

WOKWI simulierte IoT Projekte im Browser

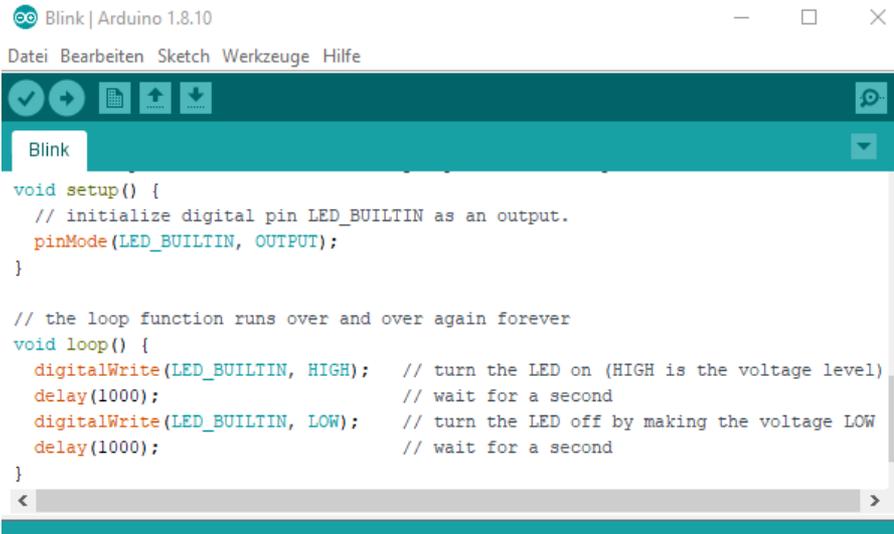
- Was will ich machen?
- Arduino IDE & Steckbrett
- WOKWI
- WOKWI Editor & Simulator
- Mikrokontroller & Programmiersprachen
- Schaltung 1: LED ein
- Teil einfügen
- Schaltung 2: Blinkende LED
- Schaltung 3: Taster & Monitor
- Schaltung 4: Taster & LED
- Schaltung 5: DHT22
- Schaltung 6: LCD
- Schaltung 7: PORTs
- Aufgabe 1: Pointer
- Vorteile, Nachteile
- Erfahrungen
- WOKWI im Internet I bis IV
- diagram.json I und II
- Informationen zu „part“
- Shortcuts Versuchsaufbau
- Shortcuts allgemein
- Sonstiges
- WOKWI Projekte

Was will ich machen?

- Vergleich WOKWI-Simulator und Arduino-IDE.
- Vor- und Nachteile einer Simulation.
- Für wen ist WOKWI interessant?
- Was kann der Sketch-Editor?
- Ist ein visualisierter Versuchsaufbau sinnvoll?
- Welche Mikrokontroller sind integriert?
- Ist WOKWI für Einsteiger geeignet?
- Welche Hilfen und Lernprogramme sind verfügbar?

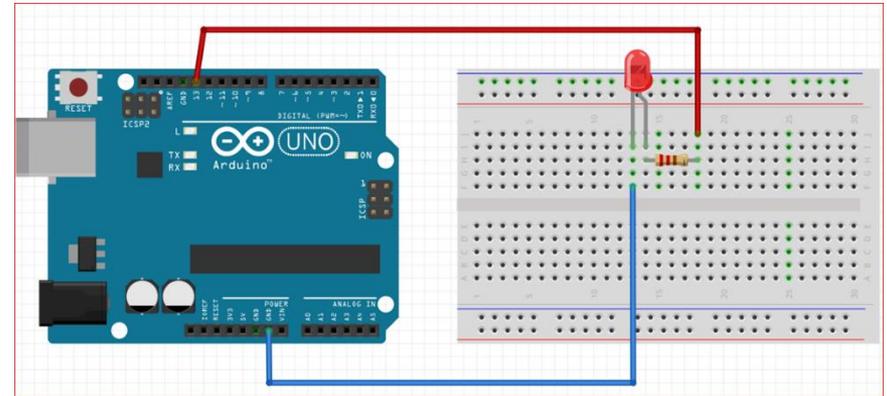
Arduino IDE & Steckbrett

Der Einstieg	Will man sich mit dem Arduino befassen und Erfahrungen mit Sensoren (Taster) und Aktoren (LED) sammeln, so steht zu Beginn ein Versuchsaufbau, gefolgt von der Programmierung mit der an den Arduino angepassten Programmiersprache C++.
Arduino-IDE	Die Arduino-IDE ist auf das notwendigste reduziert. Sie bietet beim Erstellen eines Sketches (Programm) wenig Unterstützung.
	Der Versuchsaufbau stellt eine weitere Hürde dar, da neben einem Steckbrett Aktoren und Sensoren benötigt werden.



```
Blink | Arduino 1.8.10
Datei Bearbeiten Sketch Werkzeuge Hilfe
Blink
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}
```

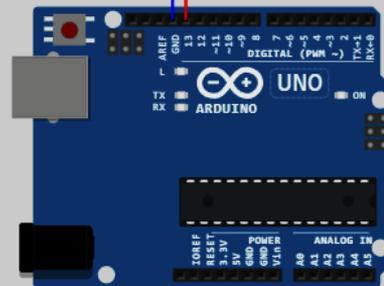
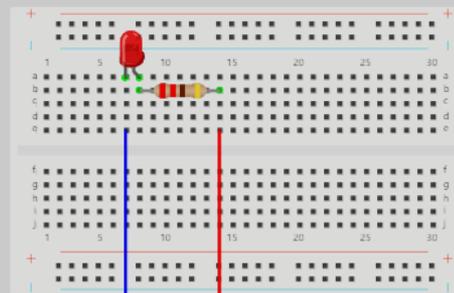




sketch.ino diagram.json Library Manager

```
1 void setup() {
2   // put your setup code here, to run once:
3   pinMode(13, OUTPUT);
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8   digitalWrite(13, HIGH);
9   delay(1000);
10  digitalWrite(13, LOW);
11  delay(1000);
12 }
13
```

Simulation



<https://wokwi.com>

WOKWI Editor & Simulator

Fenster	WOKWI besteht aus 2 Fenstern. Das linke Fenster beinhaltet den Sketch-Editor. Das rechte Fenster den Versuchsaufbau und die Simulation.
Tab „sketch.ino“	Der Editor kommt mit einer hinreichenden Funktionalität (siehe unten).
Tab „diagram.json“	Der Versuchsaufbau wird im JSON-Format gespeichert. Im linken Fenster, hinter dem Tab „diagram.json“, kann man Anordnung und Werte editieren. Hier werden z.B. die Werte eines Widerstandes eingestellt.
Tab „Library Manager“	Dient zum Hinzufügen weiterer Arduino-Bibliotheken.
Versuchsaufbau	Die graphische Darstellung des Versuchsaufbaus beinhaltet die Schaltung in Symbolen.
Simulation	 Über die Schaltfläche Simulation wird der Sketch ausgeführt und dessen Aktionen im Fenster Versuchsaufbau sichtbar.

Mikrokontroller & Programmiersprachen

Klicke auf

+ NEW PROJECT

Mikrokontroller
wählen durch klicken.

Arduino Uno

Arduino Mega

Arduino Nano

ATtiny85

ESP32

ESP32-S2

ESP32-S3 (beta)

ESP32-C3

MicroPython on ESP32

Raspberry Pi Pico

Raspberry Pi Pico (SDK)

MicroPython on Raspberry Pi Pico

CircuitPython on Raspberry Pi Pico

Franzininho

Franzininho WIFI (ESP32-S2)

Rust on ESP32

Rust on ESP32-S2

Rust on ESP32-C3

Rust on ESP32 Rust Board

Rust on ESP32 (nostd)

Rust on ESP32-S2 (nostd)

Rust on ESP32-C3 (nostd)

Rust on ESP32 Rust Board (nostd)

Programmiersprachen

Adruino C++, MicroPython, Rust

Schaltung 1: LED ein

Aufgabe

Eine LED einschalten.

Teile

Arduino

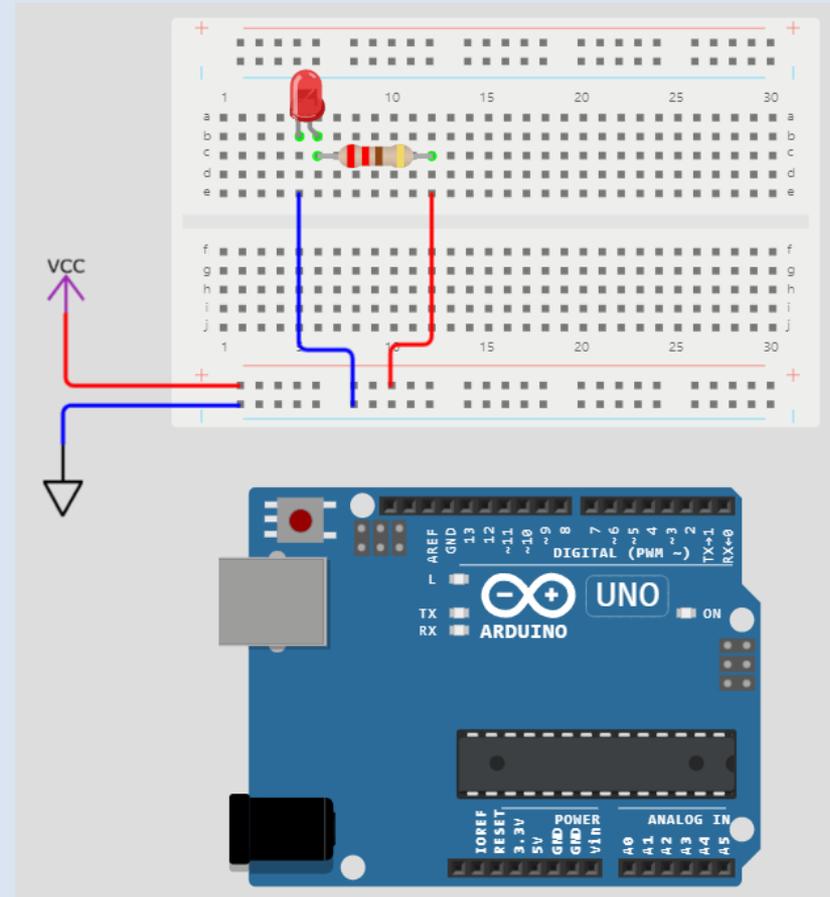
Breadboard

GND

VCC

LED

Widerstand 220 Ω

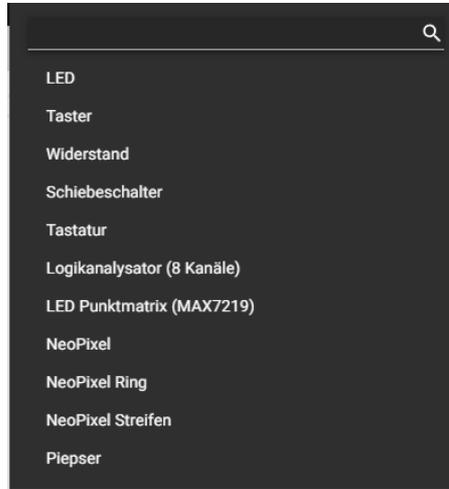


Teil einfügen

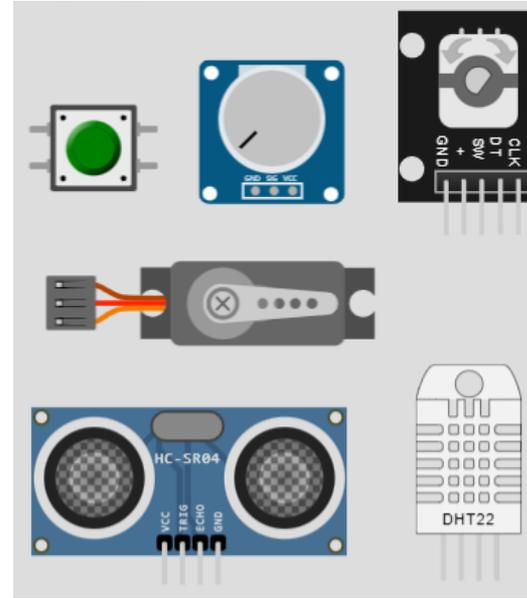
Klicken auf 



Klicken auf Teil



... und weitere



Informationen zum Teil

Klicken auf Teil & Fragezeichen.



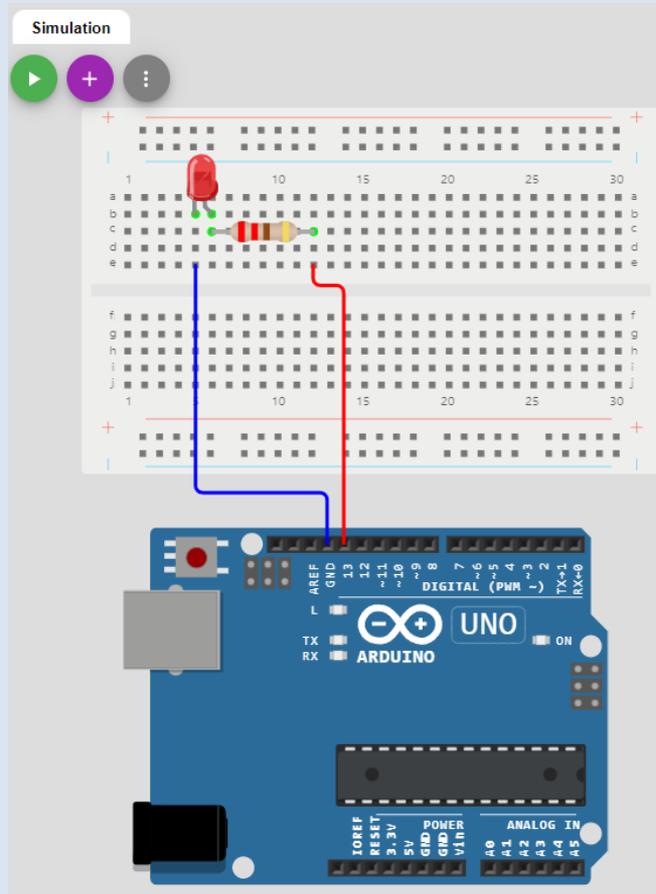
Schaltung 2: Blinkende LED

Aufgabe

Die LED soll regelmäßig an- und ausgehen.

Teile

Arduino
Breadboard
LED
Widerstand 220 Ω



Schaltung 2: Blinkende LED, Sketch

Aufgabe

Die LED soll regelmäßig an- und ausgehen.

```
sketch.ino ● diagram.json ● Library Manager ▼  
  
1 void setup() {  
2   // put your setup code here, to run once:  
3   pinMode(13, OUTPUT);  
4 }  
5  
6 void loop() {  
7   // put your main code here, to run repeatedly:  
8   digitalWrite(13, HIGH);  
9   delay(1000);  
10  digitalWrite(13, LOW);  
11  delay(1000);  
12 }  
13
```

Schaltung 2: Blinkende LED „diagram.json“

Aufgabe

Widerstandswert korrigieren.

Tab „diagram.json“

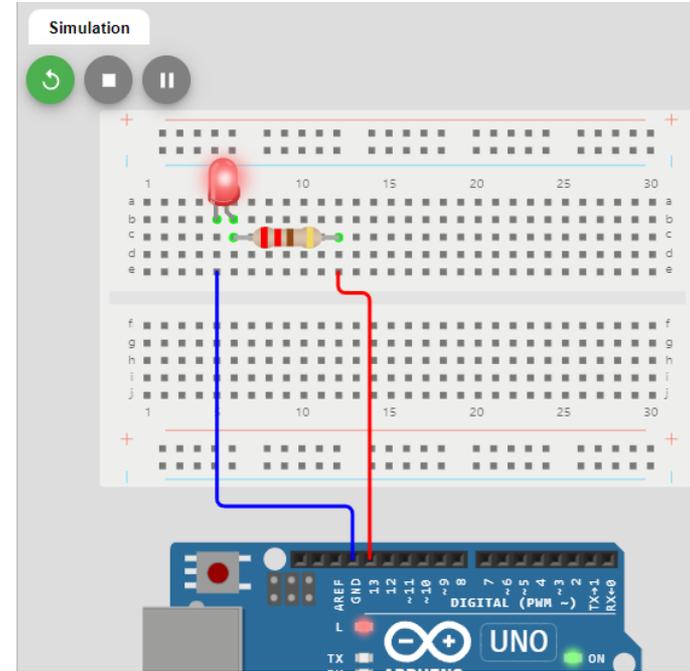
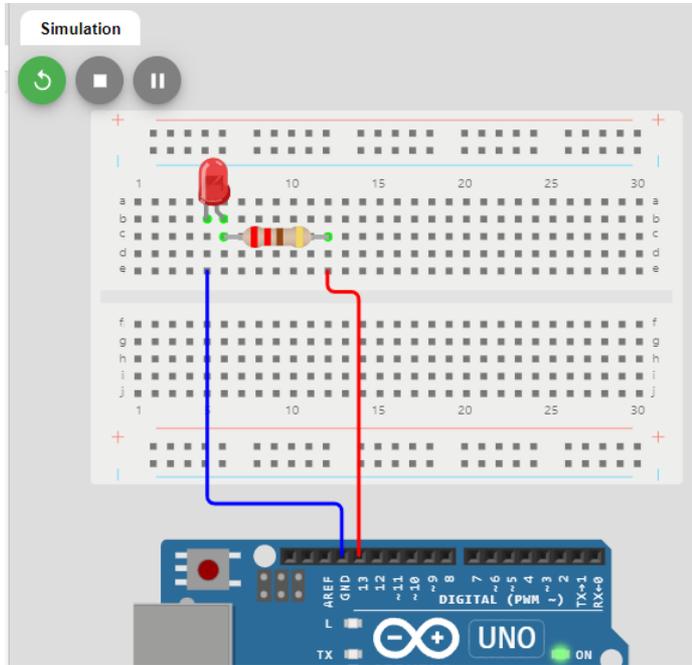
```
sketch.ino • diagram.json • Library Manager ▾  
  
1 {  
2   "version": 1,  
3   "author": "Enno Klatt",  
4   "editor": "wokwi",  
5   "parts": [  
6     { "type": "wokwi-arduino-uno", "id": "uno", "top": 52.46, "le  
7     { "type": "wokwi-breadboard-half", "id": "bb1", "top": -185.4  
8     {  
9       "type": "wokwi-led",  
10      "id": "led1",  
11      "top": -166.8,  
12      "left": -53.8,  
13      "attrs": { "color": "red" }  
14    },  
15    {  
16      "type": "wokwi-resistor",  
17      "id": "r1",  
18      "top": -125.2,  
19      "left": -28.8,  
20      "attrs": { "value": "220" }  
21    }  
22  ],  
23 }
```



Schaltung 2: Blinkende LED, Simulation

Aus

An



Schaltung 3: Taster & Monitor

Aufgabe

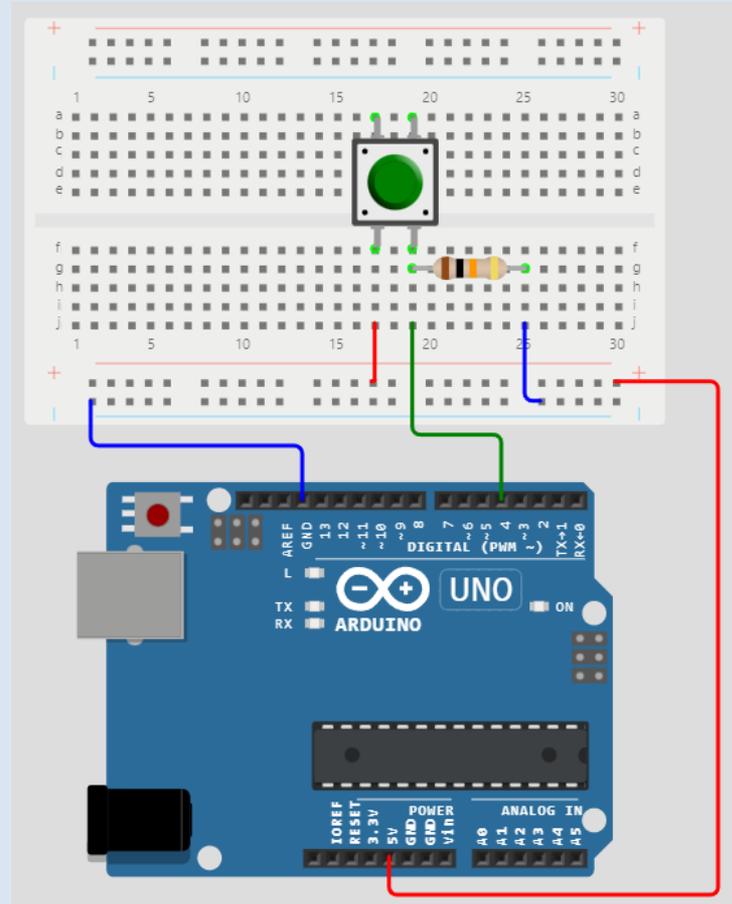
Taster gedrückt, dann
Nachricht.

Teile

Arduino
Taster
Widerstand 10 kΩ

diagram.json

```
16 {
17   "type": "wokwi-resistor",
18   "id": "r2",
19   "top": -0.4,
20   "left": 163.2,
21   "attrs": { "value": "10000" }
22 }
```



Schaltung 3: Taster & Monitor, Sketch

Aufgabe

Taster gedrückt, dann Nachricht..

sketch.ino ●

diagram.json ●

Library Manager ▼

```
1 void setup() {
2   // put your setup code here, to run once:
3   Serial.begin(9600);
4   Serial.println("Taster testen.");
5   pinMode(4, INPUT);
6 }
7
8 void loop() {
9   // put your main code here, to run repeatedly:
10  if (digitalRead(4)){
11    Serial.println("Nachricht gesendet");
12  }
13  delay(200);
14 }
15
```

Schaltung 4: Taster & LED

Aufgabe

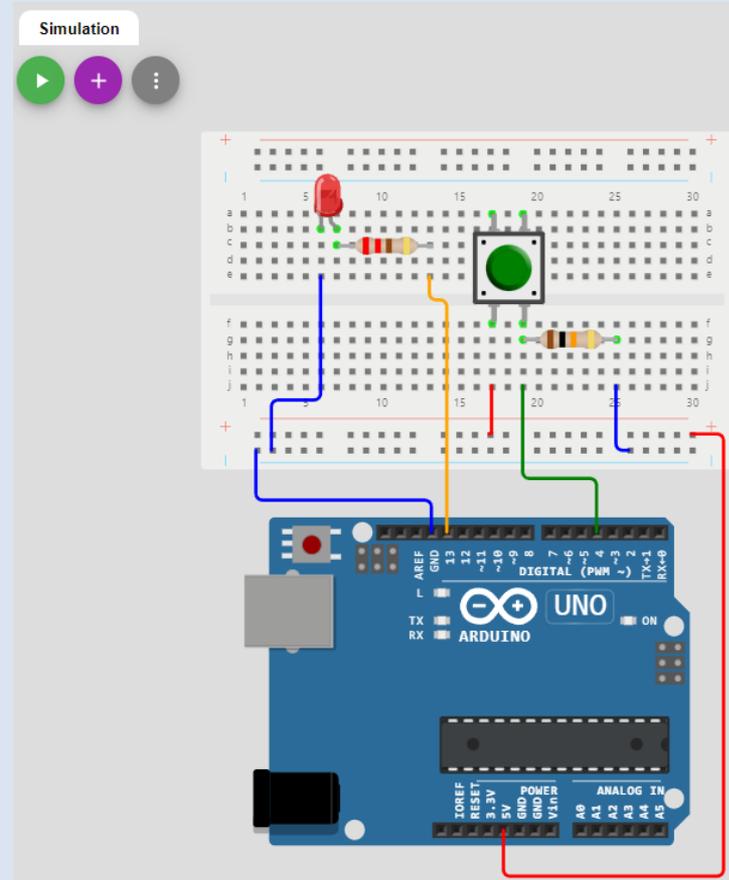
Taster gedrückt, dann LED an.
Taster gedrückt, dann LED aus.

Teile

Arduino
Taster
Widerstand 10 k Ω
Widerstand 220 Ω

diagram.json

```
{ "id": "r2",  
  "value": 10000 }  
  
{ "id": "r1",  
  "value": 220 }
```



Schaltung 4: Taster & LED, Sketch

Aufgabe

Taster LED an/aus

```
sketch.ino • diagram.json Library Manager ▾  
  
1  boolean led;  
2  void setup() {  
3      // put your setup code here, to run once:  
4      Serial.begin(9600);  
5      Serial.println("Taster & LED.");  
6      pinMode(13, OUTPUT);  
7      pinMode(4, INPUT);  
8      led = false;  
9  }  
10  
11 void loop() {  
12     // put your main code here, to run repeatedly:  
13     if (digitalRead(4) & led == false) {  
14         digitalWrite(13, HIGH);  
15         led = true;  
16         delay(1000);  
17     }  
18     if (digitalRead(4) & led == true){  
19         digitalWrite(13, LOW);  
20         led = false;  
21         delay(1000);  
22     }  
23 }  
..
```

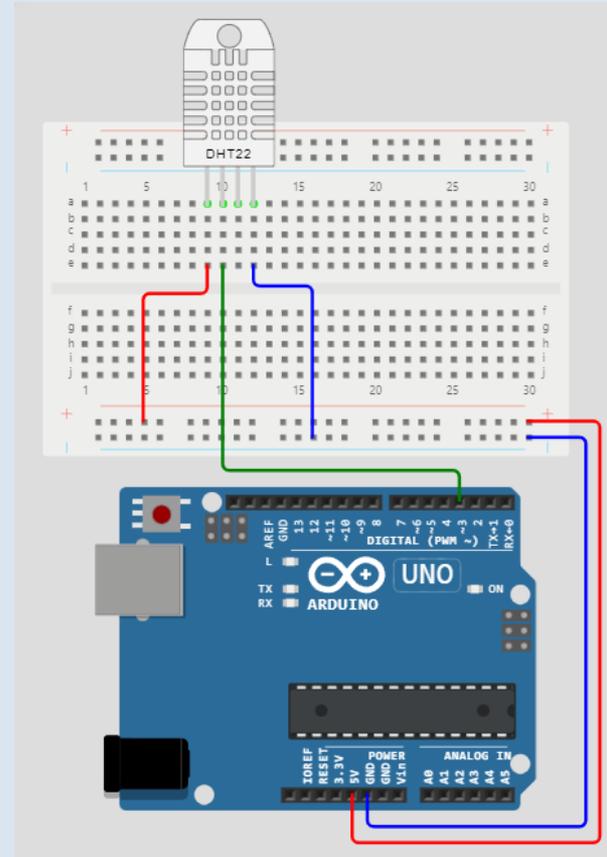
Schaltung 5: DHT22

Aufgabe

Feuchtigkeit & Temperatur messen.

Teile

Arduino
DHT22



Schaltung 5: DHT22, Sketch

Aufgabe

DHT22

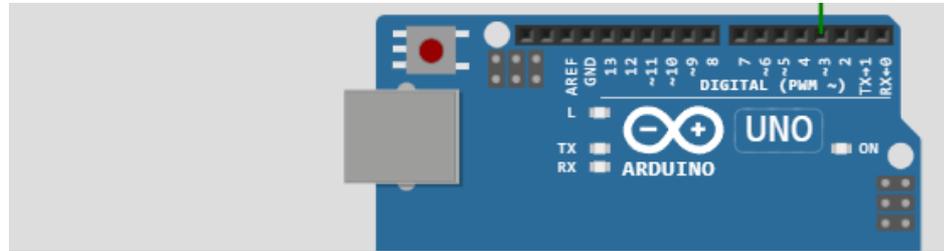
Temperatur & Feuchtigkeit
im Monitor anzeigen.

```
1  #include <DHT.h>
2
3  #define DHTPIN 3           // Digital pin connected to the DHT sensor
4  #define DHTTYPE DHT22    // DHT 22
5  DHT dht(DHTPIN, DHTTYPE); // Create object dht()
6
7  void setup() {
8      // put your setup code here, to run once:
9      Serial.begin(115200);
10     Serial.println("DHT22 Test");
11     dht.begin();
12 }
13
14 void loop() {
15     // put your main code here, to run repeatedly:
16     delay(2000);
17
18     // Reading humidity
19     float h = dht.readHumidity();
20     // Read temperature as Celsius
21     float t = dht.readTemperature();
22
23     Serial.print(F("Temperature: "));
24     Serial.print(t);
25     Serial.print(F(" °C Humidity: "));
26     Serial.print(h);
27     Serial.println(F(" %"));
28 }
```

Schaltung 5: DHT22, Monitor

Aufgabe

DHT22
Temperatur & Feuchtigkeit
im Monitor anzeigen.



DHT22 Test

```
Temperature: 25.50 °C Humidity: 54.50 %  
Temperature: 25.50 °C Humidity: 54.50 %  
Temperature: 25.50 °C Humidity: 54.50 %  
Temperature: 25.50 °C Humidity: 54.50 %
```

Werte über Schieberegler
einstellen



Schaltung 6: LCD

Aufgabe

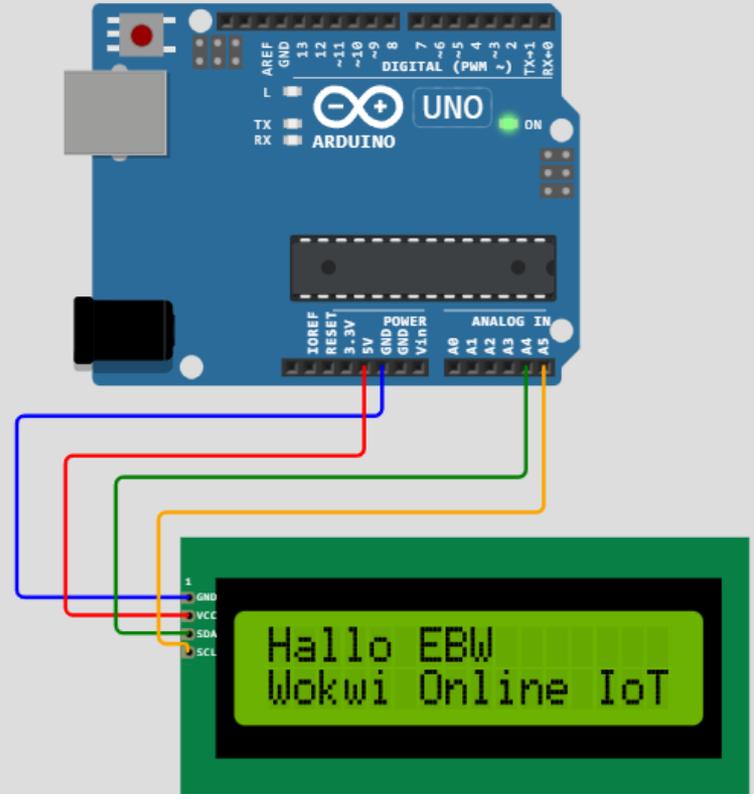
2zeiliges LCD simulieren

Teile

Arduino
LCD 16 x 2

I2C LCD1602	Arduino Uno
GND	GND
VCC	5V
SDA	A4
SCL	A5

EBW_06



Schaltung 6: LCD, Sketch

Aufgabe

2zeiliges LCD simulieren

```
1  #include <LiquidCrystal_I2C.h>
2
3  #define I2C_ADDR    0x27
4  #define LCD_COLUMNS 16
5  #define LCD_LINES   2
6
7  LiquidCrystal_I2C lcd(I2C_ADDR, LCD_COLUMNS, LCD_LINES);
8  void setup() {
9      // put your setup code here, to run once:
10     lcd.init();
11     lcd.backlight();
12
13     // Print something
14     lcd.setCursor(0, 0);
15     lcd.print("Hallo EBW");
16     lcd.setCursor(0, 1);
17     lcd.print("Wokwi Online IoT");
18 }
19
20 void loop() {
21     // put your main code here, to run repeatedly:
22
23 }
```

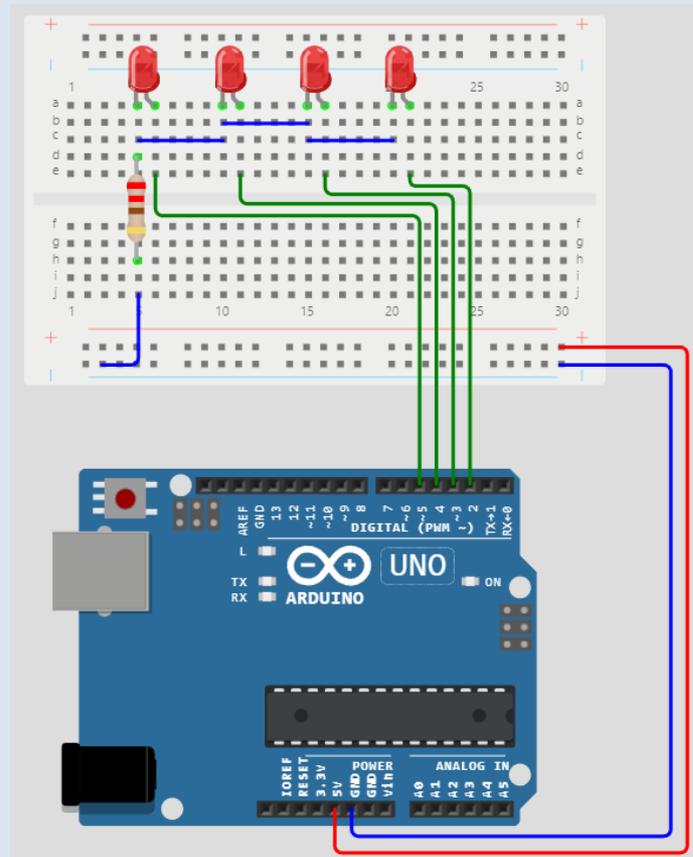
Schaltung 7: PORTs

Aufgabe

PORTs untersuchen

Teile

Arduino
Steckbrett
4 x LED
Widerstand 220 Ω



EBW_07

Schaltung 7: PORTs, Sketch

Aufgabe

PORTs untersuchen

```
1 // Anmerkung: Funktioniert auch wenn DDRD nicht gesetzt; PIND liefert falschen Wert
2 #define UsedPin PD3
3
4 void setup(){
5     Serial.begin(9600);
6     DDRD |= (1 << UsedPin);           // make PDx an output, changes only PDx
7     // DDRD |= 0b00000100;           // Binary
8     // DDRD |= 0x08;                 // Hex number
9     printBits("DDRD ", DDRD);       // Monitor information
10
11     PORTD = (1 << UsedPin);          // PDx on
12     printBits("PORTD ", PORTD);
13     printBits("PIND",PIND);
14 }
15
16 void loop(){
17     _delay_ms(1000);                 // AVR function
18     PORTD ^= (1 << UsedPin);         // toggle PDx
19     _delay_ms(1000);                 // AVR function
20 }
21
22 // a function used for debugging
23 void printBits(String _msg, byte b)
24 {
25     Serial.println(_msg);
26     Serial.println("76543210");
27     for (int i = 7; i >= 0; i--) Serial.print(bitRead(b, i));
28     Serial.println();
29     Serial.flush();
30 }
```

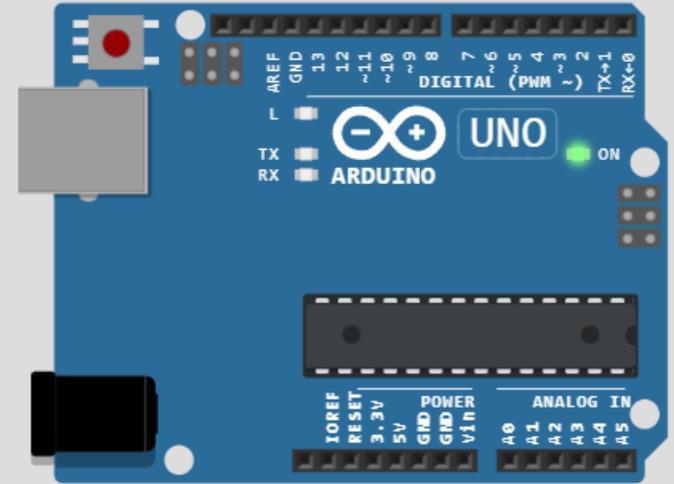
Aufgabe 1: Pointer

Aufgabe

Pointer untersuchen

Teile

Arduino



```
ptrMemAdr 8FB
*ptrMemAdr 10
*ptrMemAdr 20
result 20
MyDDRD 20
ptrMemAdr 8FB
&ptrMemAdr 8F9
```

EBW_pointer_01

Aufgabe 1: Pointer, Sketch

Aufgabe

Pointer untersuchen

```
1 // Pointer arithmetic
2 void setup() {
3   Serial.begin(9600);
4
5   uint8_t MyDDRD = 0x10; // declare a variable and initialize with 0x10
6   uint8_t *ptrMemAdr; // "*" => declare ptrMemAdr as a pointer to uint8_t data type
7   ptrMemAdr = &MyDDRD; // "&" => give me the address of MyDDRD
8
9   Serial.print("ptrMemAdr ");
10  Serial.println((long)ptrMemAdr,HEX); // => 8FB (contents of ptrMemAdr = address of MyDDRD)
11  Serial.print("*ptrMemAdr ");
12  Serial.println(*ptrMemAdr,HEX); // => 10 (contents of address pointed to by ptrMemAdr).
13
14  | | | | | | | | | | | | | | | | // the "*" sets or gets contents depending on syntax
15  *ptrMemAdr = 0x20; // "*" => set the contents of address pointed to by ptrMemAdr
16  uint8_t result = *ptrMemAdr; // "*" => get the value of the address of pointer ptrMemAdr
17
18  Serial.print("*ptrMemAdr ");
19  Serial.println(*ptrMemAdr,HEX); // => 20 (contents of address pointed to by ptrMemAdr).
20  Serial.print("result ");
21  Serial.println(result, HEX); // => 20 (contents of address pointed to by ptrMemAdr).
22  Serial.print("MyDDRD ");
23  Serial.println(MyDDRD,HEX); // => 20 (contents of MyDDRD)
24
25  Serial.print("ptrMemAdr ");
26  Serial.println((long)ptrMemAdr,HEX); // => 8FB (contents of ptrMemAdr = address of MyDDRD)
27  Serial.print("&ptrMemAdr ");
28  Serial.println((long)&ptrMemAdr,HEX); // => 8F9 (give me the address of ptrMemAdr)
29 }
30 void loop() {}
```

Vorteile, Nachteile

Schaltung	Versuchsaufbau im Browser.
Vorteile	<ul style="list-style-type: none">• Keine Hardware erforderlich.• Einfacher Aufbau der Schaltung.• Fehler führen nicht zu beschädigten Komponenten.
Nachteile	<ul style="list-style-type: none">• Nur rudimentäre Schaltungs-Simulation.• Schaltung kann unübersichtlich werden.• Sensoren werden simuliert (Maussteuerung).
Programm	Sketch im Browser-Editor.
	Der Editor bietet einfache Editiermöglichkeiten. Die „automatische Vervollständigung von Befehlen“ unterstützt bei der Eingabe von Befehlen. Lediglich der Anfangsbuchstabe eines Befehls muss bekannt sein.
Simulation	Das Zusammenspiel von Sketch und Versuchsaufbau wird gut visualisiert.

Erfahrungen

Arduino IDE Version	1.8.5	Simulation
Simulation	Widerstände können beliebigen Wert annehmen.	Ohne Auswirkung
	Pullup- / Pulldown-Widerstände sind notwendig	Mit Auswirkung
	LEDs können ohne Schutzwiderstand betrieben werden.	Ohne Auswirkung
	DDRx-Befehle werden nicht ausgewertet.	Ohne Auswirkung
Fazit:	Hardware-Simulation verbesserungswürdig.	
IDE	Es kann in Arduino-C++ (*.ino) als auch in C++ (*.cpp) programmiert werden.	

WOKWI im Internet I

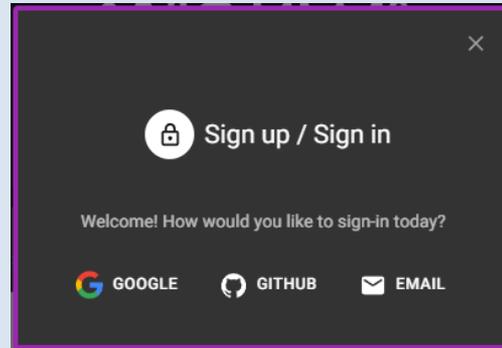
Link

<https://wokwi.com/>

Klicke auf
REGISTRIEREN/ANMELDEN



Wähle aus den 3
Registriermöglichkeiten eine
aus: z.B. GOOGLE



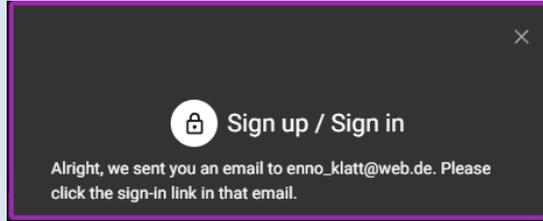
Folge den weiteren Anweisungen.

Achtung

Die Wahl der Registrierung entscheidet über den Zugang zu den in der Cloud gespeicherten Projekten. Eine spätere Änderung führt zum Verlust der Projekte.

WOKWI im Internet II

Registrierung per E-Mail



E-Mail

Sign in to Wokwi



Von: noreply@wokwi.com

29.11.2022 um 11:24 Uhr

Hello,

We received a request to sign in to Wokwi using this email address. If you want to sign in with your enno_klatt@web.de account, click this link:

[Sign in to Wokwi](#)

If you did not request this link, you can safely ignore this email.

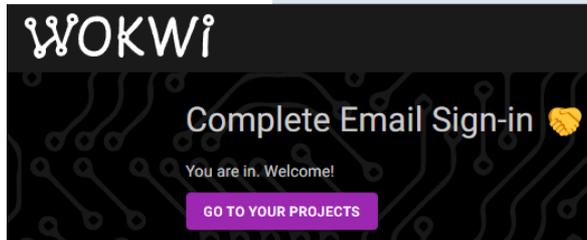
Thanks,

Your Wokwi team

Klicke auf den Link

[Sign in to Wokwi](#)

WOKWI-Startfenster



WOKWI im Internet III

WOKWI ohne Anmeldung!

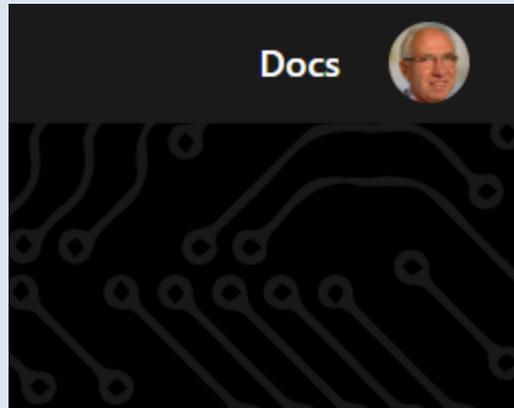
Auch das ist möglich!

Das WOKWI-Startfenster bietet eine Reihe von Projekten an, aus der ein Projekt durch Klicken gewählt werden kann.
Allerdings ist das Speichern hier wirkungslos.

WOKWI mit Anmeldung

Klicke auf REGISTRIEREN/ANMELDEN.
Klicke auf zuvor gewählte Registrierung.

Klicke auf das Hauptmenü
rechts neben „Docs“



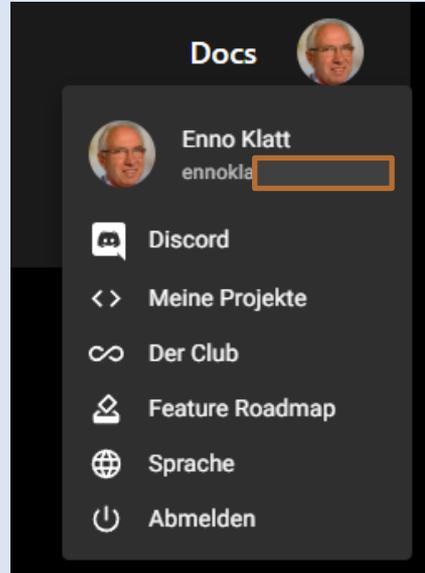
Statt des Fotos ist zu Beginn ein Personen-Symbol vorhanden.

WOKWI im Internet IV

Hauptmenü

Klicke auf

<> Meine Projekte



Klicke auf

+ NEW PROJECT

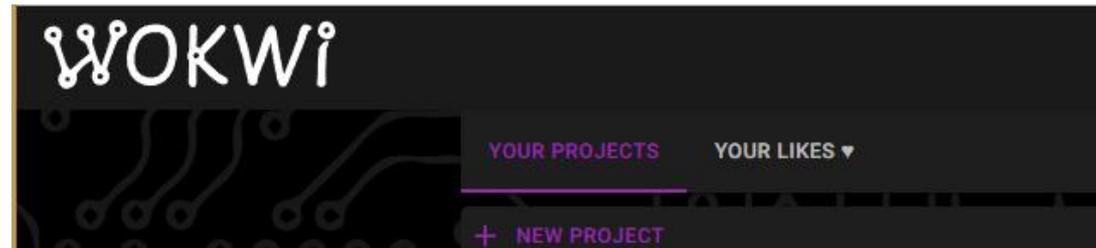


diagram.json I

JSON Die JavaScript Object Notation (JSON) ist ein kompaktes **Datenformat** in einer einfach lesbaren **Textform** für den Datenaustausch zwischen Anwendungen. JSON ist von Programmiersprachen unabhängig.

Link JSON https://www.w3schools.com/js/js_json_intro.asp

- JSON Syntax Regeln
- Daten liegen in "Name":Wert - Paaren vor.
 - **Daten** werden durch *Kommas* getrennt.
 - *Geschweifte* Klammern halten **Objekte**.
 - *Eckige* Klammern enthalten **Arrays**.

Objekt `{"type":"wokwi-lcd1602"}`

Datenliste (JSON string) `{"type":"wokwi-lcd1602", "top":80, "abc":null}`

Array

```
{ "employees": [
    { "firstName":"John", "lastName":"Doe" },
    { "firstName":"Anna", "lastName":"Smith" },
    { "firstName":"Peter", "lastName":"Jones" }
  ]
}
```

diagram.json II

Link

<https://docs.wokwi.com/diagram-format>

Die Diagrammdatei ist eine JSON-Datei mit mehreren Abschnitten.

Dateistruktur

```
{  
  "version": 1,  
  "author": "Enno Klatt",  
  "editor": "wokwi",  
  "parts": [],  
  "connections": []  
}
```

„parts“ Eigenschaft

```
"parts": [  
  { "id": "led1",  
    "type": "wokwi-led",  
    "left": 100,  
    "top": 50,  
    "attrs": { "color": "red" }  
  }  
]
```

Informationen zu „part“

Link https://docs.wokwi.com/?utm_source=wokwi#https://elements.wokwi.com/?path=/story/7-segment--red-4

Beispiel „wokwi-led“ <https://docs.wokwi.com/parts/wokwi-led>

Attribute	Name	Description	Default value
	color	The color of the LED body	"red"
	lightColor	The color of the light	depends on the color
	label	Text that appears below the led	
	gamma	Gamma correction factor	"2.8"
	flip	Flips the led horizontally	""

JSON

```
{  
  "type": "wokwi-led",  
  "id": "led1",  
  "top": -166.8,  
  "left": -53.8,  
  "attrs": { "color": "red" }  
}
```

Shortcuts Versuchsaufbau

	Key	Function
	-	Zoom out
	+	Zoom in
	F	Fit diagram to window (auto zoom)
	D	Duplicate (copy) the selected part
	R	Rotate the selected part
	Delete	Delete the selected part / wire
	?	Open documentation for the selected part
	Escape	Cancel wire (in wiring mode)
	G	Toggle the grid
	Shift	Toggle coarse grid snapping while dragging
	Alt	Toggle fine grid snapping while dragging
	Ctrl	Toggle fine grid snapping while dragging
	Ctrl+Z	Undo *
	Ctrl+Y	Redo *

Shortcuts allgemein

	Description	Windows / Linux	Mac
	Start simulation	Ctrl + Enter	⌘ Enter
	Save project	Ctrl + S	⌘ S
	Auto format code	Alt + Shift + F	⌘ ↑ F
	Trigger auto complete	Ctrl + Space	⌘ Space
	Show list of available commands	F1	F1
	Jump to next error in file	F8	F8
	Jump to previous error in file	Shift + F8	↑ F8

Sonstiges

Arduino PROJECT HUB	https://create.arduino.cc/projecthub/Hack-star-Arduino/projects https://create.arduino.cc/projecthub/search?q=wokwi
Welcome to Wokwi!	https://docs.wokwi.com/
Vorhandene Libraries	https://github.com/wokwi/wokwi-library-index/blob/gh-pages/docs/arduino.json
GitHub	https://github.com/wokwi
instructurables	https://www.instructables.com/Web-Based-Arduino-Simulator-From-Wokwi/ https://www.instructables.com/Online-Arduino-Simulator-From-Wokwi-for-Teachers-S/ https://www.instructables.com/Best-Free-Online-Wokwi-Arduino-Simulator-Why-Is-Th/ https://www.instructables.com/How-to-Simulate-Arduino-to-Blink-an-LED/ https://www.instructables.com/How-to-Runtest-Your-Arduino-Code-Online-for-Free/ https://www.instructables.com/LED-Chaser-With-7-Patterns-wokwi-Arduino/
Elektronik-Labor	https://www.elektronik-labor.de/Lernpakete/TPS/TPS46.html
Hackster	https://www.hackster.io/Hack-star-Arduino/push-buttons-and-arduino-a-simple-guide-wokwi-simulator-c2281f#toc-example-1---pushbutton-connected-to-an-arduino--pin-read-method-1
MAKER PRO	https://maker.pro/education/category/automation

WOKWI Projekte

EBW_01.ino	Schaltung 1: LED ein
EBW_02.ino	Schaltung 2: Blinkende LED
EBW_03.ino	Schaltung 3: Taster & Monitor
EBW_04.ino	Schaltung 4: Taster & LED
EBW_05.ino	Schaltung 5: DHT22
EBW_06.ino	Schaltung 6: LCD
EBW_07.ino	Schaltung 7: PORTs
EBW_pointer_01.ino	Pointer allgemein
EBW_pointer_02.ino (*)	PORT direct access
EBW_pointer_03.ino (*)	PORT-Aufbau, pointer

(*) nicht im Handout enthalten