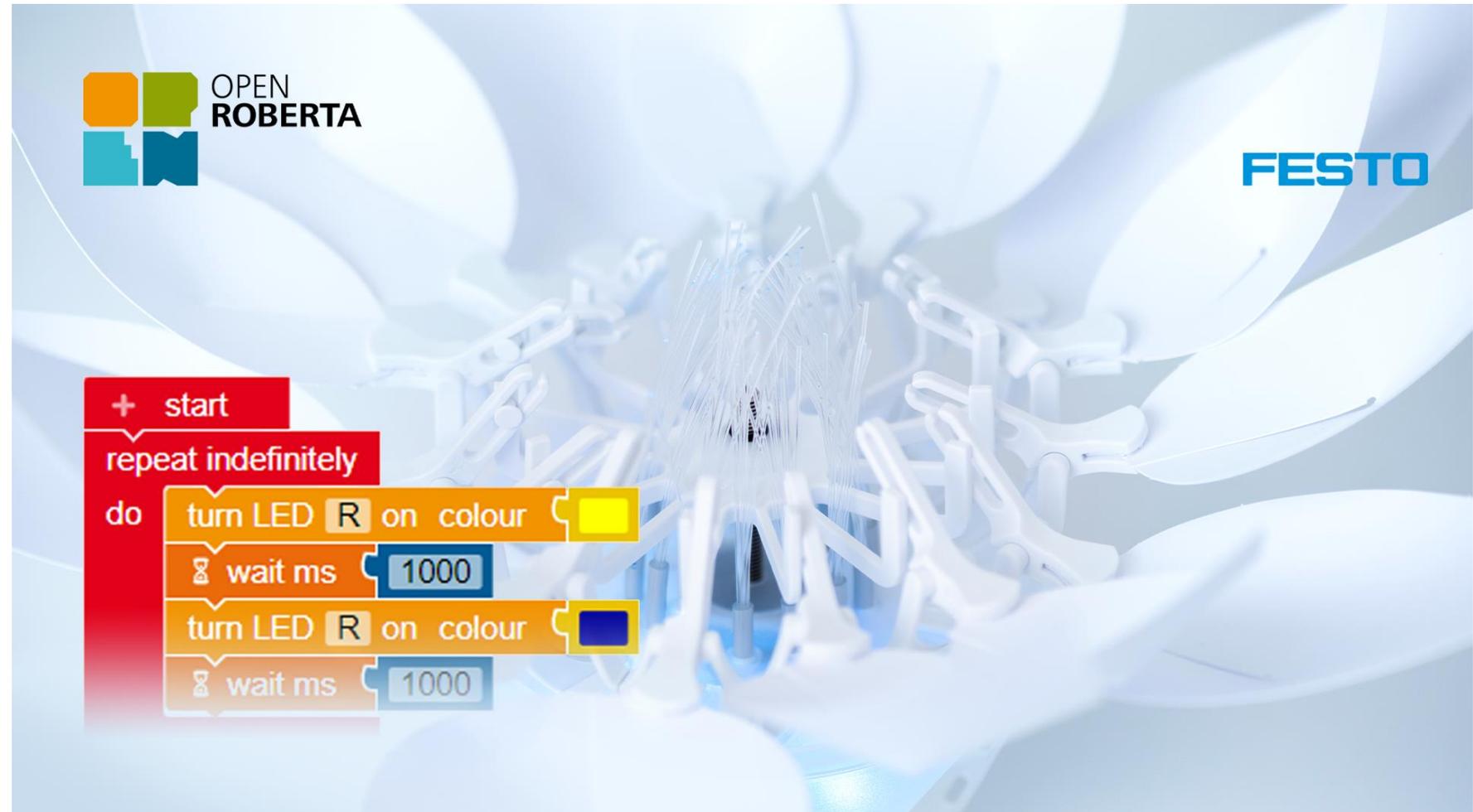


Bionics4Education:

Bionic Flower Courseware für Open Roberta



The image shows a Scratch script for a bionic flower project. The script is overlaid on a background image of a white bionic flower with a central fiber optic light display. The script starts with a red 'start' block, followed by a 'repeat indefinitely' loop. Inside the loop, there are four blocks: 'turn LED R on colour' with a yellow color picker, 'wait ms' with a value of 1000, 'turn LED R on colour' with a blue color picker, and another 'wait ms' with a value of 1000. The background image also features the 'OPEN ROBERTA' logo in the top left and the 'FESTO' logo in the top right.

OPEN ROBERTA

FESTO

```
+ start
repeat indefinitely
do
  turn LED R on colour [yellow]
  wait ms [1000]
  turn LED R on colour [blue]
  wait ms [1000]
```

Bionic Flower: Einführung in Coding mit Open Roberta

Struktur des Kapitels

1. Einführung Bionische Blume & biologische Hintergrundinformationen
2. Einführung Hardware Bionic Flower
3. Einführung Programmierung
4. Einleitung Open Roberta
5. DIY-Projekte



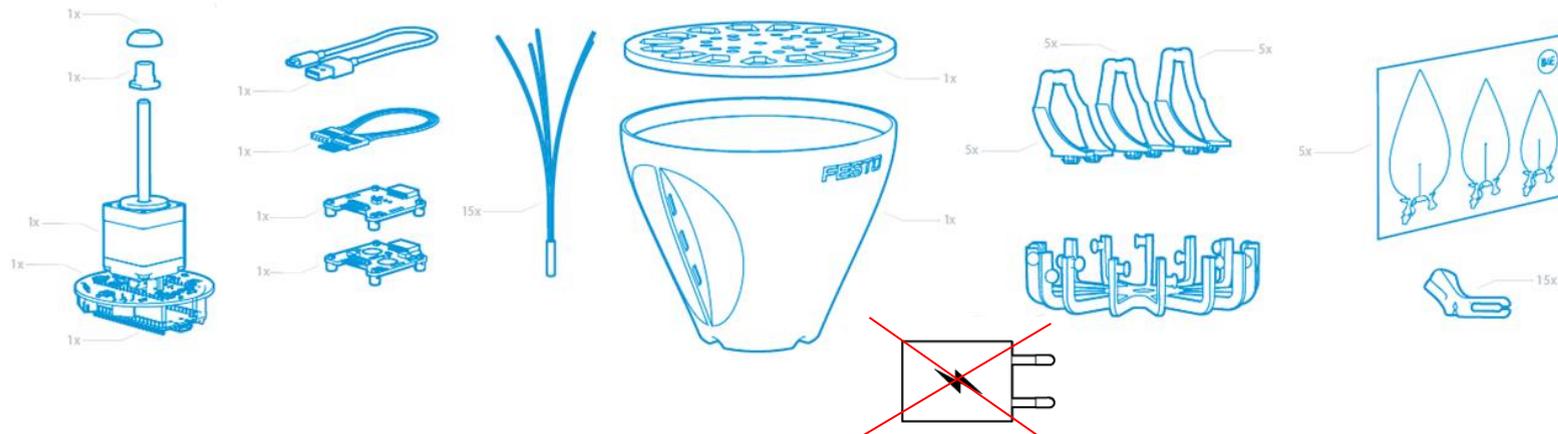
Bionics4Education - Die Bionic Flower

Was ist die Bionic Flower?

Die Bionic Flower ist eine sensorgesteuerte elektronische Blume, die von Festo Didactic entwickelt wurde. Sie können verschiedene Experimente mit der Bionic Flower durchführen und dabei sowohl biologische Phänomene als auch technische Fragestellungen entdecken. Du arbeitest zum Beispiel mit Sensoren oder einem Mikrocontroller und erfährst, wie sie die Bewegungen der Blütenblätter beeinflussen können.



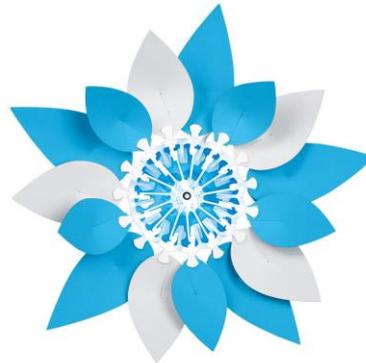
Experimentierset: Hardware-Komponenten



Bionics4Education - Die Bionic Flower

Eine von der Pflanzenwelt inspirierte Roboterblume

- 🌸 Festo Didactic hat sich bei der Entwicklung der Bionic Flower die Wirkmechanismen von Seerose und Mimose zum Vorbild genommen.
- 🌸 Diese Pflanzen haben eines gemeinsam: das Öffnen und Schließen ihrer Blütenblätter oder Blätter auf Grund äußerer Reize. Die Effekte dienen der Fortpflanzung und dem Schutz vor natürlichen Feinden.



Bionics4Education - Die Bionic Flower

Biologische Hintergrundinformationen

Natürliches Vorbild - Seerosen

Die Seerose ist eine Wasserpflanze, die in Gewässern verankert ist. Die einzeln stehenden Blüten haben eine spiralförmige Struktur, die durch Licht ausgelöst wird und so das Öffnen und Schließen der Blütenblätter ermöglicht. Tagblühende Seerosen öffnen ihre Blütenblätter am Morgen und locken so Insekten zur Bestäubung der Blüten an. Doch wie genau funktioniert die Bewegung der Blütenblätter?

Das Prinzip beruht auf Wachstumsunterschieden zwischen der Außenseite und der Innenseite der Blütenblätter. So wächst die Innenseite der Blütenblätter bei hellem Licht und die Blüte öffnet sich. Wenn nicht genug Licht vorhanden ist, wächst die Außenseite und die Blüte schließt sich.



Natürliches Vorbild - Mimosenpflanze

Die Blätter der Mimose klappen bei mechanischen Reizen wie Berührung oder Vibration in der Nähe nacheinander ein, und der Stängel senkt sich. Dies dient dem Schutz vor möglichen Fressfeinden. Die sukzessive Faltung der Fiederblättchen läuft wie eine Kettenreaktion ab (Dominoeffekt). Ein ausgeklügelter Mechanismus innerhalb der wassergefüllten Zellstrukturen in den Fiederblättchen sorgt dafür, dass bei einem äußeren Reiz Wasser aus der Zelle abgegeben wird. Das bedeutet, dass kein Wasser mehr von innen gegen die Zellwände drückt, die Zelle entspannt sich und die Fiedern ziehen sich zusammen.



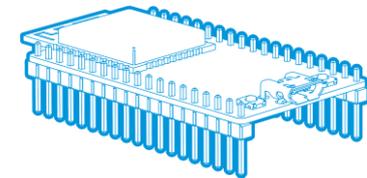
Bionic Flower - Hardware

Theorieteil . Elektronisches Bauteil - Mikrocontroller

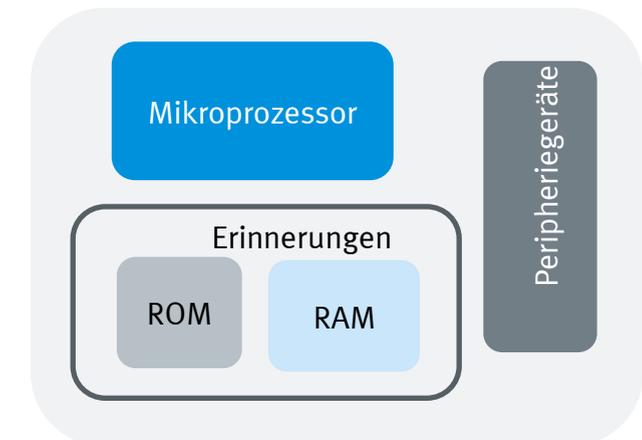
Die Bionic Flower wurde mit einem ESP32-Mikrocontroller gesteuert. Mikrocontroller sind in unserem täglichen Leben allgegenwärtig (Haushaltsgeräte, Roboter,..). Ein Mikrocontroller besteht aus 4 Teilen:

- **Mikroprozessor:** Er ist eine Komponente, die es ermöglicht, die in den Mikrocontroller eingebetteten Programme auszuführen, um die Informationsverarbeitung und das Senden von Befehlen zu unterstützen.
- **programmierbarer Speicher** (ROM : Read Only Memory): dieser Speicher enthält die Programmanweisungen.
- **Datenspeicher** (RAM : Random Access Memory): In diesem Speicher werden die für die Berechnungen erforderlichen temporären Daten gespeichert.
- **Peripheriegeräte:** Einige Anschlüsse sind für die Interaktion mit externen Geräten wie Sensoren, Aktoren, Displays usw. verfügbar. So kann der Mikrocontroller Daten empfangen, aber auch Befehle senden.

Der Mikrocontroller ist also in der Lage, ein Programm auszuführen, Informationen zu verarbeiten und mit anderen Komponenten zu kommunizieren. Der ESP32 ermöglicht die Steuerung des Öffnungsgrads der Blume und der Farbe der LEDs.



Mikrocontroller



Bionic Flower - Hardware

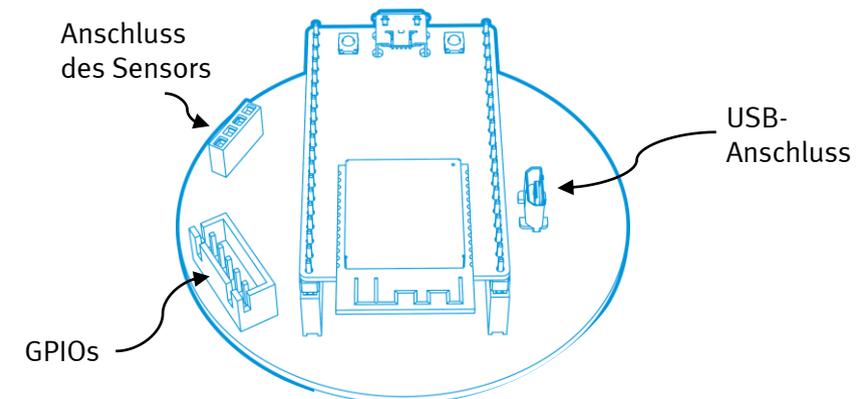
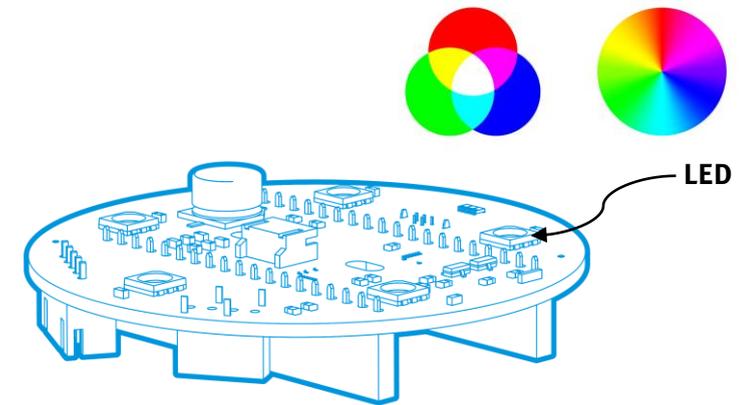
Theorieteil. Elektronische Bauteile - PCB

Für die Bionic Flower entwickelten die Teams von Festo eine Leiterplatte (PCB), auf der alle notwendigen Komponenten für das gute Funktionieren montiert sind.

Auf dieser Platine befinden sich 5 **LEDs**. Jede LED ist eine RGB-LED, das bedeutet, dass jede LED die Kombination von drei Farben (Rot, Grün, Blau) ist: Dank dieser drei Farben können alle Farben erzeugt werden.

Auf der anderen Seite der Leiterplatte befinden sich :

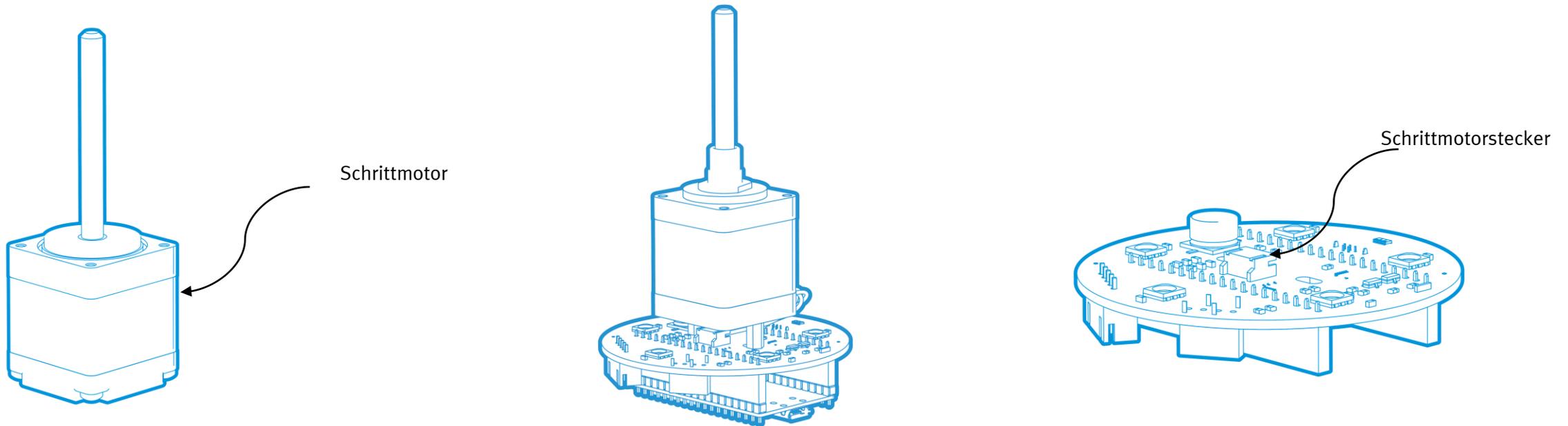
- **USB-Anschluss:** zur Stromversorgung und/oder zum Hochladen Ihres Programms auf den Mikrocontroller.
- **Sensor's port :** Es handelt sich um einen I²C-Port, an den die Sensoren der BionicFlower angeschlossen werden können.
- **GPIOs:** Einige GPIOs (General Purpose Input/Output) sind verfügbar.



Bionic Flower - Hardware

Theorie Teil. Elektronisches Bauteil - Schrittmotor

Ein Schrittmotor ermöglicht die Umwandlung eines elektrischen Impulses in eine Winkelbewegung. In der Bionic Flower ist der Schrittmotor mit einer Schraube verbunden, die es ermöglicht, die Winkelbewegung in eine Translationsbewegung umzuwandeln. Dies ermöglicht das Öffnen oder Schließen der Bionic Flower.

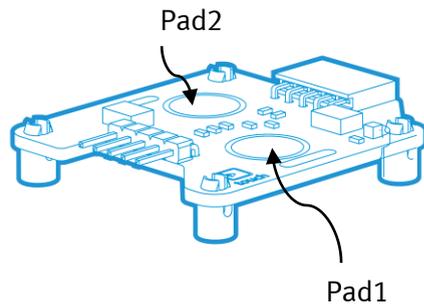


Bionic Flower - Hardware

Theorieteil. Elektronisches Bauteil - Sensoren

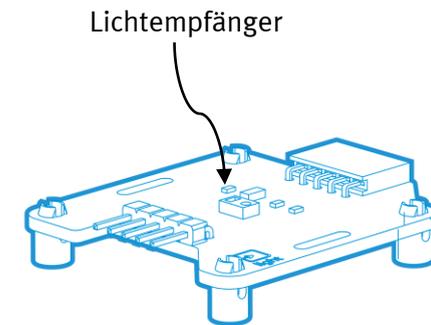
Berührungssensor

Der Berührungssensor besteht aus zwei Pads: Pad1 und Pad2. Der Sensor erkennt, ob eines der Pads oder beide Pads gedrückt werden. Dieser Sensor sendet boolesche Daten: wahr oder falsch.



Lichtsensord

Der Lichtsensor ermöglicht die Messung der Lichtstärke in Lux. Lux ist die Maßeinheit zur Messung der Lichtstärke, die pro Flächeneinheit empfangen wird.



Einführung - Programmierung

Was ist Programmieren?

Programmieren ist der Prozess des Schreibens von Computerprogrammen. Ein Programm ist eine Folge von Anweisungen, die der Computer ausführen muss.

Um mit dem Computer zu kommunizieren und somit Programme zu schreiben, wird eine **Programmiersprache** verwendet. Der Mensch kann nämlich die Maschinensprache nicht verstehen, sie ist viel zu komplex und unlesbar, es ist das **Binärprogramm**. Hierfür haben wir verschiedene Sprachen geschaffen. Die Programmiersprache ermöglicht es, den **Quellcode** zu erstellen: ein Programm, das von Menschen geschrieben und verstanden werden kann. So kann das gleiche Programm in verschiedenen Sprachen geschrieben werden (Java Script, Python, C, PHP,..). Je nach Anwendung sind jedoch einige Sprachen besser geeignet. Zum Beispiel sind die Sprachen C und C++ die in der Robotik am häufigsten verwendeten Programmiersprachen.

Die Programmierungsschritte sind:

- **Entwurf:** Bestimmung des Ziels des Programms und Analyse des funktionalen Aspekts des Programms, insbesondere die Entwicklung eines Algorithmus oder die Analyse der Dateneingaben/-ausgaben.
- **Implementierung:** Kodierung des Programms mit einer Programmiersprache nach Wahl, wodurch der Algorithmus in Quellcode umgewandelt wird.
- **Kompilierung:** Übersetzung des Quellcodes in Binärcode.

Sobald das Programm kompiliert ist, wird die Binärdatei über den USB-Anschluss an den Mikrocontroller gesendet. Open Roberta kann verwendet werden, um den Code zu schreiben, zu kompilieren und auf den Bionic Flower hochzuladen.



Einführung - Programmierung

Theorie-Teil. Roberta-Werkzeug öffnen

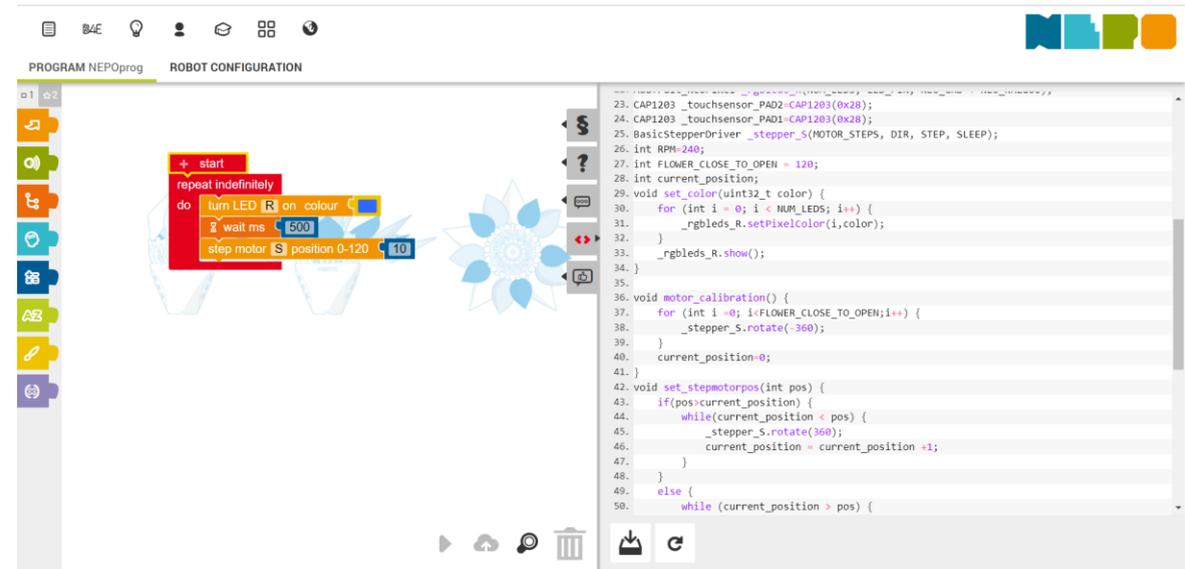
Open Roberta ist ein Web-Programmierwerkzeug und bietet Programmbausteine, um einigen Robotern, darunter auch dem Bionic Flower, Anweisungen zu geben.

Die Blocksprache ermöglicht es, das Programmieren auf einfache Weise zu erlernen. In der Tat machen es Blöcke möglich, sich keine Gedanken über die Einrückung, die Syntax, den Datentyp usw. des Programms zu machen. Wenn ein Block verwendet wird, wird er in die vereinfachte Sprache C umgeschrieben.

Wenn Sie neugierig sind, können Sie sich den Quellcode ansehen, indem Sie auf diesen Punkt klicken:



```
void set_color(uint32_t color) {
    for (int i = 0; i < NUM_LEDS; i++) {
        _rghblds_R.setPixelColor(i,color);
    }
    _rghblds_R.show();
}
```

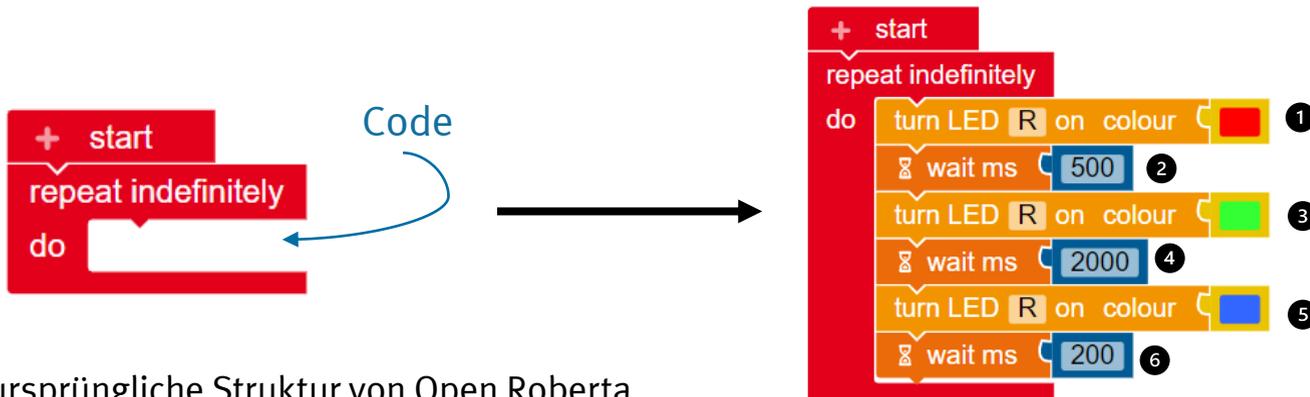


Roberta-Labor öffnen: <https://lab.open-roberta.org/>

Um Ihr Programm zu erstellen, können Sie verschiedene Anweisungen in Blockform auf der linken Seite des Bildschirms verwenden. Im Konfigurationsmenü des Roboters können Sie angeben, welche Hardware in Ihrem Programm verwendet wird: LEDs und Schrittmotor sind immer angegeben, aber Sie können Sensoren hinzufügen: Berührungs- und Lichtsensor.

Einführung - Programmierung

Theorie Teil. Anleitung : Struktur auf Open Roberta Lab



Die ursprüngliche Struktur von Open Roberta verwendet eine "unendliche" Schleife. Das bedeutet, dass der Code so lange wiederholt wird, wie der Mikrocontroller eingeschaltet ist.

Der Mikrocontroller führt die Befehle immer in der gleichen Reihenfolge aus. Der höchste Befehl wird zuerst ausgeführt, dann der nächste und so weiter. In diesem Beispiel wird also zuerst der Befehl 1 ausgeführt, dann der Befehl 2 usw.

Außerdem ist bei der Endlosschleife, solange der Mikrocontroller Strom hat, die Ausführungsreihenfolge 1,2,3,4,5,6,1,2,3,4,5,6,1,2, usw.

Einführung - Programmierung

Theorieteil. Anweisungen: Warten

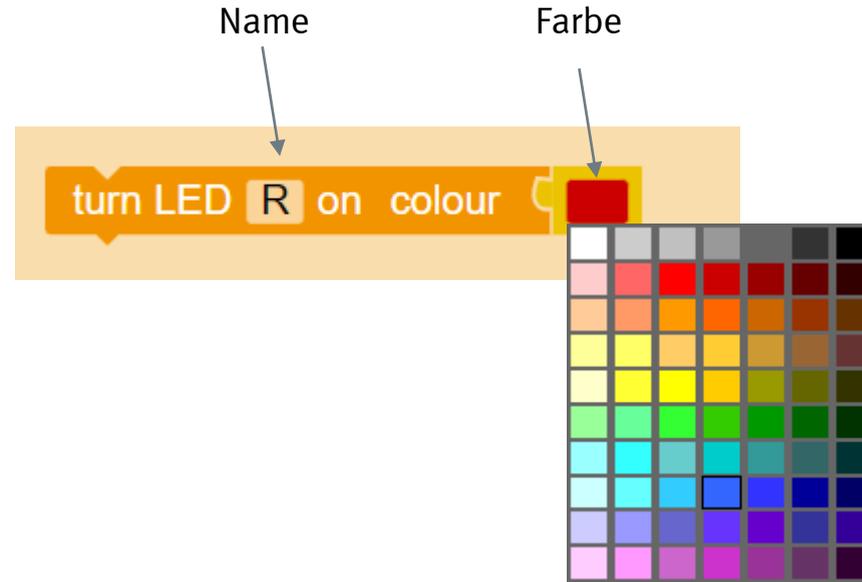
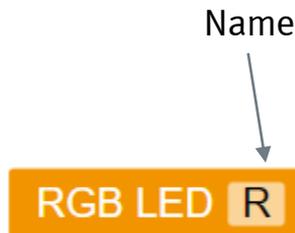


Die Warteinstruktion ermöglicht es, eine Wartezeit zwischen zwei Anweisungen zu setzen. Diese Anweisung muss einen Parameter haben (in blau). Es ist eine Zahl, die die Verzögerung in Millisekunden bestimmt.

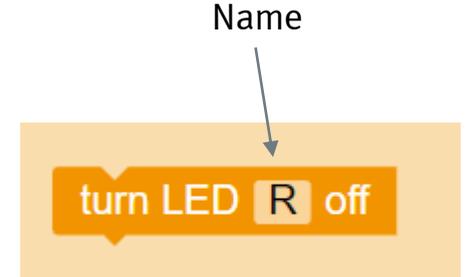
Einführung - Programmierung

Theorieteil. Funktion für Bionic Flower: LED einschalten und LED ausschalten

Im Konfigurationsroboter finden Sie das Objekt "RGB LED", der Name dieses Objekts ist standardmäßig R, aber Sie können den Namen ändern. Es ist nicht möglich, die LEDs einzeln zu steuern, dieses Objekt repräsentiert alle 5 LEDs.



Die Anweisung "**LED R einschalten**" ermöglicht es, die 5 LEDs in der Farbe einzuschalten, die mit der Farbtabelle ausgewählt wurde. Es gibt 70 Farben.

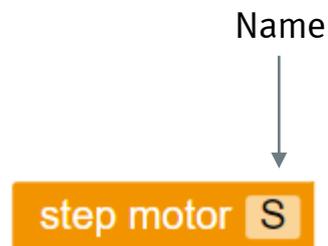


Die Anweisung **LED ausschalten** ermöglicht das Ausschalten der 5 LEDs.

Einführung - Programmierung

Theorieteil. Funktion für Bionic Flower: Schrittmotor

In der Konfiguration des Roboters finden Sie das Objekt "Schrittmotor". Der Name dieses Objekts ist standardmäßig S, aber Sie können den Namen ändern.

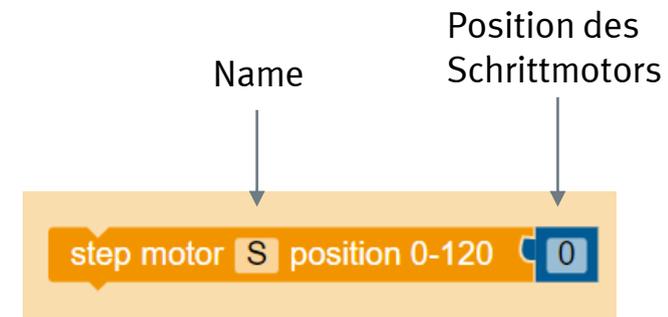


Der Schrittmotorbefehl ermöglicht die Bewegung des Schrittmotors.

Um von ganz geschlossen zu ganz geöffnet zu gelangen, muss der Schrittmotor 120 Umdrehungen machen.

Diese Schrittmotoranweisung muss einen Parameter (in blau) enthalten. Es ist eine Zahl, die die Anzahl der Umdrehungen angibt, um die Position der Öffnung der Blume zu bestimmen. Um die Bionic Flower vollständig zu schließen, ist diese Zahl 0. Um die Bionic Flower vollständig zu öffnen, ist diese Zahl 120.

! Die Einstellung der Motorposition entspricht der aktuellen Position des Motors, er wird nicht bewegt.



Einführung - Programmierung

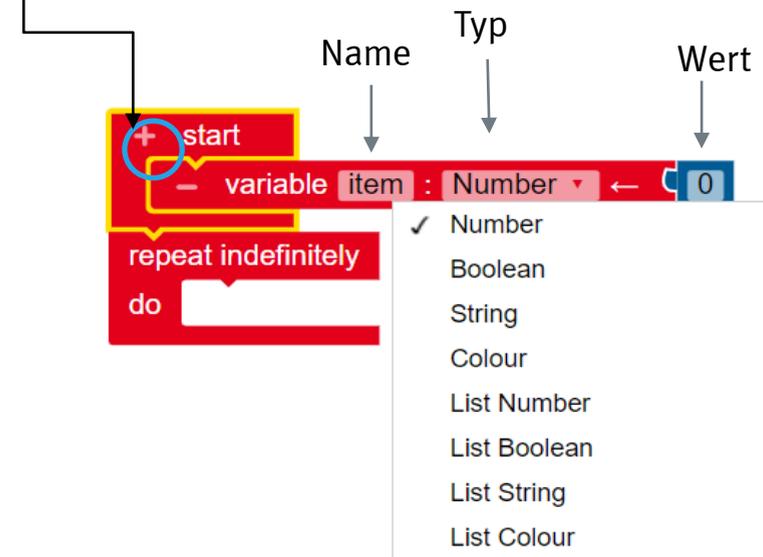
Theorieteil. Variable

Variablen werden zum Speichern von Daten verwendet.

Jede **Variable** wird durch 3 Attribute definiert:

- **Name:** Der Name einer Variablen kann eine beliebige Zeichenfolge sein. Alle Namen sind möglich, wie zum Beispiel "toto". Um das Verständnis des Codes zu erleichtern, ist es jedoch meist üblich, einen Namen zu verwenden, der die Variable beschreibt. Wenn diese Variable zum Beispiel Daten von einem Näherungssensor speichern soll, können wir sie "Abstand" nennen.
- **Typ:** Der Typ der Variablen legt fest, welche Art von Daten in dieser Variablen gespeichert wird. Dies ermöglicht die beste Organisation des Speichers. Die am häufigsten verwendeten Typen sind Zahlen, Boolesche Werte (wahr oder falsch) und Zeichenketten oder Zeichen.
- **Wert:** der Wert der Variablen, in der die Daten gespeichert werden. Zum Beispiel könnte der Wert der Variablen Abstand 10 für 10 cm sein.

Um eine Variable auf Open Roberta zu erstellen, müssen Sie auf das Hinzufügen-Symbol neben dem "Start" klicken. Legen Sie dann den Namen, den Typ und den Anfangswert der Variablen fest.



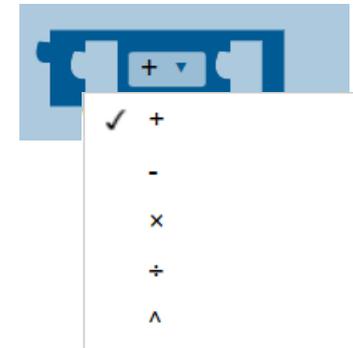
Einführung - Programmierung

Theorieteil. Variable und mathematische Operationen

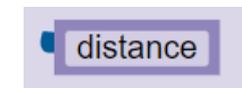
Sobald die Variable erstellt ist, kann sie mit der Anweisung set block to im Code geändert werden. Diese Anweisung erfordert einen Parameter desselben Typs wie der für die Variable deklarierte Typ. Nach dieser Anweisung wird die Variable also auf den Wert des Parameters gesetzt.



Der folgende Block erlaubt es, eine ganze Zahl als Parameter anzugeben.



Um die Werte der Variablen oder die Bedingungen zu ändern, haben wir die Möglichkeit, mathematische Operationen wie Addition, Subtraktion, Multiplikation, Division und Potenz zu verwenden. Darüber hinaus benötigt diese Anweisung zwei Parameter, die beide ganzzahlig sind.



Darüber hinaus ist es möglich, den Wert einer Variablen mit dieser Anweisung zu verwenden.

Einführung - Programmierung

Theorieteil. Zufällig

Der Zufallsbefehl ermöglicht die Erzeugung einer Zufallszahl zwischen der ersten und der zweiten enthaltenen Zahl. Das folgende Beispiel erzeugt eine Zufallszahl zwischen 1 und 100.



```
random integer from 1 to 100
```

Einführung - Programmierung

Theorieteil. Schleife: Unbegrenzt wiederholen



Die Struktur **repeat indefinitely** ist eine Schleife, die es ermöglicht, die in dieser Schleife enthaltenen Anweisungen unendlich oft zu wiederholen. Man kommt nie aus dieser Schleife heraus. Deshalb gibt es auch keine Ausstiegsbedingung.

Einführung - Programmierung

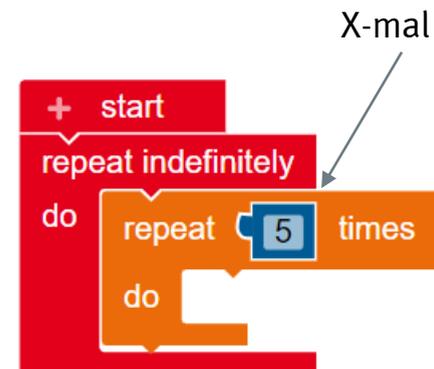
Theorieteil. Schleife: X-mal wiederholen

Die **X-malige Schleife** ermöglicht es, Anweisungen X-mal auszuführen.

Das bedeutet, dass die Anweisungen, die in der Struktur der Schleife enthalten sind (Anweisungen A), X-mal ausgeführt werden. Diese Schleife erfordert eine positive Zahl , X. Diese Zahl gibt an, wie oft die Anweisungen ausgeführt werden sollen. Danach fährt der Code mit den folgenden Anweisungen fort (Anweisungen B).

X-mal wiederholen

```
{  
    /*Anweisungen A*/  
}  
/*Anweisungen B*/
```



Einführung - Programmierung

Theorieteil. Variable: Wert vom Lichtsensor erhalten

In der Roboterkonfiguration können Sie den Lichtsensor hinzufügen. Sie können den Namen ändern. Der Name ist standardmäßig L.

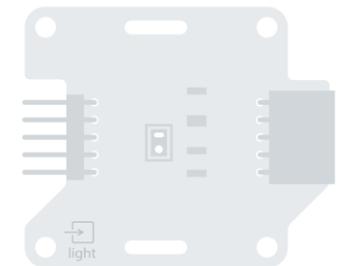
Name



light sensor L

Der Lichtsensor-Block liefert die Daten des Lichtsensors, d.h. den Wert der Helligkeit in Lux.

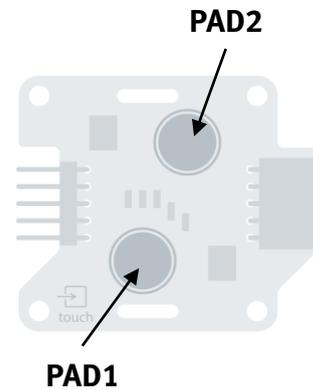
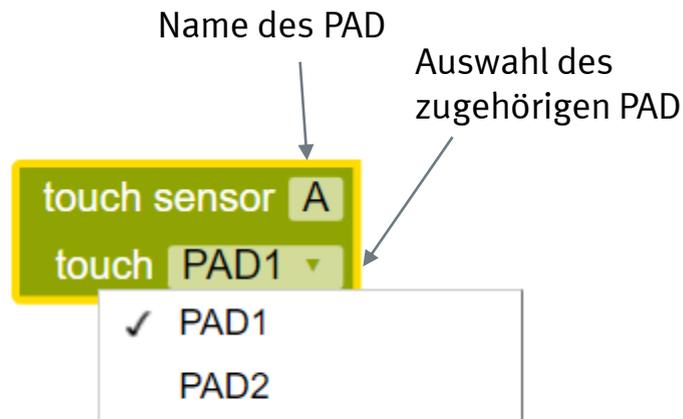
get value lx light sensor L



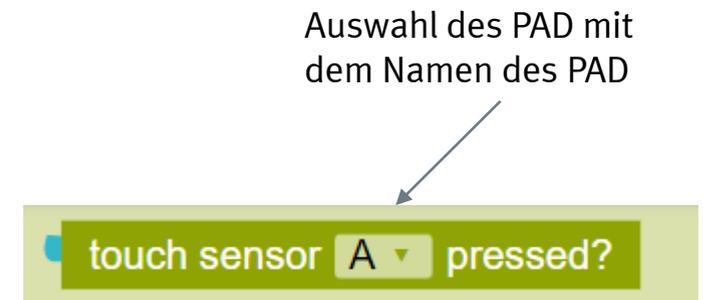
Einführung - Programmierung

Theorieteil. Variable: Wert vom Berührungssensor erhalten

In der Roboterkonfiguration können Sie den Berührungssensor hinzufügen. Sie können den Namen des PADS ändern. Jedes PAD muss einen eindeutigen Namen haben.



Der Berührungssensorblock liefert einen booleschen Wert (True oder False). Dieser Boolesche Wert ist True, wenn das ausgewählte PAD gedrückt wird, ansonsten False.



Einführung - Programmierung

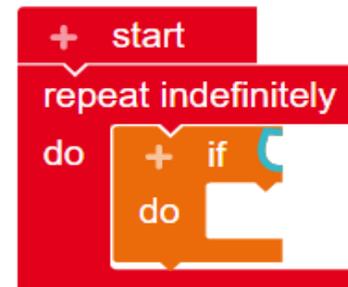
Theorieteil. Schleife: if do / if do else / If do else if else if

Die if-Strukturen sind in der Programmierung sehr verbreitet. Sie ermöglichen es, ein neues Verhalten Ihres Codes auf der Grundlage von Parametern wie der Zeit, dem Wert einer Variablen usw. zu erstellen.

If-Strukturen beruhen auf folgendem Prinzip: Eine Bedingung wird getestet, und wenn sie wahr ist, führt das Programm die Anweisung aus, die in dem mit dieser Bedingung verbundenen Körper enthalten ist; andernfalls ignoriert der Code die Anweisungen und setzt den Code fort.

- Die if do-Struktur ist die einfachste if-Struktur. In der Tat wird die Bedingung getestet. Wenn die Bedingung wahr ist, werden die Anweisungen im Körper der if-Struktur ausgeführt (Anweisungen A), dann fährt das Programm fort und führt die folgenden Anweisungen aus (Anweisungen B). Andernfalls, wenn die Bedingung falsch ist, ignoriert das Programm die Anweisungen in der if-Struktur und führt die folgenden Anweisungen aus.

```
wenn (Bedingung 1)
{
  /*Anweisungen A */
}
```

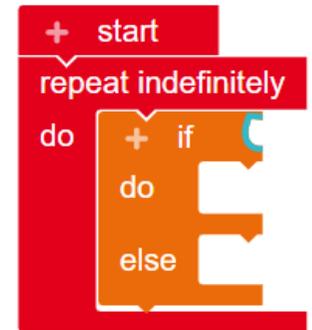


Einführung - Programmierung

Theorieteil. Schleife: if do / if do else / If do else if else if

- Die Struktur **if do else** erlaubt es, 2 Fälle zu unterscheiden: die Bedingung ist wahr und die Bedingung ist falsch. Und je nach dem Ergebnis gibt es zwei verschiedene Verhaltensweisen. In der Tat wird die Bedingung getestet. Wenn die Bedingung wahr ist, werden die Anweisungen im if-Körper, die mit der Bedingung verbunden sind, ausgeführt (Anweisungen A), dann fährt das Programm fort und führt die folgenden Anweisungen aus. Andernfalls ist die Bedingung falsch, das Programm führt die Anweisungen im else-Körper aus (Anweisungen B) und fährt dann mit der Ausführung der folgenden Anweisungen fort.
- Die **if do else if**-Struktur ermöglicht es Ihnen, mehrere Fälle zu trennen. Die Bedingungen werden nacheinander getestet, bis die Bedingung den Wert true liefert. Dann werden die mit dieser Bedingung verbundenen Anweisungen ausgeführt. Alle anderen Bedingungen werden dann ignoriert. Das Programm verlässt diese if-Struktur und führt die folgenden Anweisungen aus.
 Wenn zum Beispiel die Bedingung 1 falsch ist, wird die Bedingung 2 getestet. Wenn diese Bedingung wahr ist, wird die Anweisung B ausgeführt, und die anderen Bedingungen werden ignoriert. Wenn die Bedingung 2 falsch ist, wird die folgende Bedingung getestet usw. Wenn keine Bedingung wahr ist, wird keine der Anweisungen in der if-Struktur ausgeführt.

```
wenn (Bedingung 1)
{
  /*Anweisungen A */
}
sonst
{
  /*Anweisungen B */
}
```



Wenn sonst Struktur

```
wenn (Bedingung 1)
{
  /*Anweisungen A */
}
sonst wenn (Bedingung 2)
{
  /*Anweisungen B */
}
...
sonst wenn
{
  /*Anweisungen N */
}
```



If do else if Struktur

Einführung - Programmierung

Theorieteil. Schleife: Wiederholen, während

Die **while-Schleife** ermöglicht die Ausführung von Anweisungen, solange die zugehörige Bedingung erfüllt ist.

Das bedeutet, dass bei jedem Durchlauf der Schleife die Bedingung getestet wird. Wenn die Bedingung wahr ist, werden die Anweisungen im Schleifenkörper ausgeführt (Anweisungen A). Andernfalls, d. h. wenn der Bedingungstest falsch ist, verlässt der Code die Schleife und führt die folgenden Anweisungen aus (Anweisungen B).

```
wiederholen while (Bedingung)
{
    /*Anweisungen A */
}
/*Anweisungen B */
```



Einführung - Programmierung

Theorieteil. Schleife: Wiederholen bis

Die **until-Schleife** ermöglicht die Ausführung von Anweisungen, bis die zugehörige Bedingung erfüllt ist.

Das bedeutet, dass bei jedem Durchlauf der Schleife die Bedingung getestet wird. Wenn die Bedingung falsch ist, werden die Anweisungen im Schleifenkörper ausgeführt (Anweisungen A). Andernfalls, d. h. wenn der Bedingungstest wahr ergibt, verlässt der Code die Schleife und führt die folgenden Anweisungen aus (Anweisungen B).

wiederholen bis (Bedingung)

```
{  
  /*Anweisungen A */  
}  
/*Anweisungen B */
```



Einführung - Programmierung

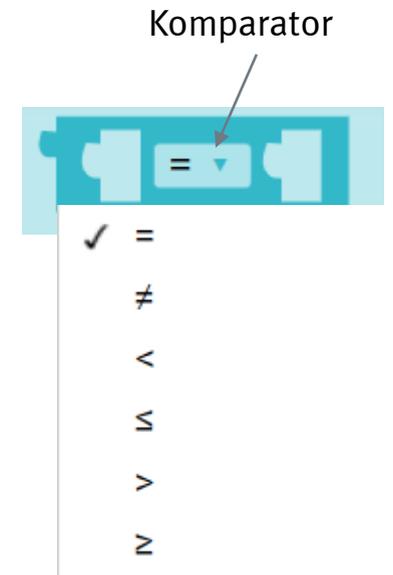
Theorieteil. Schleife: Bedingung/Vergleich

Bedingungen sind wichtig, denn sie ermöglichen die Einführung von Strukturen wie if-Strukturen, while-Schleifen oder until-Schleifen.

Das Testen einer Bedingung liefert einen booleschen Wert: **wahr** oder **falsch**.

Wenn Sie zwei Werte von zwei Variablen des Typs Zahl vergleichen wollen, können Sie die Vergleichsstruktur verwenden. Um zwei Zahlen zu vergleichen, können Sie die folgenden Komparatoren verwenden: Gleichheit, Nicht-Gleichheit, minderwertig, minderwertig oder gleich, überlegen und überlegen oder gleich.

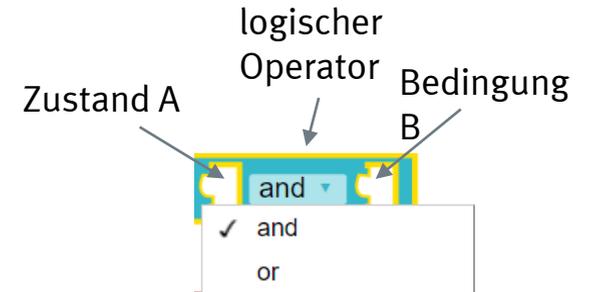
Dieser Block benötigt einen Komparator und zwei Zahlen. Es kann eine Blocknummer oder eine Variable vom Typ Zahl sein.



Einführung - Programmierung

Theorieteil. Schleife: Bedingung / logischer Operator AND OR

Um die Bedingungen komplexer zu gestalten, können Sie die logischen Operatoren **AND** und **OR** verwenden. Dieser logische Operator ermöglicht es, zwei Bedingungen miteinander zu verknüpfen.



Beim AND-Operator müssen beide Bedingungen wahr sein, damit das Ergebnis wahr ist.

AND

Zustand A	Zustand B	Ergebnis
Falsch	Falsch	Falsch
Wahr	Falsch	Falsch
Falsch	Wahr	Falsch
Wahr	Wahr	Wahr

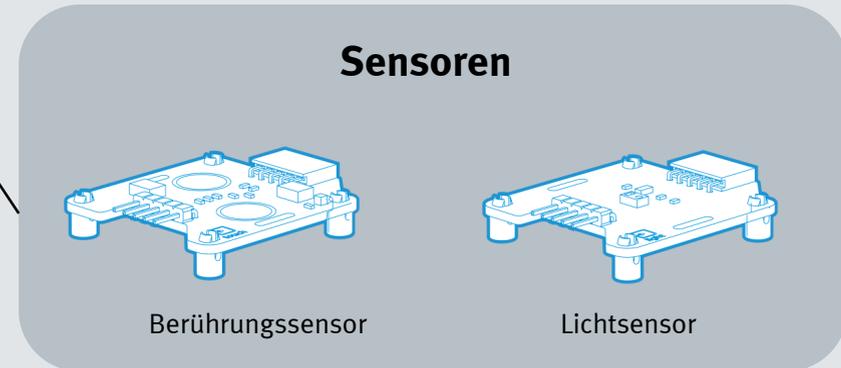
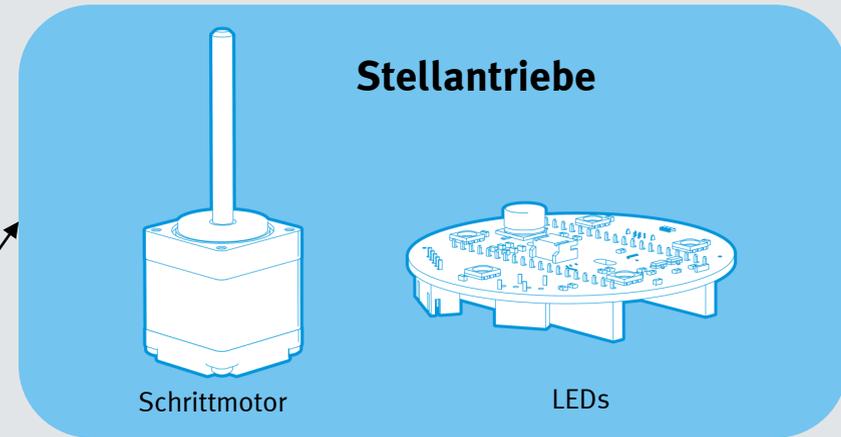
Der Operator **OR** muss mindestens eine Bedingung erfüllen, um wahr zu sein.

OR

Zustand A	Zustand B	Ergebnis
Falsch	Falsch	Falsch
Wahr	Falsch	Wahr
Falsch	Wahr	Wahr
Wahr	Wahr	Wahr

Einführung - Open Roberta

Programmierung der Bionic Flower mit Open Roberta

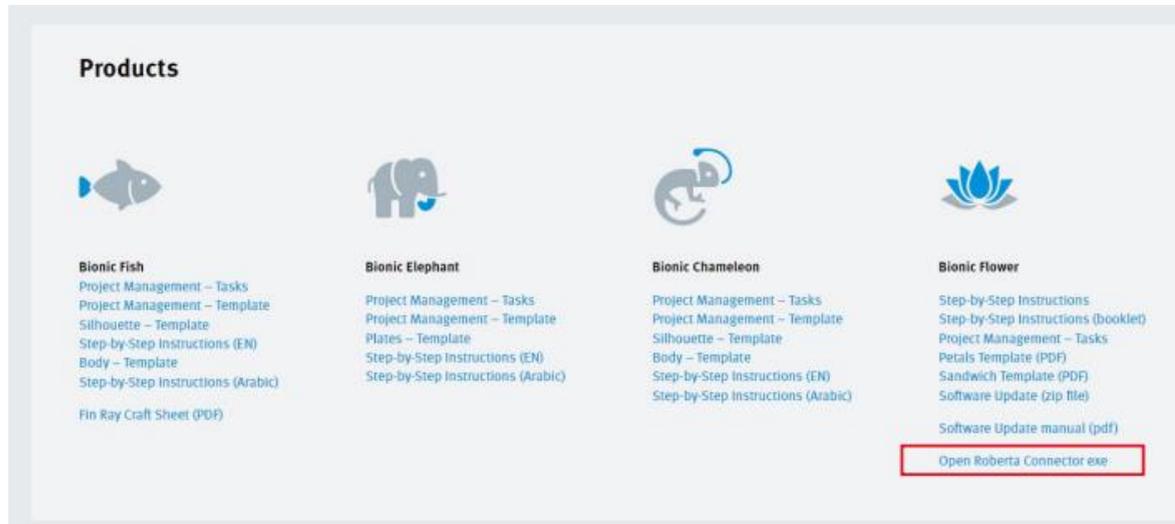


Bionic Flower – Roberta öffnen

Wie installiert man den Connector von Open Roberta?

Herunterladen und Entpacken des Connectors

- Besuchen Sie die Website von Bionics4Education: <https://www.bionics4education.com/startseite/support>
- Laden Sie die Datei Open Roberta Connector.exe herunter



Bevorzugen Sie ein Video, das erklärt, wie man Open Roberta installiert?
Rufen Sie das YouTube-Video mit folgendem Link auf:
<https://www.youtube.com/watch?v=JNCAL6aCfbl>

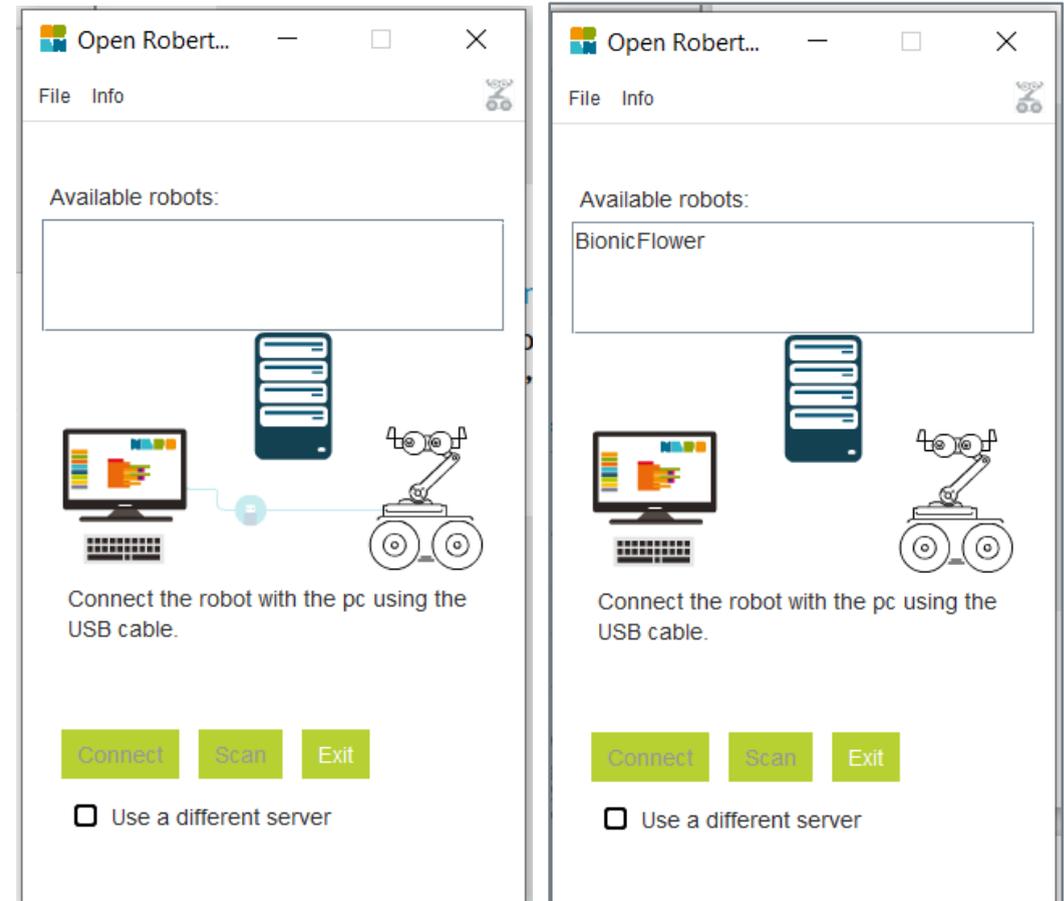
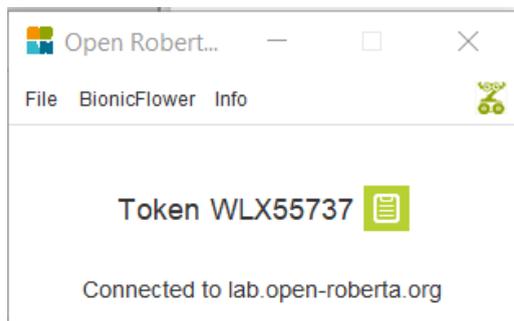
- Entpacken Sie die Datei

Bionic Flower - Roberta öffnen

Wie installiert man den Connector von Open Roberta?

Einrichtung Open Roberta Connector

- Öffnen Sie den Open Roberta Connector
 `OpenRobertaConnector.exe`
- Verbinden Sie die Bionic Flower über das USB-Kabel mit Ihrem PC
- Warten, dass die Bionic Flower erscheint
- Klicken Sie auf Verbinden
- Kopieren und Einfügen des Tokens



Bionic Flower - Roberta öffnen

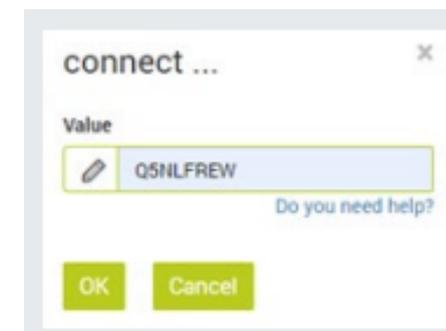
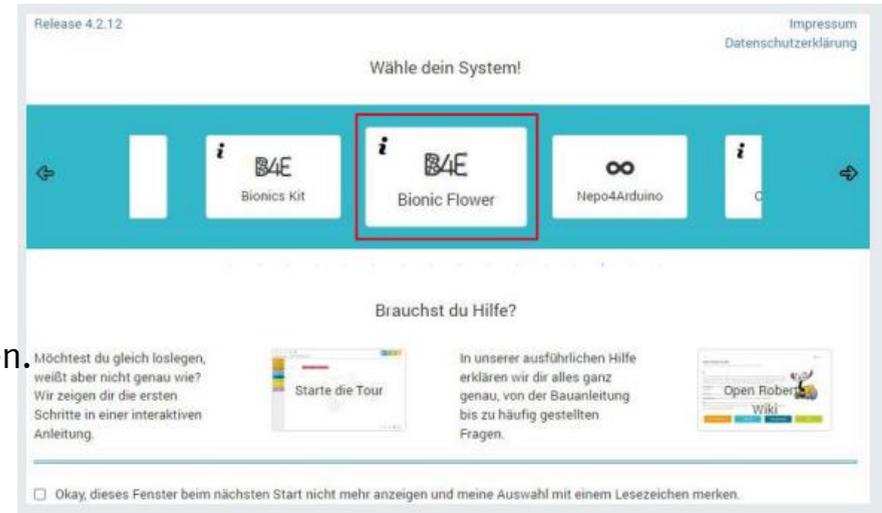
Wie installiert man den Connector von Open Roberta?

Einrichten Ihrer Bionic Flower auf Open Roberta

- Besuchen Sie die Website Open Roberta: <https://lab.open-roberta.org/>
- Wählen Sie die Bionic Flower als Ihr System
- Gehen Sie zum Menü B4E und klicken Sie auf , um die Bionic Flower zu verbinden.



- Geben Sie den Token ein, um die Bionic Flower zu verbinden und klicken Sie auf "Ok".



Bionic Flower - Roberta öffnen

Wie installiert man den Connector von Open Roberta?

Laden Sie einen Code auf Ihre Bionic Flower hoch

- Machen Sie Ihren ersten Code
- Um Ihren Code auf Ihre Bionic Flower hochzuladen, klicken Sie auf den Pfeil unten rechts
- Das B4E-Logo blinkt, um anzuzeigen, dass der Code auf die Bionic Flower hochgeladen wird.



Bionic Flower - Roberta öffnen

Wie installiert man den Connector von Open Roberta?

Importieren/Exportieren/Zurücksetzen

- Importieren

Gehen Sie zum ersten Menü und klicken Sie auf das Exportprogramm Eine Datei mit dem Format .xml wird in Ihrem Download-Ordner erstellt

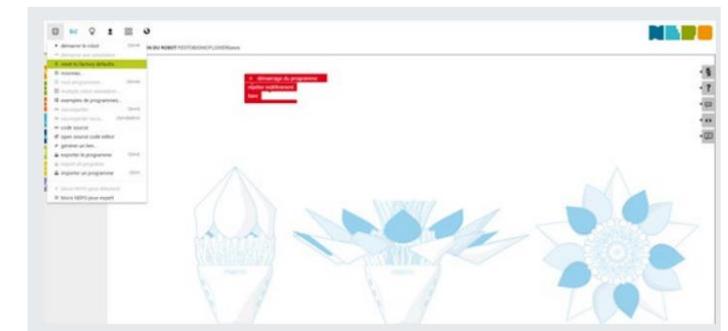
- Exportieren

Gehen Sie zum ersten Menü und klicken Sie auf das Importprogramm. Ihr Code erscheint auf der Schnittstelle Open Roberta.

- Standardcode zurücksetzen Bionic Flower

Wenn Sie den Bionic Flower mit dem ursprünglichen Code zurücksetzen möchten, gehen Sie zum ersten Menü und klicken Sie auf "Auf Werkseinstellungen zurücksetzen".

Beenden: Bevor Sie Open Roberta beenden, denken Sie daran, Ihren Code zu importieren, wenn Sie ihn das nächste Mal verwenden wollen. Um Open Roberta zu beenden, trennen Sie auf dem Open Roberta Connector die Blume und trennen Sie Ihre Blume von Ihrem Computer.



Coding-Projekte - Überblick

Mit Hilfe von **Open Roberta** wirst du Programme schreiben, um mit der Blume zu interagieren und Projekte zu erstellen. Um dir dabei zu helfen, sind drei Projekte bereits fertig, und wir werden gemeinsam Schritt für Schritt sehen, wie du eine Geschichte mit deiner Bionic Flower programmieren und erstellen kannst.



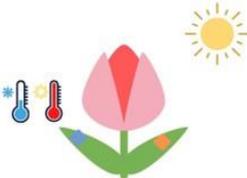
Jedes Heimwerkerprojekt ist mit einer anderen Programmierungsstufe **verbunden**.



- Ebene 1 : **Fleischfressende Pflanze**

Sie werden die unglaubliche Fähigkeit fleischfressender Pflanzen nachahmen, Insekten zu fangen.

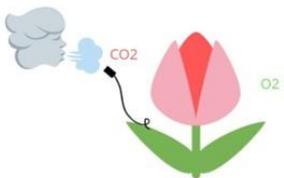
Zu Beginn ist die Blume offen und leuchtet grün. Wenn Sie den Berührungssensor (der das Insekt simuliert) drücken, schließt sich die Blume und blinkt rot. Nach einer Verzögerung öffnet sich die Blume wieder und leuchtet grün auf.



- Ebene 2: **Tag der Blume**

Sie werden einen ganzen Tag lang eine Seerosenblüte nachbilden. Tagsüber öffnet und schließt sich die Blüte nachts.

Je nach Leuchtkraft ändert die Blume die Farbe der LEDs und öffnet oder schließt sich.



- Ebene 3: **Photosynthese**

Sie werden die Fotosynthese nachstellen. Der Prozess der Photosynthese benötigt unter anderem Chlorophyllzellen und Sonnenenergie.

Mit dem Berührungssensor wählen Sie die richtige Farbe für die Photosynthese: Grün für die Chlorophyllzellen. Wenn die Leuchtkraft ausreicht und die Chlorophyllzellen vorhanden sind, öffnet sich die Blume und nimmt eine neue Farbe an, was bedeutet, dass alle Voraussetzungen für den Prozess gegeben sind.

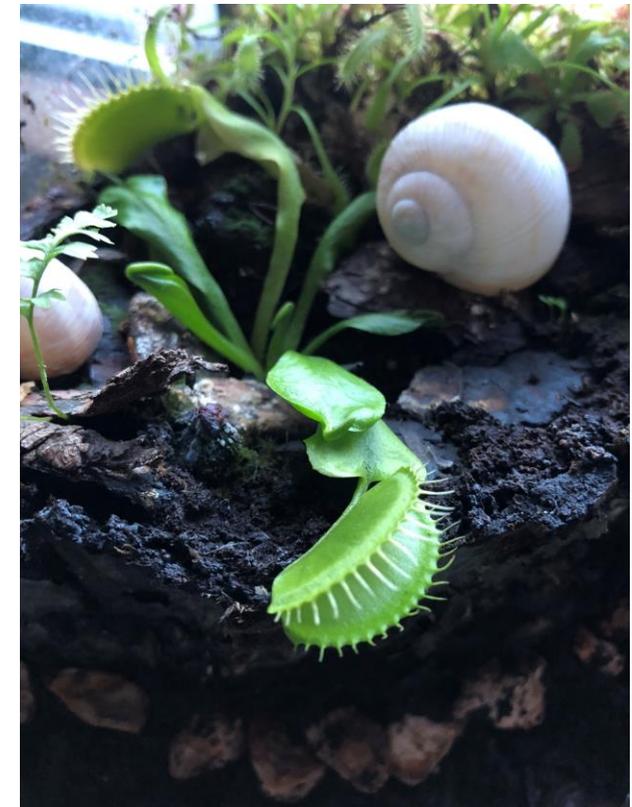
Coding Projekt - **Fleischfressende Pflanze**

Biologische Hintergrundinformationen

Fleischfressende Pflanzen sind einzigartig in ihrer Fähigkeit, ihre Beute, meist Insekten, anzulocken, zu fangen und (ganz oder teilweise) zu assimilieren. Einige Arten können sogar kleine Frösche oder Nagetiere, wie Mäuse, verdauen.

Die Fallen fleischfressender Pflanzen werden als "aktiv" bezeichnet, wenn sie mobil und schnell sind. Wenn die fleischfressende Pflanze nämlich ein Insekt entdeckt, schließt sie sich sofort und fängt die Beute ein. Nach der Verdauung öffnet sich die Pflanze wieder und wartet auf die nächste Beute.

Im nächsten Schritt werden Sie ein Programm schreiben, das mit der Bionic Flower interagiert. Zu Beginn ist die Bionic Flower offen und leuchtet grün. Wenn Sie den Berührungssensor berühren (um die Beute zu simulieren), schließt sich die Blume schnell und blinkt rot. Nach einer Verzögerung öffnet sich die Blume wieder und leuchtet grün. Darüber hinaus können Sie dieses Szenario auch mit dem Gestensensor verbessern. Sie können die 8 Bewegungen des Gestensensors verwenden, um verschiedene Verhaltensweisen der Blume zu definieren. Zum Beispiel die Bewegung "vorwärts", um das Schließen der Blume und das Blinken der LEDs auszulösen. Die Bewegung "zurück", um die Blume wieder zu öffnen. Die Bewegungen "rechts" und "links", um die Farbe der Blume zu ändern. Die Bewegungen "auf" und "ab", um die Intensität der Farbe zu verändern. Und die Bewegungen "im Uhrzeigersinn" und "gegen den Uhrzeigersinn", um eine Ladebewegung mit den LEDs im bzw. gegen den Uhrzeigersinn zu erzeugen.



Coding-Projekt - Fleischfressende Pflanze

Coding Aktivitäten

SCHRITT 1: Blinken der LED

Zielsetzung: Ein Blinken der LED erzeugen.

Tipps: Denken Sie daran, Verzögerungen zwischen dem Ausschalten und Einschalten der LED hinzuzufügen.

SCHRITT 2: Implementierung des Berührungssensors

Zielsetzung: Schaffe eine Gefahrensituation mit der LED.

Die Blume ist grün. Und wenn eine Berührung erkannt wird, blinkt die Blume rot.

Tipps: Denken Sie daran, den Berührungssensor der bionischen Blume hinzuzufügen.

SCHRITT 3 : Verwendung des Motors

Zielsetzung: Die LED und den Motor in eine Gefahrensituation bringen.

Die Blume ist grün und geöffnet. Und wenn eine Berührung erkannt wird, ist die Blume rot und schließt sich. Nach einer Verzögerung öffnet sich die Blume wieder.

SCHRITT 4 : Schaffen Sie eine Gefahrensituation für die fleischfressende Pflanze

Zielsetzung: Erzeuge eine Gefahrensituation, indem du mit der LED und dem Motor blinkst.

Die Blume ist grün und geöffnet. Und wenn eine Berührung erkannt wird, blinkt die Blume rot und schließt sich. Nach einer Verzögerung öffnet sich die Blume wieder.

Tipps: Verwenden Sie eine Variable "Step_motor", um die Schrittzahl des Motors zu speichern, und verwenden Sie eine mathematische Funktion, um sie zu erhöhen oder zu verringern.

Coding-Projekt - Tag der Blume

Biologische Hintergrundinformationen

Seerosen sind Wasserpflanzen, die in Gewässern verankert sind. Die einzeln stehenden Blüten haben eine spiralförmige Struktur, die durch Licht ausgelöst wird und so das Öffnen und Schließen der Blütenblätter ermöglicht. Tagblühende Seerosen öffnen ihre Blütenblätter am Morgen und locken so Insekten zur Bestäubung der Blüten an.

Das Prinzip beruht auf Wachstumsunterschieden zwischen der Außenseite und der Innenseite der Blütenblätter. So wächst die Innenseite der Blütenblätter bei hellem Licht und die Blüte öffnet sich. Wenn nicht genug Licht vorhanden ist, wächst die Außenseite und die Blüte schließt sich.

In diesem Projekt werden Sie ein Programm schreiben, das mit der Blume interagiert und den Tag einer Blume (z. B. Seerosen) nachstellt. Die Farbe der Blume hängt von der Tageszeit ab (Morgen (Umgebungslicht), Tag (mit dem Licht des Smartphones beleuchten) oder Nacht (die Hand auf den Sensor legen)). Die Blume braucht Licht, um sich zu öffnen (sie ist offen, wenn es Tag oder Morgen ist, und geschlossen, wenn es Nacht ist). Darüber hinaus hängt der Öffnungsgrad der Blüte von der Temperatur ab. Eine hohe Temperatur (Temperatursensor in der Hand) fördert das Wachstum der Blume, so dass sich die Blüte stärker öffnet.



In der Nacht: Seerosen sind geschlossen



Tagsüber: Seerosen sind geöffnet

Coding-Projekt - Tag der Blume

Coding Aktivitäten

SCHRITT 1: Einschalten der LED

Zielsetzung: Schalte die blaue und gelbe LED ein.

Tipps: Denken Sie daran, Verzögerungen (z. B. 1000 ms) zwischen den einzelnen Farbwechseln einzubauen, damit Sie die Veränderungen sehen können, sonst wäre es zu schnell für Ihre Augen.

SCHRITT 2: Implementierung des Lichtsensors

Zielsetzung: Definition von Tag und Nacht in Abhängigkeit vom Helligkeitswert.

Speichern Sie den Helligkeitswert in einer Variablen. Wenn die Helligkeit größer als 10 ist, ist es Tag, die Blume muss dann gelb leuchten. Andernfalls ist es Nacht, und die Blume sollte blau leuchten.

Tipps: Denken Sie daran, den Lichtsensor in die bionische Blume zu integrieren.

SCHRITT 3 : Verwendung des Motors

Zielsetzung: Öffnen und Schließen des Motors.

Speichern Sie den Helligkeitswert in einer Variablen. Wenn die Helligkeit höher als 10 ist, ist es Tag, die Blume muss dann gelb leuchten und die Blume öffnet sich. Andernfalls ist es Nacht, und die Blume muss blau leuchten und die Blume schließt sich.

SCHRITT 4 : Den Tag der Blume erstellen

Zielsetzung: Gestalte den Tag der Blume.

Die Blume wird am Tag geöffnet und in der Nacht geschlossen. Speichern Sie den Helligkeitswert in einer Variablen. Wenn die Helligkeit größer als 10 ist, ist es Tag, die Blume muss dann gelb leuchten und der Motor öffnet schrittweise. Andernfalls ist es Nacht, und die Blume soll blau leuchten und der Motor schrittweise geschlossen werden.

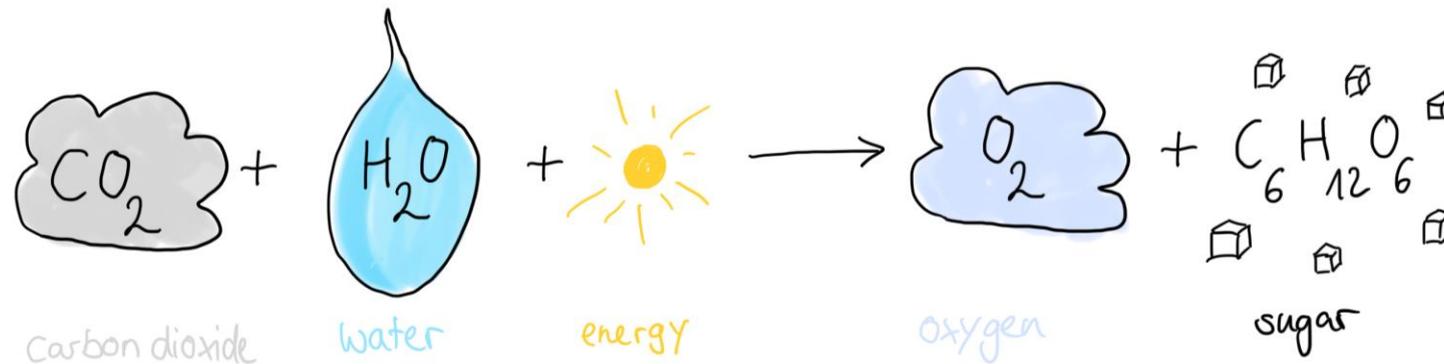
Tipps: Verwenden Sie eine Variable "Step_motor", um die Schrittzahl des Motors zu speichern, und verwenden Sie eine mathematische Funktion, um sie zu erhöhen oder zu verringern.

Coding-Projekt - Photosynthese

Biologische Hintergrundinformationen

Die Photosynthese ist einer der wichtigsten biochemischen Prozesse in der Natur. Ohne dieses Phänomen gäbe es kein Leben auf unserer Erde.

Bei der Photosynthese werden Kohlendioxid (CO₂) und Wasser (H₂O) mit Hilfe von Energie (Sonnenlicht) in Sauerstoff (O₂) und Zucker (C₆H₁₂O₆) umgewandelt. Der Zucker wird von den Pflanzen zum Leben und Wachsen verwendet. Der nicht benötigte Sauerstoff wird an die Umwelt abgegeben, die wiederum von anderen Lebewesen zum Überleben benötigt wird.



In diesem Projekt werden Sie den Effekt der Photosynthese nachbilden. Sie werden ein Programm schreiben, um mit der Blume zu interagieren. Mit dem Berührungssensor wählen Sie "grün" aus (was Chlorophyllzellen darstellt). Mit dem Lichtsensor können Sie festlegen, ob es Tag oder Nacht ist. Die Blume öffnet sich, wenn es Tag ist, und schließt sich, wenn es nicht Tag ist. Darüber hinaus können Sie auf den CO₂-Sensor pusten, um das Vorhandensein von CO₂ darzustellen.

Coding-Projekt – Photosynthese

Coding Aktivitäten

SCHRITT 1: Schalten Sie eine zufällige Farbe ein

Zielsetzung: Schalte die LED in einer zufälligen Farbe ein. Bestimmen Sie eine zufällige Farbe zwischen 4 Farben (rot, grün, blau und gelb) und schalten Sie die LED in dieser Farbe ein.

Tipps: Verwenden Sie die Zufallsfunktion, um eine Zufallszahl zwischen 1 und 4 zu erhalten.

SCHRITT 2: Implementierung des Berührungssensors

Zielsetzung: Mit dem Berührungssensor die Farbe der Blume ändern. Definieren Sie eine zufällige Farbe zwischen 4 Farben (rot, grün, blau und gelb) und schalten Sie die LED in dieser Farbe ein. Auf unbestimmte Weise, wenn eine Berührung erkannt wird, ändert sich die Farbe für die nächste Farbe.

Tipps: Denken Sie daran, den Berührungssensor zur bionischen Blume hinzuzufügen. Verwenden Sie eine Variable "Farbe", um die Nummer der Farbe zu speichern und verwenden Sie eine mathematische Funktion, um sie zu erhöhen oder zu verringern.

SCHRITT 3: Implementierung des Lichtsensors

Zielsetzung: Definition von Tag und Nacht in Abhängigkeit vom Helligkeitswert. Speichern Sie den Helligkeitswert in einer Variablen. Wenn die Helligkeit größer als 10 ist, ist es Tag. Andernfalls ist es Nacht. Wenn es Tag ist und die Farbe grün ist, so blinkt die Blume 5 mal mit einer neuen Farbe (lila). Nach einer Verzögerung leuchtet die Blume wieder in einer zufälligen Farbe.

Tipps: Denken Sie daran, den Lichtsensor an der bionischen Blume anzubringen

SCHRITT 4: Erstellen der Fotosynthese

Zielsetzung: Erstellen Sie die Fotosynthese. Du brauchst eine Chlorophyllzelle und ein Licht, damit die Blume gut wächst. Bestimmen Sie eine zufällige Farbe. Du kannst die Farbe mit dem Berührungssensor ändern. Wenn die Farbe grün ist und die Leuchtkraft ausreichend ist, öffnet sich die Blume und leuchtet in einer neuen Farbe. Und nach einer Verzögerung schließt sich die Blume und leuchtet in einer neuen Zufallsfarbe.

Geschafft!

Bionics4Education: Bionic Flower Courseware für Open Roberta

Herzlichen Glückwunsch!

Sie haben diesen Kurs abgeschlossen!

