

GEFÖRDERT VOM



Bundesministerium
für Bildung
und Forschung



Tag 1

1

Blitzlicht!

smile

Informatik für Schülerinnen SMART – FUTURE - ME

- Einblick in die Informatik
- Workshops - Learning by doing
- Science Slams
- Berufsorientierung durch Kontakte und Vorbilder

Kurs Überblick

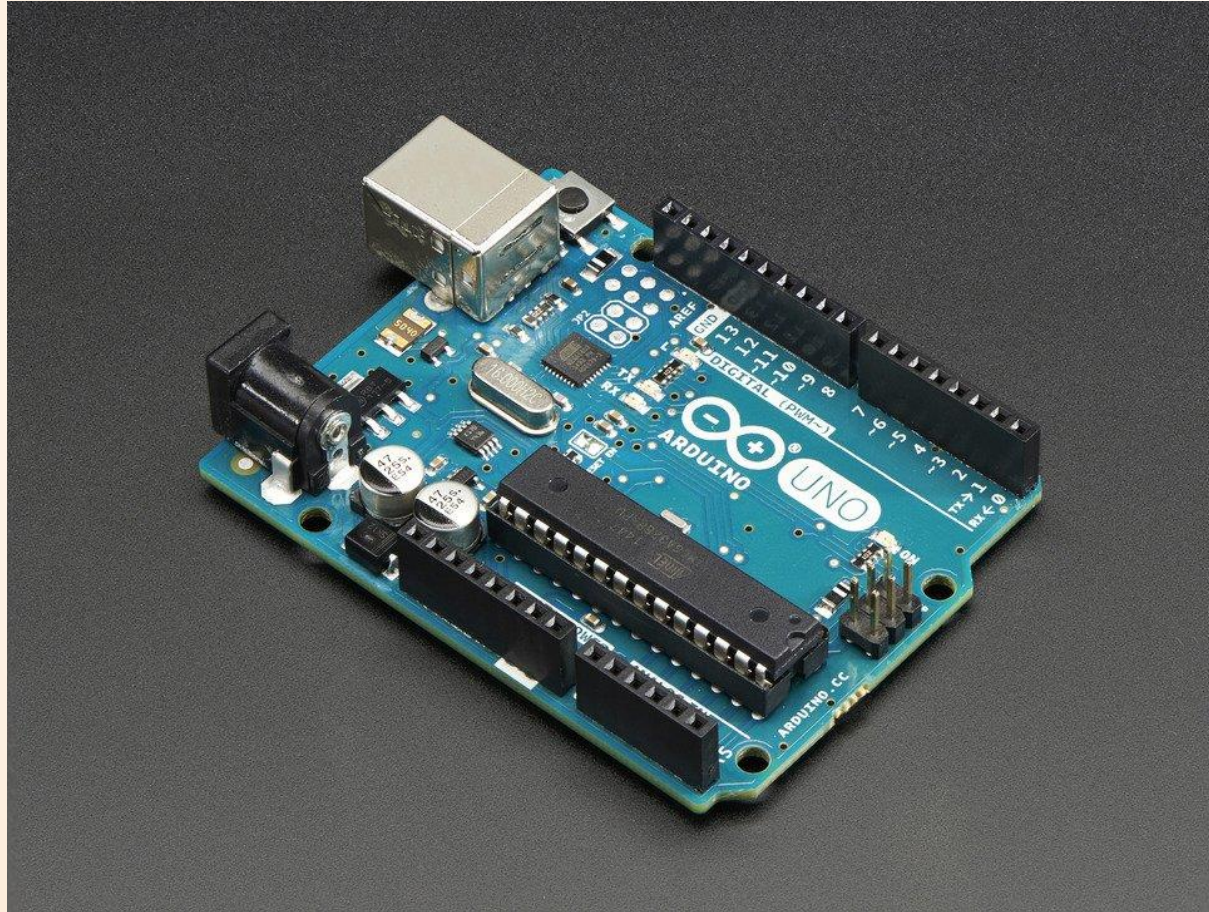
- Tag 1
 - Einführung
 - Themen Vorstellung!
 - HandsOn Arduino
- Tag 2
 - Sensoren und Aktoren!
 - Prototyp Planung
- Tag 3
 - Exkurs
- Tag 4
 - Domain (Themen) Erschließung
 - PROTOTYPING
 - PROTOTYPING
- Tag 5
 - Plakat
 - PRÄSENTATION

Pflanze mit Charakter!

- Eure Pflanze sagt euch, ob sie alles zum wohlfühlen hat!
- Was für eine Persönlichkeit hat eure Pflanze?
- Was braucht Sie?
- Wie kommuniziert sie mit Euch oder mit Anderen?
- Nehmt Euch in den Gruppen 5 Minuten Zeit, Euch über das Thema Gedanken zu machen, OHNE an technische Umsetzbarkeit zu denken und macht Euch Stichpunkte
- Danach Vorstellung in einer Gruppendiskussion



Arduino HandsOn



Arduino – Eine open source Prototyping Plattform



- Der Arduino ist ein Mini-Computer
- Mit dem Arduino lassen sich schnell und einfach Sensoren und LEDs oder Motoren ansteuern
- Durch den Arduino vereinfacht sich der Umgang mit Elektronik

Seeed Grove –
Sensoren (Licht, Temperatur, etc)
Aktoren (Motoren, Displays, LEDs, etc.)
Kabel zum Stecken statt löten



Steckboard

Analoge Steckplätze
A0 – A3

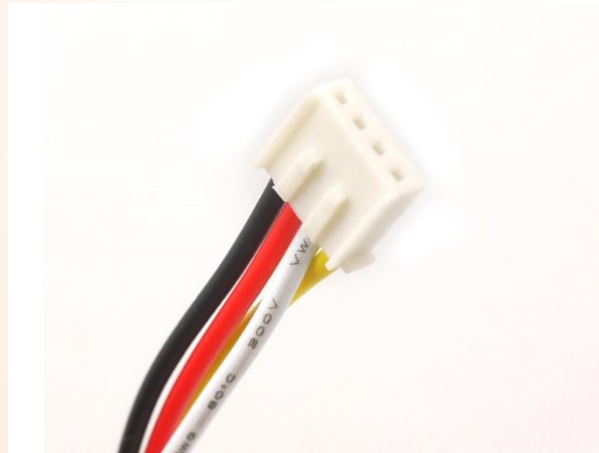
Digitale Steckplätze
D2 – D8



Display Steckplätze
I2C

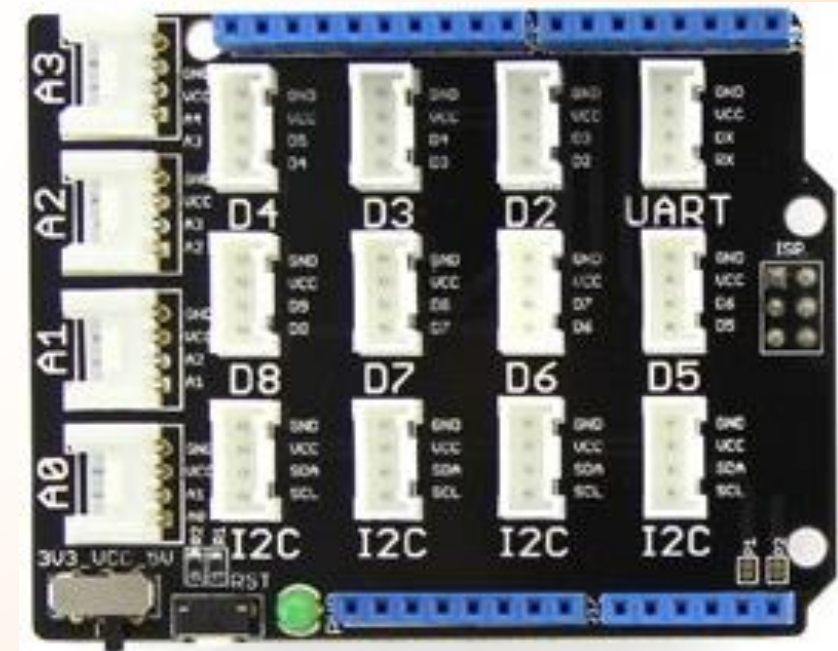
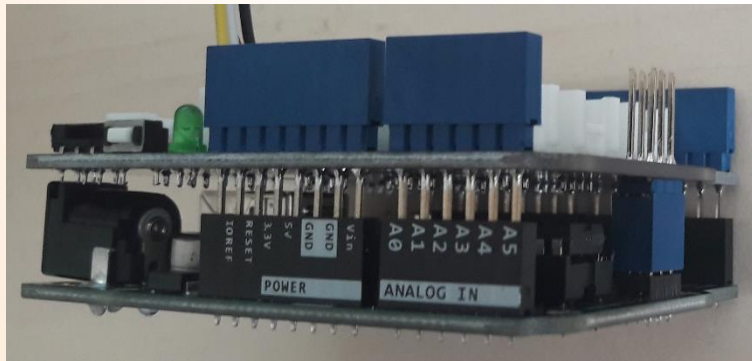
Kabel

- Die Steckkabel dienen zum Verbinden der einzelnen Komponenten (LEDs, Sensoren, ...)
- Die Stecker haben eine „Richtung“, zu erkennen an den „Nasen“ an einer Seite, die Gegenseite (Buchse) dazu die passenden Ausbuchtungen



Steckboard

- Steckt das Steckboard vorsichtig auf den Arduino auf, alle „Stifte“ müssen in die entsprechende Gegenseite
- Der kleine Schalter links unten muss auf 5V (rechte Schalterstellung) stehen



GEFÖRDERT VOM



Bundesministerium
für Bildung
und Forschung



Mein erstes Programm!

1

Einführung: Übersicht

- Was ist ein Programm?
- Was sind Befehle, Schleifen, Variablen und Bedingungen?
- Was ist grafische Programmierung?
- Und wie funktioniert das alles?

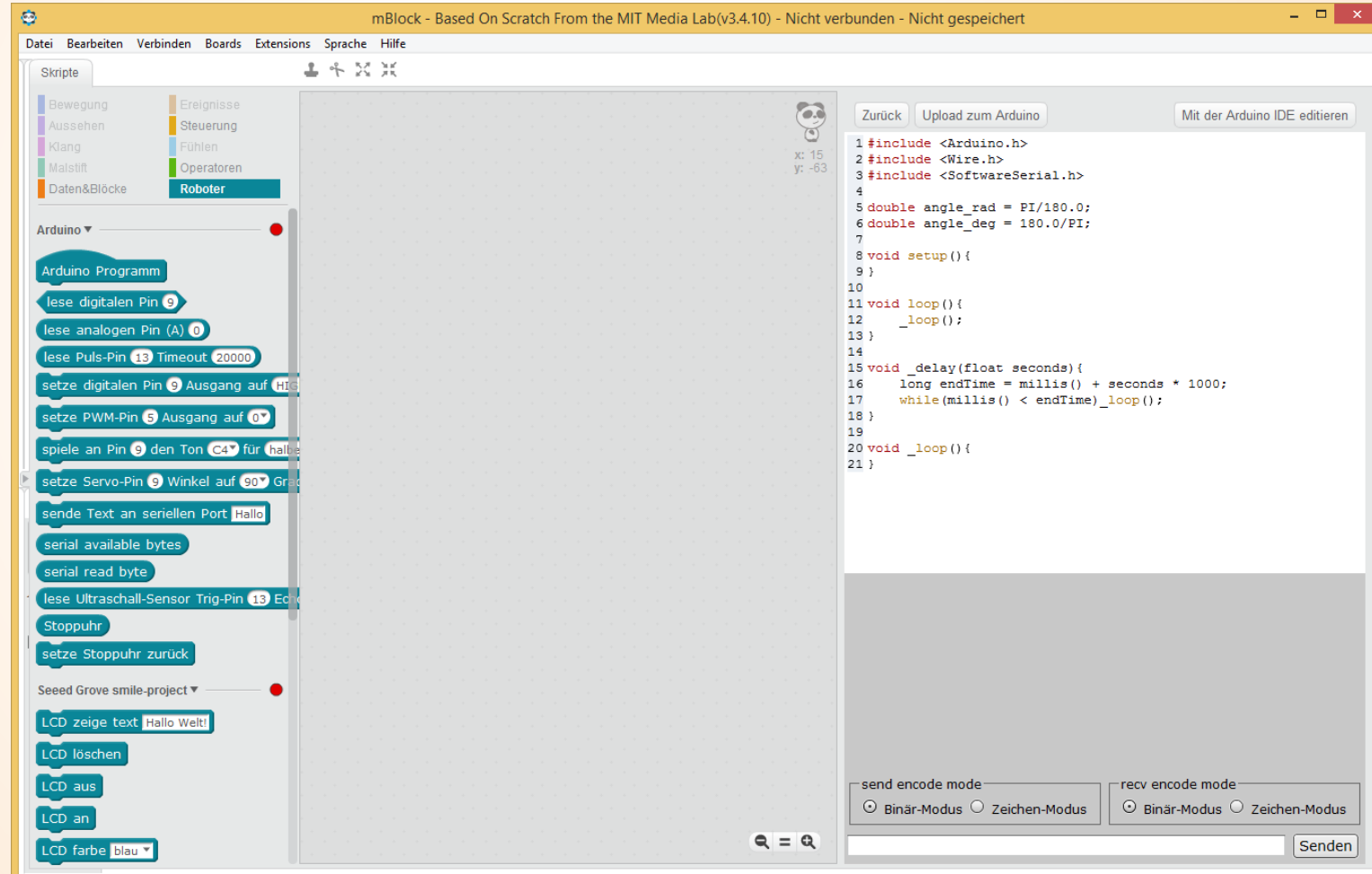
Einführung: Programme

- Was ist eigentlich ein Programm?
- Habt Ihr da Ideen?

Einführung: Programme

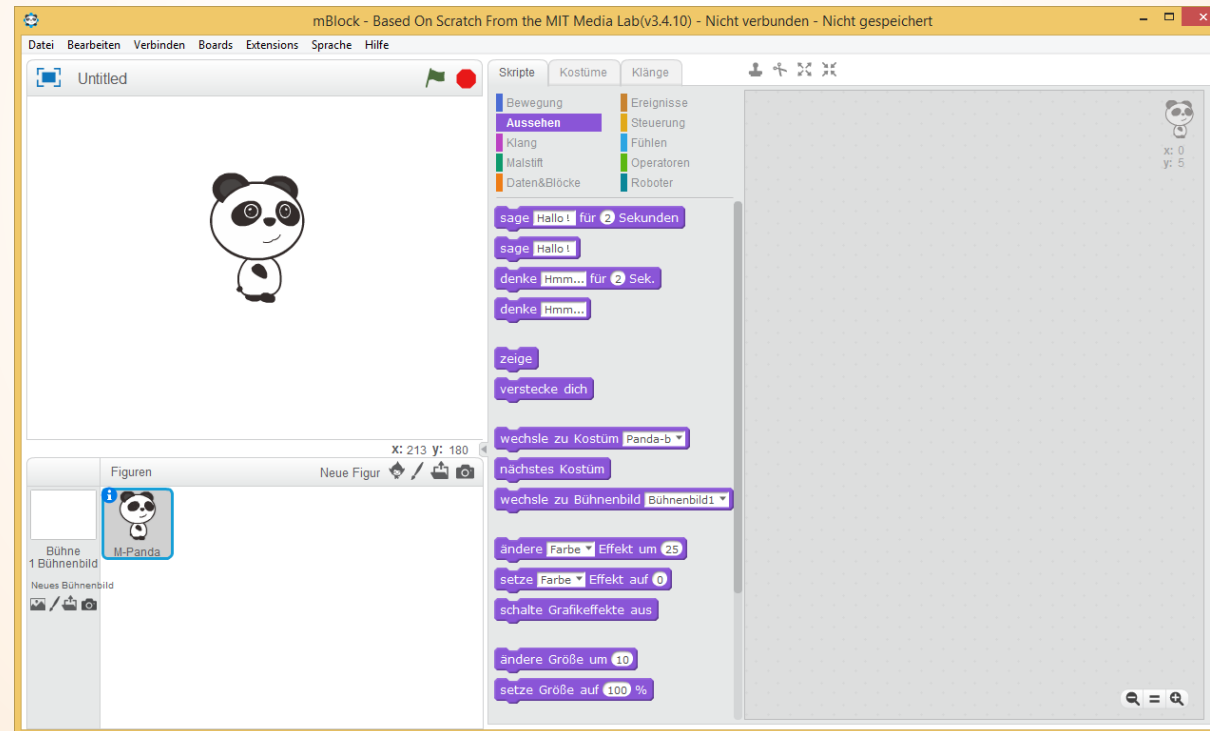
- ▶ Ein Programm ist eine Liste von Befehlen an einen Computer
- ▶ Wie ein Kochrezept oder eine Anleitung
- ▶ Die Befehle werden in der Reihenfolge abgearbeitet
- ▶ Oft sind Programmiersprachen aus simplen Befehlen aufgebaut, diese können aber kombiniert werden
- ▶ In vielen Programmiersprachen erkennst Du Befehle an den runden Klammern danach, Beispiele mp3-player: „play()“, „next()“, „stop()“

mBlock: Eine grafische Programmierumgebung (Scratch)



mBlock: Umschalten auf Arduino Modus

- Über „Bearbeiten“ im Menü gehen, dann „Arduino-Modus auswählen“



mBlock: Übersicht

The screenshot displays the mBlock software interface, which is based on Scratch. The main workspace is titled "mBlock - Based On Scratch From the MIT Media Lab(v3.4.8) - Nicht verbunden - Nicht gespeichert". The interface includes a menu bar (Datei, Bearbeiten, Verbinden, Boards, Extensions, Sprache, Hilfe), a left sidebar with category tabs (Skripte, Bewegung, Aussehen, Klang, Malstift, Daten&Blöcke, Ereignisse, Steuerung, Fühlen, Operatoren, Roboter), and a central workspace. The workspace contains an "Arduino Programm" block, which is highlighted with a red rectangle and labeled "Programmierbereich". This block contains a "wiederhole fortlaufend" loop with the following steps: "setze light auf lese analogen Pin (A) 0", "sende Text an seriellen Port light", and "warte 1 Sek.". To the right of the workspace is a code editor showing the corresponding C++ code for the Arduino program. The code includes headers for Arduino.h, Wire.h, and SoftwareSerial.h, and defines variables for angle and light. The setup function initializes the serial port and pin mode, while the loop function reads the analog pin, prints the value, and delays. The bottom of the interface features a terminal window showing the output of the program, including the serial data "100% 0" and the verification status "avrdude: 5352 bytes of flash verified".

Arduino Programm

wiederhole fortlaufend

setze light auf lese analogen Pin (A) 0

sende Text an seriellen Port light

warte 1 Sek.

Programmierbereich

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7 double light;
8
9 void setup() {
10   Serial.begin(115200);
11   pinMode(A0+0, INPUT);
12 }
13
14 void loop() {
15   light = analogRead(A0+0);
16   Serial.println(light);
17   _delay(1);
18   _loop();
19 }
20
21 void _delay(float seconds) {
22   long endTime = millis() + seconds * 1000;
23   while(millis() < endTime) _loop();
24 }
25
26 avrdude: load data flash data from input file C:\Users\ab\AppData\Local\Temp\build1881702473048
27 avrdude: input file C:\Users\ab\AppData\Local\Temp\build1881702473048
28 avrdude: reading on-chip flash data:
29
30 Reading | ##### | 100% 0
31
32 avrdude: verifying ...
33 avrdude: 5352 bytes of flash verified
34
35 avrdude done. Thank you.
```

mBlock: Übersicht

The screenshot displays the mBlock software interface, which is based on Scratch. The window title is "mBlock - Based On Scratch From the MIT Media Lab(v3.4.8) - Nicht verbunden - Nicht gespeichert". The interface includes a menu bar (Datei, Bearbeiten, Verbinden, Boards, Extensions, Sprache, Hilfe), a toolbar, and a main workspace. On the left, a "Skripte" (Scripts) panel is highlighted with a red box, containing a "Block-Palette" (Block Palette) with various Arduino-related blocks. The central workspace shows a Scratch script for an Arduino program, starting with "wiederhole fortlaufend" (loop forever), followed by "setze light auf lese analogen Pin (A) 0" (set light to read analog pin A0), "sende Text an seriellen Port light" (send text to serial port light), and "warte 1 Sek." (wait 1 second). On the right, a code editor shows the corresponding C++ code for the Arduino program, including headers, setup, and loop functions. The code is as follows:

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7 double light;
8
9 void setup() {
10   Serial.begin(115200);
11   pinMode(A0+0, INPUT);
12 }
13
14 void loop() {
15   light = analogRead(A0+0);
16   Serial.println(light);
17   _delay(1);
18   _loop();
19 }
20
21 void _delay(float seconds) {
22   long endTime = millis() + seconds * 1000;
23   while(millis() < endTime) _loop();
24 }
```

Below the code editor, a terminal window shows the output of the program, including the text "Reading | ##### | 100% 0" and "avrduide: verifying ...". At the bottom, there are buttons for "send encode mode" and "recv encode mode", each with radio buttons for "Binär-Modus" (Binary Mode) and "Zeichen-Modus" (Text Mode), and a "Senden" (Send) button.

mBlock: Übersicht

The screenshot displays the mBlock software interface, which is based on Scratch from the MIT Media Lab (v3.4.8). The window title is "mBlock - Based On Scratch From the MIT Media Lab(v3.4.8) - Nicht verbunden - Nicht gespeichert". The interface includes a menu bar (Datei, Bearbeiten, Verbinden, Boards, Extensions, Sprache, Hilfe), a "Skripte" panel on the left with categories like Bewegung, Aussehen, Klang, Malstift, Daten&Blöcke, Ereignisse, Steuerung, Fühlen, Operatoren, and Roboter, and a central workspace. In the workspace, an "Arduino Programm" block is selected, containing a "wiederhole fortlaufend" loop with three steps: "setze light auf lese analogen Pin (A) 0", "sende Text an seriellen Port light", and "warte 1 Sek.". On the right side, a red-bordered box highlights the "Arduino-Bereich", which contains a "Zurück" button, an "Upload zum Arduino" button, a link to "Mit der Arduino IDE editieren", and a text area showing the compiled C++ code. The code includes headers for Arduino.h, Wire.h, and SoftwareSerial.h, defines variables for angle and light, and implements a setup function to initialize the serial port and a loop function to read the analog pin and print the value. Below the code, there is a terminal window showing the output of the compilation and upload process, including messages like "avrdude: load data flash data from input file", "avrdude: input file", "avrdude: reading on-chip flash data:", "Reading | 100% 0", "avrdude: verifying ...", "avrdude: 5352 bytes of flash verified", and "avrdude done. Thank you.".

Arduino-Bereich

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7 double light;
8
9 void setup() {
10   Serial.begin(115200);
11   pinMode(A0+0, INPUT);
12 }
13
14 void loop() {
15   light = analogRead(A0+0);
16   Serial.println(light);
17   _delay(1);
18   _loop();
19 }
20
21 void _delay(float seconds) {
22   long endTime = millis() + seconds * 1000;
23   while(millis() < endTime) _loop();
24 }
25
26 avrdude: load data flash data from input file C:\Users\ab\AppData\Local\Temp\build1881702473048
27 avrdude: input file C:\Users\ab\AppData\Local\Temp\build1881702473048
28 avrdude: reading on-chip flash data:
29
30 Reading | ##### | 100% 0
31
32 avrdude: verifying ...
33 avrdude: 5352 bytes of flash verified
34
35 avrdude done. Thank you.
```


Übung

Das erste Programm



- Startet mBlock auf eurem Rechner
- Wechselt in den Arduino Modus (Bearbeiten -> Arduino Modus)
- Schließt das LCD an einen „I2C“ Steckplatz an
- Verbindet den Arduino und das Notebook über das Kabel
- Verbindet mBlock und den Arduino über (im Menü) „Verbinden“ -> „serieller Port“ -> „Com2“. Die Zahl hinter „Com“ kann unterschiedlich sein, das ist egal
- Erstellt ein erstes Arduino Programm
- Ladet es in den Arduino hoch
- „Arduino Programm“ findet ihr unter „Roboter“
- LCD Befehle auch



Übung

Das erste Programm

- Das Resultat sollte ungefähr so aussehen

Arduino Programm

LCD zeige text Hallo Welt!



Aktor RGB LCD (I2C)



Übung RGB LCD

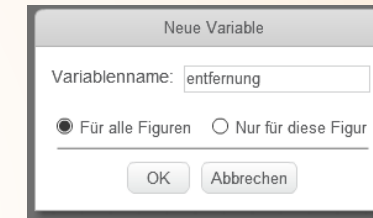
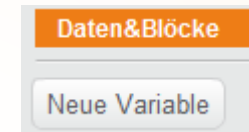
- Recherchiert:
 - Wofür steht LCD?
 - Wofür steht RGB?
- Ändert euer Programm:
 - Ändert den Text, den ihr ausgeben
 - Ändert die Hintergrundfarbe auf Rot, Grün, Blau
 - Wie viel Text passt auf das Display?
 - Verändert die Position des Textes mit „LCD Pos“
- Speichert euer Programm ab

Einführung: Variablen

- Eine Variable ist ein Platz, um Werte zu speichern
- Ähnlich wie ein Karton, in den ich etwas hinein tun kann
- Dieser Karton hat einen eindeutigen Namen, also nur er heißt so
- Analog zu Variablen in der Mathematik (x, \dots)
- Gebt den Variablen „sprechende“ Namen, damit ihr nächste Woche noch wisst, was die Variable enthält („abstand“ statt „a“)
- Bei mBlock können Variablen **NUR** Zahlen enthalten (keinen Text)

Einführung: Variablen anlegen

- Anlegen eine Variablen im Punkt Daten&Blöcke
- Dabei muss der Variable ein Name gegeben werden (Bitte nicht nur a oder x)
- Danach erscheinen neue Befehle unter Daten & Blöcke für diese Variable



Einführung: Variablen anlegen

- Variablen können gelesen und beschrieben werden
- Zum setzen einer Variable Nutzt das „setze auf x“
- Variablen können Zahlen oder Texte beinhalten
- Zum Lesen zieht die Variable in einen andere Operation rein

The image shows a block editor interface with two panels. The left panel, titled 'Daten&Blöcke', contains a 'Neue Variable' button and a list of variable operations: 'Wert' (checked), 'setze Wert auf 0', 'ändere Wert um 1', 'zeige Variable Wert', and 'verstecke Variable Wert'. The right panel shows a sequence of blocks: 'buzzer D4 play tone 1519 for dura', 'temperature sensor A0', 'sound sensor A0', 'rotary angle sensor A0', 'Piezo vibration sensor A0', 'button D4 pressed', 'touch sensor D4 pressed', '3-axis digital accelerometer x', 'Seed Grove smile-project', 'LCD zeige text Hallo Welt!', 'LCD löschen', 'LCD aus', 'LCD an', 'LCD Pos Zeile: 0 Spalte: 0', 'LCD farbe blau', and 'Chain LED LED0 r: 0 g: 0 b: 0'. Red arrows point from the 'setze Wert auf 0' and 'zeige Variable Wert' blocks in the left panel to the 'setze Wert auf 0' and 'LCD zeige text Hallo Welt!' blocks in the right panel, illustrating the flow of variable creation and usage.

Übung Variablen 1

- ▶ Ändert euer Programm:
 - ▶ Legt eine Variable an und weist dieser einen Wert zu
 - ▶ Gebt den Wert der Variable auf dem Display aus
 - ▶ Benutzt „verbinden“ um zusätzlich noch einen Text auszugeben

Lösung Übung Variablen 1

- Ändert euer Programm:
 - Legt eine Variable an und weist dieser einen Wert zu
 - Gebt den Wert der Variable auf dem Display aus
 - Benutzt „verbinden“ um zusätzlich noch einen Text auszugeben



Übung Variablen 2

- ▶ Ändert euer Programm:
 - ▶ Wartet nach der Ausgabe von „Wert“ für eine Sekunde
 - ▶ Erhöht dann den Wert der Variable „Wert“ um eins
 - ▶ Gebt die geänderte Variable aus
 - ▶ Wiederholt dieses drei mal

Lösung Übung Variablen 2

- Ändert euer Programm:
 - Wartet nach der Ausgabe von „Wert“ für eine Sekunde
 - Erhöht dann den Wert der Variable „Wert“ um eins
 - Gebt die geänderte Variable aus
 - Wiederholt dieses drei mal

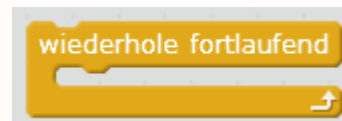


Übung Variablen

- Wofür braucht Ihr Variablen bei der Umsetzung eurer Pflanze?

Einführung Schleifen/Wiederholungen

- Wäre es nicht praktisch, die Programmteile immer wieder untereinander schreiben zu müssen?
- Dafür gibt es „Schleifen“ unter dem Punkt „Steuerung“
- Die Befehle Teil in der Schleife werden dann wiederholt (eine bestimmte Anzahl oder fortlaufend, die sogenannte „Endlosschleife“)



Übung Schleifen/Wiederholungen

- Ändert Euer Programm so ab, dass ihr anstatt der Wiederholung der Befehle eine Schleife verwendet. Ob 10 Wiederholungen oder fortlaufend ist dabei egal.

Lösung Übung Schleifen/Wiederholungen

- ▶ Ändert Euer Programm so ab, dass ihr anstatt der Wiederholung eine Schleife verwendet. Ob 10mal oder fortlaufend ist dabei egal



Übung Schleifen

- ▶ Wofür braucht Ihr Schleifen bei der Umsetzung eurer Pflanze?

Feedback Tag 1

➤ Was denkt Ihr?