2.run(5)
            pins.servoSetPulse(AnalogPin.P0, 1500)
            pins.servoWritePin(AnalogPin.P2, 0)
            basic.pause(5000)
            pins.servoWritePin(AnalogPin.P2, 82)
            basic.pause(2000)
            pins.analogWritePin(AnalogPin.P2, 90)
            basic.pause(2000)
        }
    }
})
```

Python Code des gesamten Programmes

```
Bodenfeuchtigkeit = 0
def on_button_pressed_a():
    global Bodenfeuchtigkeit
    Bodenfeuchtigkeit = Math.map(pins.analog_read_pin(AnalogPin.P0), 0, 800, 0, 100)
    basic.show_number(Math.floor(Bodenfeuchtigkeit))
    basic.clear_screen()
input.on_button_pressed(Button.A, on_button_pressed_a)

def on_forever():
    global Bodenfeuchtigkeit
    basic.pause(7200)
    Bodenfeuchtigkeit = Math.map(pins.analog_read_pin(AnalogPin.P0), 0, 800, 0, 100)
    if Bodenfeuchtigkeit <= 40:
        basic.show_leds("""
            # # # # #
                    . # # # .
                    . . # . .
                    . . . . .
                    . . # . .
        """)
        basic.show_string("GIESSEN")
        for index in range(3):
            music.play_tone(262, music.beat(BeatFraction.WHOLE))
            music.rest(music.beat(BeatFraction.WHOLE))
        for index2 in range(3):
            servos.P2.run(5)
            pins.servo_set_pulse(AnalogPin.P0, 1500)
            pins.servo_write_pin(AnalogPin.P2, 0)
            basic.pause(5000)
            pins.servo_write_pin(AnalogPin.P2, 82)
            basic.pause(2000)
            pins.analog_write_pin(AnalogPin.P2, 90)
            basic.pause(2000)
basic.forever(on_forever)
```