

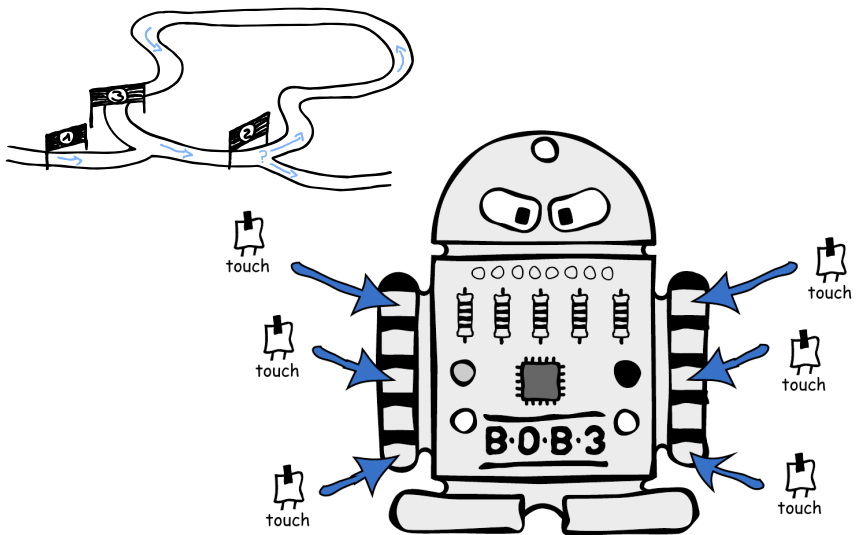
INFORMATIK

Programmieren lernen mit BOB3

Lernbegleitheft + Lösungen zu den Arbeitsblättern - Sekundarstufe I
Einführung in die textuelle Programmierung mit dem Roboter BOB3

Modul 2

Sensoren & Kontrollstrukturen



Dieses Dokument steht unter der Creative Commons Namensnennung 4.0 International (CC BY-SA 4.0) Lizenz. Autor: Katja Bach. Herausgeber: www.bob3.org, Stolberg, 2022.
DIGITALE BILDUNG: Programmieren lernen mit BOB3, Begleitheft

Modul 2

Sensoren & Kontrollstrukturen

Sensoren und Kontrollstrukturen

Dieses Dokument dient zur Übersicht des Kurses „Programmieren lernen mit BOB3“ für die Sekundarstufe I. Es werden die Lehr- und Lernmaterialien, die benötigten Zeiteinheiten und Vorschläge für konkrete Unterrichtseinheiten gegeben. Im Folgenden ist eine Unterrichtseinheit für 45 min ausgelegt.

Übersicht Modul 2:

Modul 2 umfasst insgesamt 6 Unterrichtseinheiten mit je ca. 45 Minuten



Lehrerbegleitheft mit Konzepten und Ablaufplänen der einzelnen Einheiten

12 Programmier-Tutorial Einheiten:



Die Schüler lernen Verzweigungen anhand der Kontrollstruktur ‚if / else‘ kennen und arbeiten mit Vergleichsoperatoren und Wahrheitswerten. In kleinen Programmbeispielen wird die Verzweigung konkret zur Auswertung des IR-Sensors eingesetzt. In diesem Zusammenhang lernen die Schüler das Konzept von Variablen kennen und anwenden. Sie lernen die einzelnen Hardware-Bestandteile des Roboters und die jeweiligen Funktionen kennen. Anhand der Software-Bibliothek des BOB3 setzen sie sich mit Methoden, Funktionen und Parametern auseinander und erlernen als weitere Kontrollstruktur das Prinzip und die Anwendung von ‚for-Schleifen‘. Zum Abschluss der Lerneinheit erfolgt eine einfache Fehlersuche, bei der die Schüler lernen, Fehler im Programmcode zu entdecken und zu beheben.



Arbeitsblatt 5 - „**if-else-Anweisung**“
+ Lösungen zum AB 5



Arbeitsblatt 6 - „**Bob’s Hardware**“
+ Lösungen zum AB 6



Arbeitsblatt 7 - „**Bob’s Software**“
+ Lösungen zum AB 7



Arbeitsblatt 8 - „**for-Schleife**“
+ Lösungen zum AB 8

OER-Materialien: <http://www.bob3.org/mint>
Hauptseite des Tutorials: <http://www.progBob.org>

1. Unterrichtseinheit

In der ersten Unterrichtseinheit lernen die Schülerinnen und Schüler die **if-else-Anweisung** als erste **Kontrollstruktur** kennen. In dem Zusammenhang lernen die SuS **Bedingungen, Wahrheitswerte** und **Vergleichsoperatoren** kennen und anwenden.



Zeitbedarf: ca. 45 min



Vorkenntnisse: BOB3 Modul 1



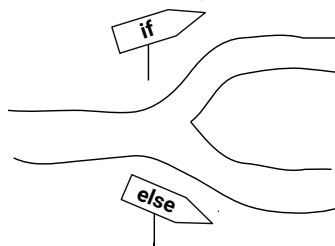
Material: Papier, Stift, Arbeitsblatt 5, PC oder Laptop, BOB3 mit Helm

Ablauf

Die Schüler bearbeiten das **Arbeitsblatt 5** und lernen als erste Programmierstruktur die Kontrollstruktur „**if-else**“ kennen:



„Eine **if-else-Anweisung** ist eine **zweiseitige Verzweigung**“



Die SuS erlernen die abstrakte Definition einer **zweiseitigen Verzweigung** kennen und verstehen das Konzept anschließend anhand von konkreten Programmier-Beispielen.

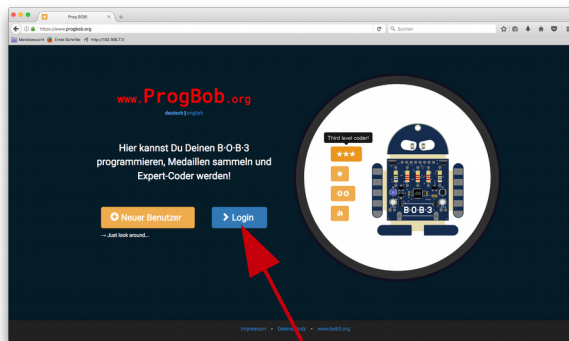
Das Arbeitsblatt gibt zusätzlich eine Übersicht über mathematische **Vergleichsoperatoren** und die entsprechenden Ausdrücke beim Programmieren.

Vergleichsoperatoren:

Mathe	Vergleichsoperator	Beispiel	Erklärung
=	==	a == b	→ ergibt „ wahr “, falls die Werte a und b gleich sind
≠	!=	a != b	→ ergibt „ wahr “, falls die Werte a und b ungleich sind
>	>	a > b	→ ergibt „ wahr “, falls a größer als b ist
<	<	a < b	→ ergibt „ wahr “, falls a kleiner als b ist
≥	>=	a >= b	→ ergibt „ wahr “, falls a größer oder gleich b ist
≤	<=	a <= b	→ ergibt „ wahr “, falls a kleiner oder gleich b ist

Anschließend bearbeiten die SuS die Aufgaben 1 – 8 des Arbeitsblatts und besprechen ihre Lösungen.

Die Schüler loggen sich mit ihren jeweiligen Accounts auf der Seite <http://www.ProgBob.org> ein und starten das „Intro II“-Tutorial:



!! Wichtig: erneutes Einloggen mit bereits vorhandenen Daten über „**Login**“ !!

Die Schüler bearbeiten die **erste Tutorial-Einheit** und vertiefen mit dem Szenario „**BOB3 als Wahrheitsfinder**“ das zuvor erworbene Wissen:

```
1 #include <BOB3.h>
2
3 // B-0-B-3 als Wahrheitsfinder
4
5 void setup() {
6
7 }
8
9 void loop() {
10
11 // Falls die Bedingung wahr ist -> Augen grün
12 if (1 == 1) {
13     bob3.setEyes(GREEN, GREEN);
14 }
15 // Sonst -> Augen rot
16 else {
17     bob3.setEyes(RED, RED);
18 }
19 }
20 }
21 }
```

In der Lerneinheit wird zunächst eine Bedingung vorgegeben, die von den Schülern ausprobiert wird. Anschließend bekommen sie die Aufgabe, verschiedene andere Bedingungen einzugeben und auf dem BOB3 zu testen. Falls die eingegebene Bedingung **wahr** ist, dann leuchten am Bob die Augen **grün**. Falls die Bedingung **falsch** ist, dann leuchten die Augen **rot**.

In der **Wissensüberprüfungs-Einheit** des ersten Teils werden anschließend noch einzelne Aufgaben zum Thema **Bedingungen** bzw. **wahr** oder **falsch** gestellt. Die Schüler bearbeiten diese Aufgaben und können ihre Lösung direkt auswerten lassen und reflektieren!

__quiz__

1: Wahr oder falsch: 😊
 wahr falsch

2: Wahr oder falsch: 😊
 wahr falsch

3: Wahr oder falsch: 😊
 wahr falsch

4: Wahr oder falsch: 😞
 wahr falsch

😊 Schon ein paar richtig...

2. Unterrichtseinheit

In der zweiten Unterrichtseinheit bearbeiten die Schülerinnen und Schüler die **Lerneinheiten zwei** und **drei** des „Intro II“-Tutorials, die sich ebenfalls mit der Kontrollstruktur „**if-else**“ beschäftigen. Sie experimentieren mit dem bestehenden Programm und vertiefen ihre Kenntnisse über die Programmierung mit **Vergleichsoperatoren** „**!=**“, „**<**“ und „**>**“. Als Anwendung des Gelernten bietet die dritte Lerneinheit das Programm, „**Grundschüler oder nicht?**“, mit dem sich die Schüler anhand eines kleinen Algorithmus einklassifizieren können.



Zeitbedarf: ca. 45 min



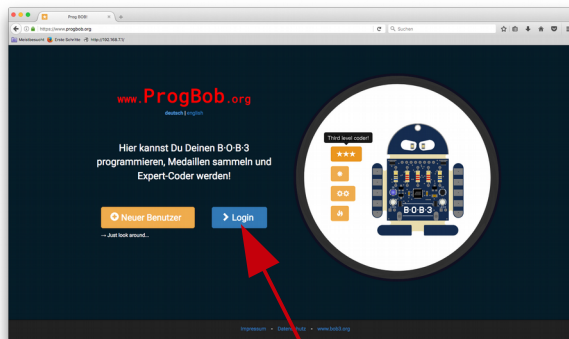
Vorkenntnisse: BOB3 Modul 1



Material: PC oder Laptop, BOB3 mit Helm

Ablauf

Die SuS starten den Webbrowser, gehen auf die Seite **http://www.ProgBob.org**, loggen sich mit Ihrem Account ein und gehen zum „Intro-II“-Tutorial:



„Login“

Als Beispiel aus der **eigenen Erlebniswelt** der Schüler probieren sie nun ein Programm aus, das anhand des **Geburtsjahres** einer Person ermittelt, ob diese Person noch in die Grundschule geht oder nicht mehr! Die Schüler geben ihr eigenes Geburtsjahr ein und können mit dem Programm experimentieren.

```
1 #include <BOB3.h>
2
3 // BOB3 als Wahrheitsfinder:
4 // Grundschüler oder nicht?
5
6 void setup() {
7
8 }
9
10 void loop() {
11
12     int geburtsjahr;
13     geburtsjahr = 2005;
14
15     // Grundschüler detektiert -> Augen orange
16     if (geburtsjahr > 2007) {
17         bob3.setEyes(ORANGE, ORANGE);
18     }
19     // Sonst -> Augen blau
20     else {
21         bob3.setEyes(ROYALBLUE, ROYALBLUE);
22     }
23
24 }
```

Die Lerneinheit erklärt den Sinn und die Verwendung einer **Variablen**. Die SuS lernen, wie eine Variable **deklariert** und **initialisiert** wird. Anschließend wird der Variable ein konkreter Zahlenwert **zugewiesen**. In der folgenden if-else-Abfrage wird die Variable in der Bedingung verwendet.

__info__

In Zeile 13 wird unsere neue Variable **initialisiert**, das heißt sie bekommt erstmalig einen Wert zugewiesen:

```
geburtsjahr = 2005;
```

Für den Compiler ist ab jetzt die Variable `geburtsjahr` gleichbedeutend mit der Zahl `2005`.

Jetzt können wir in unserem eigentlichen Wahrheitsfinder (Zeile 16 - 22) die Variable verwenden!

3. Unterrichtseinheit

In der dritten Unterrichtseinheit bearbeiten die Schülerinnen und Schüler die **Lerneinheiten vier** und **fünf** des „Intro II“-Tutorials, die sich mit dem **IR-Sensor** des Roboters beschäftigen. Sie lernen, einen **Sensorwert abzufragen**, das Ergebnis in einer **Variablen zu speichern** und im folgenden Programm mit dieser Variablen zu arbeiten. Je nach Variablenwert soll der Roboter unterschiedlich reagieren.



Zeitbedarf: ca. 45 min



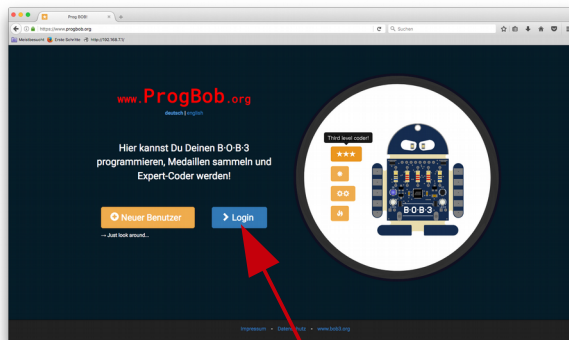
Vorkenntnisse: BOB3 Modul 1



Material: Papier, Stift, Arbeitsblatt 6, PC/Laptop/Tablet, BOB3 mit Helm/Dock

Ablauf

Die SuS starten den Webbrowser, gehen auf die Seite <http://www.ProgBob.org>, loggen sich mit Ihrem Account ein und gehen zum „Intro-II“-Tutorial:



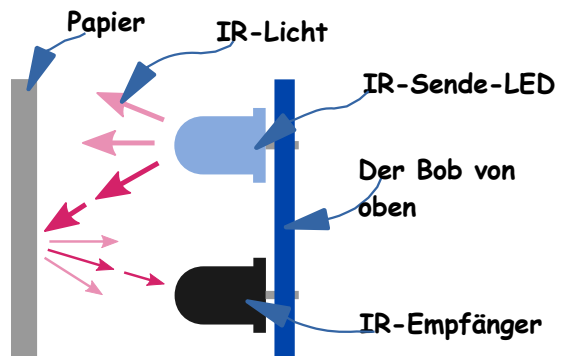
„Login“

Insbesondere lernen die Schüler, wie der **IR-Sensor** des BOB3 funktioniert. Der Sensor besteht aus zwei Teilen: Einer **IR-Sende-LED** und einem **IR-Empfänger**. Auf diese Weise können zwei Roboter-Einheiten drahtlos miteinander kommunizieren oder sich gegenseitig fernsteuern. Beispiele hierzu sind in den späteren Vertiefungs-Tutorials enthalten.

Zusätzlich kann der Roboter mit dem IR-Sensor **berührungslos Objekte** „bemerken“ und auf diese reagieren. Er bemerkt auch den Unterschied zwischen **nah** und **fern** und kann darauf verschieden reagieren. Um zu verstehen, wieso der Roboter diese Fähigkeit hat, erlernen die Schüler das Prinzip des **Reflektionsverfahrens**:

Die **IR-Sende-LED** sendet IR-Licht aus, dieses trifft dann auf ein Hindernis (z.B. ein Blatt Papier), wird von dem Hindernis **zurückreflektiert** und kann so von dem IR-Empfänger empfangen werden.

Je näher das Papier vor dem Sensor ist, **desto mehr IR-Licht detektiert** der IR-Empfänger.



Die Schüler bearbeiten das nächste Programmier-Beispiel und **experimentieren** mit verschiedenen Werten. Sie finden heraus, bei welchen Grenzwerten ihr IR-Sensor nah und fern detektiert und lernen so reale Robotik kennen. Anschließend erweitern die Schüler das Programm um eine „**else-if-Abfrage**“, um noch eine Abstufung einzubauen:

```
12 // Viel Reflektion - Objekt nah
13 if (irWert > 8) {
14     bob3.setEyes(RED, RED);
15
16 // Weniger Reflektion - Objekt weiter weg
17 } else if (irWert > 12) {
18     bob3.setEyes(BLUE, BLUE);
```

info

Das Programm soll jetzt so ergänzt werden, dass die Augen bei **wenig** Reflektion **blau** leuchten und bei **viel** Reflektion **rot** leuchten.

Dazu fügen wir in Zeile 17 eine **else if** Abfrage ein.

4. Unterrichtseinheit

In der vierten Unterrichtseinheit bearbeiten die Schülerinnen und Schüler die **Lerneinheiten sechs** und **sieben** des „Intro II“-Tutorials. Zunächst arbeiten sie mit dem im Mikrocontroller integrierten **Temperatur-Sensor** des Roboters. In diesem Zusammenhang lernen die SuS das Konzept und die Bedeutung einer **globalen Variablen** kennen und anwenden. Im nächsten Teil beschäftigen sie sich insbesondere mit der **Software-Bibliothek** des BOB3. Sie lernen **neue Methoden** kennen und vertiefen ihr Wissen über **Parameter**.



Zeitbedarf: ca. 45 min



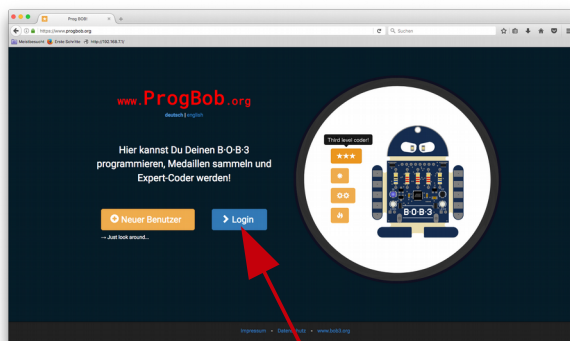
Vorkenntnisse: BOB3 Modul 1



Material: Papier, Stift, Arbeitsblatt 7, PC oder Laptop, BOB3 mit Helm

Ablauf

Die SuS starten den Webbrowser, gehen auf die Seite <http://www.ProgBob.org>, loggen sich mit Ihrem Account ein und gehen zum „Intro-II“-Tutorial:



„Login“

Zunächst probieren die Schüler das bestehende Programm-Beispiel aus: Der Mikrocontroller **misst beim Einschalten die Temperatur**, dann sind die Augen grün. Wenn man den Controller dann z.B. mit einem Finger **erwärmt**, dann steigt die Temperatur und die Augen werden **rot**. Wenn man den BOB3 dann z.B. in den Kühlschrank legt, dann wird ihm **kalt** und die Augen werden **blau**.

The screenshot shows the ProgBob IDE interface. On the left, there are navigation icons for Medals, a search icon, a file icon, a home icon, a settings icon, a star icon, a Wi-Fi icon, a trophy icon, a paw icon, and a menu icon. The main area is a code editor with the following C++ code:

```

1 #include <BOB3.h>
2
3 // Globale Variable
4 int start_temp;
5
6 void setup() {
7   bob3.setEyes(GREEN, GREEN);
8   delay(1000);
9
10  // Temperatur beim Einschalten abfragen
11  start_temp = bob3.getTemperature();
12 }
13
14 void loop() {
15
16  // Temperatur abfragen
17  int aktuelle_temp = bob3.getTemperature();
18
19  // Unterschied ausrechnen
20  int differenz = aktuelle_temp - start_temp;
21
22  if (differenz < -1) {
23    bob3.setEyes(BLUE, BLUE);
24  } else if (differenz > 1) {
25    bob3.setEyes(RED, RED);
26  } else {
27    bob3.setEyes(GREEN, GREEN);
28  }
29
30  delay(100);
31 }

```

Below the code editor is a "Compile" button. To the right of the code editor is a tutorial window titled "info" with the text: "Der Mikrocontroller auf deinem BOB3 hat einen Temperatur-Sensor integriert." Below the text are two thermometers: one with a blue bulb and one with a red bulb. To the right of the tutorial window is a "Tutorials" sidebar with buttons for "Intro I", "Intro II", "Intro III", "Sense", "Touch", "Color", "Comm", "Bugs", and "Free".

Weiterhin erlernen die Schüler das Konzept einer **globalen Variablen** und experimentieren mit verschiedenen Variablen in dem Temperatur-Programm.

Die Schüler bearbeiten das **Arbeitsblatt 7**, in dem die **Software-Bibliothek** des BOB3 erklärt wird.



„Bob's Software“

The screenshot shows the BOB3 software library documentation. It is titled "Bibliothek" and contains two sections: "BOB3 - Methoden:" and "Globale Funktionen:". The "BOB3 - Methoden:" section lists the following methods:

```

void bob3.setLed(id, color)
void bob3.setEyes(color1, color2)
void bob3.setWhiteLeds(status1, status2)
int bob3.getLed(id)
int bob3.getArm(id)
void bob3.enableArms(enable)
int bob3.getIRSensor()
int bob3.getIRLight()
void bob3.enableIRSensor(enable)
int bob3.getTemperature()
int bob3.getMillivolt()
int bob3.getID()
int bob3.receiveMessage(timeout)
void bob3.transmitMessage(message)

```

The "Globale Funktionen:" section lists the following global functions:

```

void delay(milliseconds)
int mixColor(color1, color2, w1, w2)
int rgb(red, green, blue)
void remember(value)
int recall()

```

Das Arbeitsblatt 7 erklärt insbesondere, wie man für einzelne **Methoden** und **Funktionen** gültige **Parameter** herausfinden und anwenden kann:



Die Schüler vertiefen ihr Wissen über Methoden mit **Parametern** und wenden dieses in einem eigenen Programm an.

Lerninhalte des „Intro II“-Tutorials:

- SuS lernen Bedingungen wie 'gleich', 'ungleich' oder 'kleiner als' im Kontext einer Programmiersprache kennen
- SuS setzen die Bedingungen mit dem 'if/else' Konstrukt der Programmiersprache ein und experimentieren damit
- SuS lernen das Konzept einer Ganzzahl-Variablen kennen
- SuS speichern den Wert eines Sensors in einer Variablen und werten diese mit dem 'if/else' Konstrukt aus
- SuS lernen den IR-Reflex-Sensor und den Temperatur-Sensor kennen
- SuS erlernen den Sinn und Zweck einer 'for'-Schleife zur wiederholten Durchführung
- SuS lernen wie die Parameter der 'for'-Schleife eingesetzt werden können
- SuS lernen wie man Fehler im Programm-Code lokalisieren kann und wie man diese behebt

5. + 6. Unterrichtseinheit

In der fünften und sechsten Unterrichtseinheit beenden die Schüler das „**Intro II**“ Tutorial. Sie beschäftigen sich in **fünf Tutorial-Einheiten** mit dem Konzept der **for-Schleife** als Zähl-Schleife. Es werden verschiedene Programme als Beispiele angeboten, die von den SuS zunächst ausprobiert und anschließend experimentell verändert und erweitert werden. Zusätzlich bearbeiten die SuS das **Arbeitsblatt 8 „for-Schleife“**. Hierbei vertiefen sie ihr experimentell erworbenes Wissen zur Theorie der **for-Schleife** als Kontrollstruktur.



Zeitbedarf: ca. 90 min



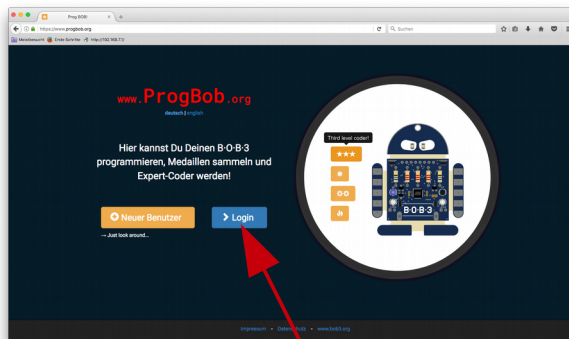
Vorkenntnisse: BOB3 Modul 1



Material: Papier, Stift, Arbeitsblatt 8, PC oder Laptop, BOB3 mit Helm

Ablauf

Die SuS starten den Webbrowser, gehen auf die Seite <http://www.ProgBob.org>, loggen sich mit Ihrem Account ein und gehen zum „**Intro-II**“-Tutorial:

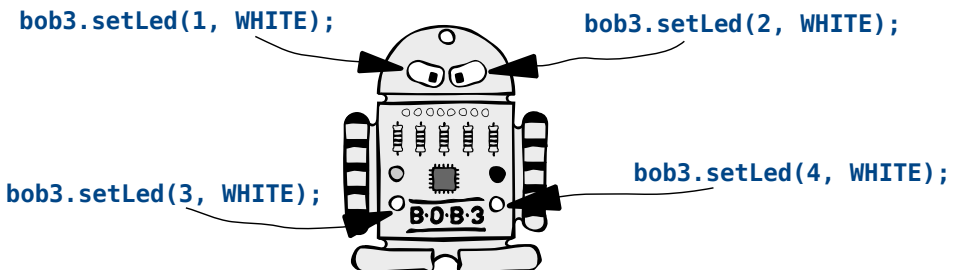


„Login“

Die Schüler lernen das **Konzept einer for-Schleife** zur wiederholten Durchführung an einem einfachen Programmierbeispiel kennen:

```
1 #include <BOB3.h>
2
3 void setup() {
4
5     int ledNumber;
6     for (ledNumber=1; ledNumber<5; ledNumber++){
7         bob3.setLed(ledNumber, WHITE);
8     }
9
10 }
11
12 void loop() {
13 }
14 }
15 }
```

Das Programm schaltet unter Verwendung einer for-Schleife nacheinander alle vier LEDs am BOB3 weiß ein:

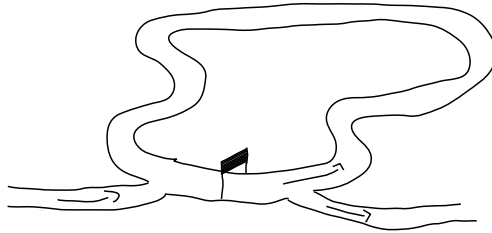


Die Schüler lernen, welche Bestandteile zur **Parameterliste** einer for-Schleife gehören und wie diese verwendet werden. Sie lernen, wie eine **Laufvariable** verwendet wird und wenden ihr zuvor erworbenes Wissen über **Bedingungsprüfungen** erneut vertieft an. In diesem Zusammenhang lernen sie Ausdrücke/Operatoren wie z.B. **ledNumber++** kennen und anwenden.

Die Schüler **experimentieren** mit dem Programm und beobachten am BOB3, welche Quellcode-Änderungen bestimmte Aktionen hervorrufen. In der nächsten Tutorial-Einheit wird das Programm-Beispiel weiter verändert, wobei jetzt auch Ausdrücke wie z.B. **ledNumber--** verwendet werden. Die Schüler erhalten vom Tutorial die Aufgabe, die Parameter der for-Schleife so abzuändern, dass nur die Augen weiß eingeschaltet werden. Hierbei zeigt sich, ob das Konzept der for-Schleife verstanden wurde.

Die Schüler bearbeiten das **Arbeitsblatt 8**:

„Eine for-Schleife dient zur wiederholten Durchführung“



Die Kontrollstruktur **for-Schleife** ist eine **Zähl-Schleife**. Sie ermöglicht, dass in Abhängigkeit von einer **Bedingung** bestimmte Anweisungen **solange immer wieder** ausgeführt werden, bis die Bedingung **nicht mehr** erfüllt ist.

```
for (Initialisierung; Bedingung; Aktualisierung) {  
    Anweisungen;  
}
```

Die Parameterliste einer for-Schleife besteht aus drei Teilen:

- Initialisierung** → Die Laufvariable wird auf einen Anfangswert gesetzt
(Dies erfolgt nur bei der ersten Ausführung der Schleife!)
- Bedingungsprüfung** → Die Bedingung legt das Abbruchkriterium der Schleife fest
- Aktualisierung** → Ende eines Durchlaufs: die Laufvariable wird aktualisiert

Die SuS setzen sich ausführlich mit der **Theorie** der for-Schleife auseinander. Sie vertiefen ihr experimentell erworbenes Wissen über die **Initialisierung** der Laufvariablen, der **Bedingungsprüfung** und der **Aktualisierung**. Das Arbeitsblatt bespricht die Vorgänge anhand eines konkreten Programmier-Beispiels. Dabei wird der **erste Schleifendurchlauf** Schritt für Schritt analysiert und im Detail erklärt. Anschließend wird erklärt, was in den folgenden Schleifendurchläufen passiert und wann und warum die Schleife schließlich abgebrochen wird.

Die Schüler bearbeiten die **Aufgaben 1 bis 6** des Arbeitsblatts. Dabei setzen sie sich unter anderem noch einmal mit den Laufvariablen auseinander:

Aufgabe 5 Betrachte die folgende for-Schleife. Welche Werte nimmt die Variable **i** im jeweiligen Durchlauf an?

```
6   for (i=0; i<10; i=i+2){
7       ...
8   }
```

Durchlauf 1: **i**=_____

Durchlauf 2: **i**=_____

Durchlauf 3: **i**=_____

Durchlauf 4: **i**=_____

Durchlauf 5: **i**=_____

In der **elften Tutorial-Einheit** programmieren die Schüler ein neues Programm, das ebenfalls eine for-Schleife verwendet. Das Programm lässt die LEDs am BOB3 nacheinander weiß blinken:

```
1 #include <BOB3.h>
2
3 void setup() {
4
5 }
6
7 void loop() {
8
9   int ledNumber;
10  for (ledNumber=1; ledNumber<5; ledNumber++){
11      bob3.setLed(ledNumber, WHITE);
12      delay(350);
13      bob3.setLed(ledNumber, OFF);
14      delay(150);
15  }
16 }
17 }
18
```

Die Schüler testen das Programm am BOB3 und besprechen, was das Programm macht. Anschließend bekommen sie die Aufgabe, den Programmcode so zu verändern, dass keine for-Schleife verwendet wird, am Roboter jedoch dasselbe passiert wie vorher. Die Schüler sollen so möglichst den Sinn und Zweck von for-Schleifen verstehen.

Zum Abschluss bearbeiten die Schüler noch die **zwölfte und letzte Tutorial-Einheit** des „Intro II“-Tutorials. Es wird ein Programm mit zwei **verschachtelten** for-Schleifen präsentiert, in das Fehler eingebaut wurden. Die Schüler erlernen, **Fehler im Programmcode** zu entdecken und zu beheben:

The screenshot shows the ProgBob IDE interface. At the top, the title bar reads "ProgBob" and "bob3.org". The main window is divided into several sections:

- Code Editor:** Contains C++ code for a Bob3 board. Lines 11-15 are highlighted in red, indicating errors. The code is:


```

1 #include <BOB3.h>
2
3 void setup() {
4
5 }
6
7 void loop() {
8
9   int time, ledNumber;
10  for (time=50; time<400; time*=2) {
11    for (ledNumber=1; ledNumber<5; led++) {
12      bob3.setLed(ledNumber, WHITE);
13      delay(time);
14      bob3.setLed(ledNumber, OFF);
15      Delay(time);
16    }
17  }
18
19 }

```
- Task Description:** A dashed box contains the text:

__aufgabe__

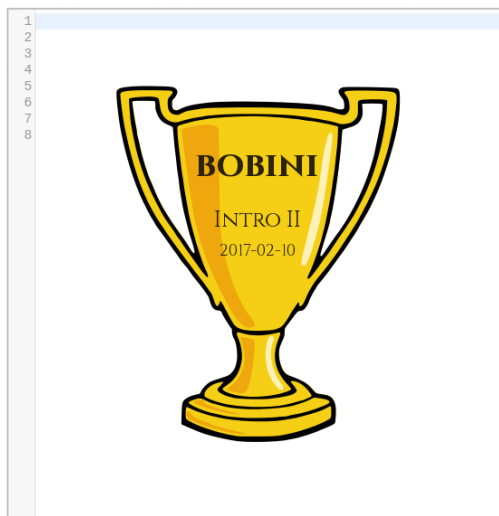
Im neuen Programm-Beispiel sind vier **Fehler** versteckt!

Findest du sie?

Nach dem **▶** Compilieren kannst du weitere Informationen zu den Fehlern bekommen, indem du auf **Bugs detected!** klickst!

Übertrage dein verbessertes Programm auf den BOB3!
- Tutorials:** A list of tutorials on the right side:
 - Intro I ✓
 - Intro II ✓
 - Intro III ✓
 - Sense
 - Touch
 - Color
 - Comm
 - Bugs
 - free
- Bottom Bar:** Contains a green "Compile" button, a red "Bugs detected!" button, and a yellow "Next chapter" button.

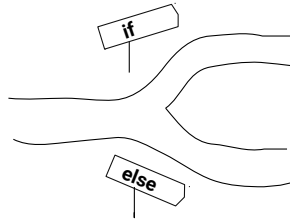
Geschafft!! Die Schüler haben nun auch das „Intro II“-Tutorial erfolgreich beendet!!



Lösungen

Arbeitsblatt 5 - if-else-Anweisung

Eine if-else-Anweisung ist eine Verzweigung:



Die Kontrollstruktur **if-else** ermöglicht, dass in Abhängigkeit von einer **Bedingung** bestimmte Anweisungen ausgeführt werden und andere dagegen nicht! **Falls (if)** die Bedingung **wahr** ist, dann werden die **Anweisungen1** ausgeführt, **sonst (else)**, also wenn die Bedingung **falsch** ist, werden die **Anweisungen2** ausgeführt:

```
if (Bedingung) {
    Anweisungen1;
} else {
    Anweisungen2;
}
```

Aufgabe 1: Was macht der BOB3, wenn du das folgende Programm überträgst?

```
4  if (1000 == 1000) {
5      bob3.setEyes(WHITE, WHITE);
6  }
7  else {
8      bob3.setEyes(OFF, OFF);
9  }
```

Auge 1: **weiß**

Auge 2: **weiß**

Aufgabe 2: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (3 == 8) {
5      bob3.setEyes(OFF, OFF);
6  }
7  else {
8      bob3.setEyes(ORANGE, WHITE);
9  }
    
```

Auge 1: **orange**

Auge 2: **weiß**

Aufgabe 3: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (3 != 3) {
5      bob3.setEyes(OFF, OFF);
6  }
7  else {
8      bob3.setEyes(WHITE, WHITE);
9  }
    
```

Auge 1: **weiß**

Auge 2: **weiß**

Aufgabe 4: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (3 != 8) {
5      bob3.setEyes(OFF, OFF);
6  }
7  else {
8      bob3.setEyes(WHITE, WHITE);
9  }
    
```

Auge 1: **aus**

Auge 2: **aus**

Aufgabe 5: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (3 < 8) {
5      bob3.setEyes(SEAGREEN, PURPLE);
6  }
7  else {
8      bob3.setEyes(ORANGE, WHITE);
9  }
    
```

Auge 1: **seagreen**

Auge 2: **purple**

Aufgabe 6: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (8 <= 8) {
5      bob3.setEyes(SEAGREEN, PURPLE);
6  }
7  else {
8      bob3.setEyes(ORANGE, WHITE);
9  }
    
```

Augen 1: **seagreen**

Augen 2: **purple**

Aufgabe 7: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (1001 < 1000) {
5      bob3.setEyes(RED, OFF);
6  }
7  else {
8      bob3.setEyes(OFF, RED);
9  }
    
```

Augen 1: **aus**

Augen 2: **rot**

Aufgabe 8: Was macht der BOB3, wenn du das folgende Programm überträgst?

```

4  if (1001 >= 1000) {
5      bob3.setEyes(RED, OFF);
6  }
7  else {
8      bob3.setEyes(OFF, RED);
9  }
    
```

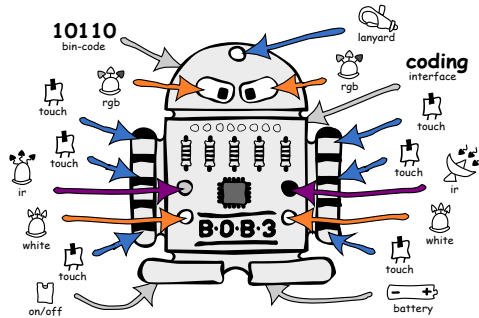
Augen 1: **rot**

Augen 2: **aus**

Lösungen

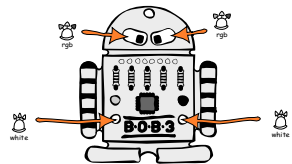
Arbeitsblatt 6 - Bob's Hardware

Aus welchen Bestandteilen besteht der BOB3?



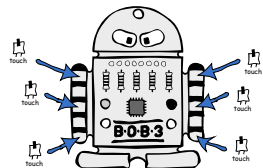
Aufgabe 1: Wie viele **Leuchtdioden** hat der BOB3, die **weiß** leuchten können?
Kreuze die richtige Antwort an:

- fünf
- eine
- zwei
- drei
- vier
- keine



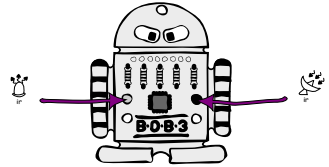
Aufgabe 2: Wo ist der BOB3 **berührungsempfindlich**?
Kreuze die richtige Antwort an:

- an den Armen
- am Kopf
- am Bauch
- an den Füßen
- nirgendwo
- an den LEDs



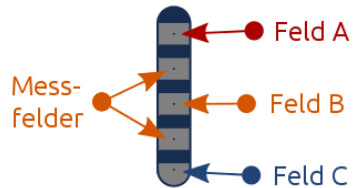
Aufgabe 3: Womit kann der BOB3 einem anderen Bob eine Botschaft senden? Kreuze die richtige Antwort an:

- mit dem Kopf
- mit den RGB-LEDs
- mit den weißen LEDs
- mit dem IR-Empfänger
- mit der IR-Sende-LED
- mit den Füßen



Aufgabe 4: Wie viele **Tastsensoren** hat der BOB3 insgesamt? Also an wie vielen **unterschiedlichen Stellen** kann der Bob Berührungen bemerken? Kreuze die richtige Antwort an:

- ein
- zwei
- vier
- sechs
- acht
- zehn



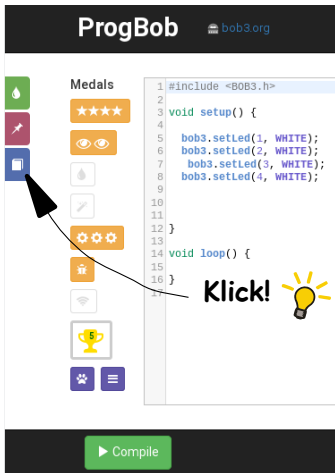
Aufgabe 5: Wie kann der BOB3 **nah** und **fern** unterscheiden? Kreuze die richtige Antwort an:

- er misst die Temperatur
- er leuchtet mit den hellen weißen LEDs und misst, wie weit er gucken kann
- mit dem IR-Sensor, per Reflektionsverfahren
- er weiß es nicht, er würfelt
- er fragt den Mikrocontroller
- er fragt einen anderen BOB3

Lösungen

Arbeitsblatt 7 - Bob's Software

Aus welchen Bestandteilen besteht die Bibliothek von BOB3?



Bibliothek

BOB3 - Methoden:

```
void bob3.setLed(id, color)
void bob3.setEyes(color1, color2)
void bob3.setWhiteLeds(status1, status2)
int bob3.getLed(id)
int bob3.getArm(id)
void bob3.enableArms(enable)
int bob3.getIRSensor()
int bob3.getIRLight()
void bob3.enableIRSensor(enable)
int bob3.getTemperature()
int bob3.getMillivolt()
int bob3.getID()
int bob3.receiveMessage(timeout)
void bob3.transmitMessage(message)
```

Globale Funktionen:

```
void delay(milliseconds)
int mixColor(color1, color2, w1, w2)
int rgb(red, green, blue)
void remember(value)
int recall()
```

Wenn man in der Programmieroberfläche links neben den Medaillen auf den blauen Slide „Bibliothek“ klickt, dann bekommt man eine **Übersicht** über die **verschiedenen Methoden**, die man auf den BOB3 anwenden kann.

Hier sind alle **Funktionen** zu finden, die schon **fertig implementiert** sind und einfach verwendet werden können, wie z.B. die Funktion **delay(500)**, die man einfach aufrufen und damit eine Verzögerung von 500 Millisekunden erzeugen kann.

Funktionen, die sich auf ein **Objekt** beziehen, nennt man **Methoden**. Da der BOB3 im programmiertechnischen Sinne ein Objekt ist, heißen seine Funktionen Methoden.

Aufgabe 1: Wie viele **Parameter** hat die Funktion **delay(milliseconds)**?

Kreuze die richtige Antwort an:

- keinen
- einen
- zwei
- eintausend

Aufgabe 2: Wie viele **Parameter** hat die Methode **bob3.getTemperature()**?

Kreuze die richtige Antwort an:

- keinen
- einen
- zwei
- eintausend

Aufgabe 3: Kann man **bob3.setLed(Auge1, ORANGE)** schreiben, um das Auge 1 vom BOB3 orange einzuschalten?

Kreuze die richtige Antwort an:

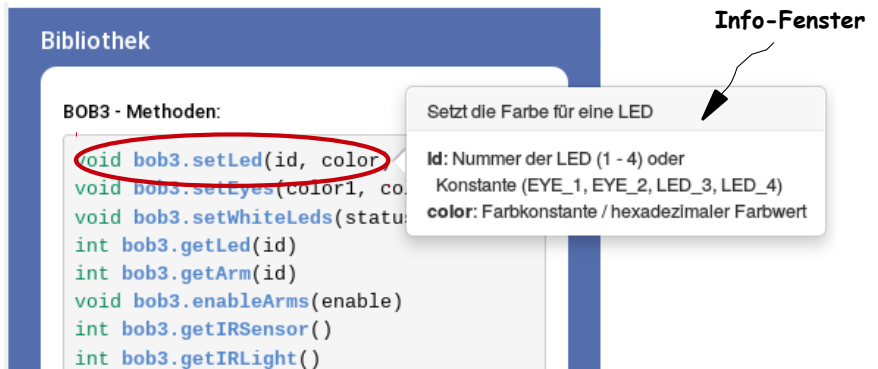
- Ja, das klappt!
- Nein, das funktioniert nicht!
- Nein, das klappt nur manchmal...

Aufgabe 4: Kann man **bob3.setLed(1, ORANGE)** schreiben, um das Auge 1 vom BOB3 orange einzuschalten?

Kreuze die richtige Antwort an:

- Ja, das klappt!
- Nein, das funktioniert nicht!
- Nein, das klappt nur manchmal...

Wenn man mit dem Mauszeiger **auf einer Methode wartet**, zeigt ein Info-Fenster alle Details zu der Methode an:



Aufgabe 5: Was darf man bei der Methode `bob3.setWhiteLeds(status1, status2)` für den Parameter `status1` einsetzen?
 Kreuze die richtigen Antworten an, es sind mehrere Antworten richtig:

- true
- 0
- ON
- false
- WHITE
- OFF
- 1

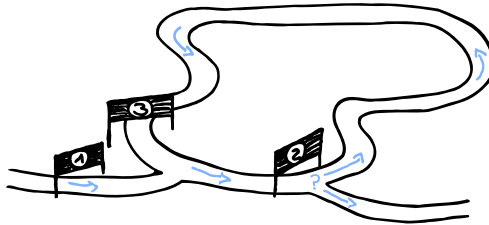
Aufgabe 6: Was liefert die Methode `bob3.getTemperature()`, wenn man sie aufruft?
 Kreuze die richtige Antwort an:

- entweder eine 0 oder eine 1
- die aktuelle Temperatur als Zahlenwert zwischen 0 und 255
- die aktuelle Temperatur als Wort, z.B. kalt, kälter, warm...
- da die Methode keinen Parameter hat, liefert sie nichts!

Lösungen

Arbeitsblatt 8 - for-Schleife

Eine for-Schleife dient zur wiederholten Durchführung:



Die Kontrollstruktur **for-Schleife** ist eine **Zähl-Schleife**. Sie ermöglicht, dass in Abhängigkeit von einer **Bedingung** bestimmte Anweisungen **solange immer wieder** ausgeführt werden, bis die Bedingung **nicht mehr** erfüllt ist.

```
for (Initialisierung; Bedingung; Aktualisierung) {
    Anweisungen;
}
```

Aufgabe 1: Betrachte das folgende Programm. Wie oft wird die Anweisung **bob3.setLed(ledNummer, WHITE);** ausgeführt?

```
5 int ledNummer;
6 for (ledNummer=1; ledNummer<5; ledNummer++){
7     bob3.setLed(ledNummer, WHITE);
8 }
```

- einmal
- viermal
- zweimal
- fünfmal
- dreimal

Aufgabe 2: Betrachte das folgende Programm. Wie oft wird die Anweisung **bob3.setLed(ledNumber, WHITE);** ausgeführt?

```

5   int ledNumber;
6   for (ledNumber=1; ledNumber<4; ledNumber++){
7       bob3.setLed(ledNumber, WHITE);
8   }
    
```

- einmal
- zweimal
- dreimal
- viermal
- fünfmal

Aufgabe 3: Betrachte das folgende Programm. Wie oft wird die Anweisung **bob3.setLed(ledNumber, WHITE);** ausgeführt?

```

5   int ledNumber;
6   for (ledNumber=3; ledNumber<4; ledNumber++){
7       bob3.setLed(ledNumber, WHITE);
8   }
    
```

- einmal
- zweimal
- dreimal
- viermal
- fünfmal

Aufgabe 4 Betrachte das folgende Programm. Wie oft wird die Anweisung **bob3.setLed(ledNumber, WHITE);** ausgeführt?

```

5   int ledNumber;
6   for (ledNumber=4; ledNumber>0; ledNumber--){
7       bob3.setLed(ledNumber, WHITE);
8   }
    
```

- einmal
- zweimal
- dreimal
- viermal
- fünfmal

Aufgabe 5 Betrachte die folgende for-Schleife. Welche Werte nimmt die Variable **i** im jeweiligen Durchlauf an?

```

6   for (i=0; i<10; i=i+2){
7       ...
8   }
```

Durchlauf 1: **i= 0**

Durchlauf 2: **i= 2**

Durchlauf 3: **i= 4**

Durchlauf 4: **i= 6**

Durchlauf 5: **i= 8**

Aufgabe 6 Betrachte die folgende for-Schleife. Welche Werte nimmt die Variable **j** im jeweiligen Durchlauf an?

```

6   for (j=1; j<64; j=j*2){
7       ...
8   }
```

Durchlauf 1: **j= 1**

Durchlauf 2: **j= 2**

Durchlauf 3: **j= 4**

Durchlauf 4: **j= 8**

Durchlauf 5: **j= 16**

Durchlauf 6: **j= 32**

