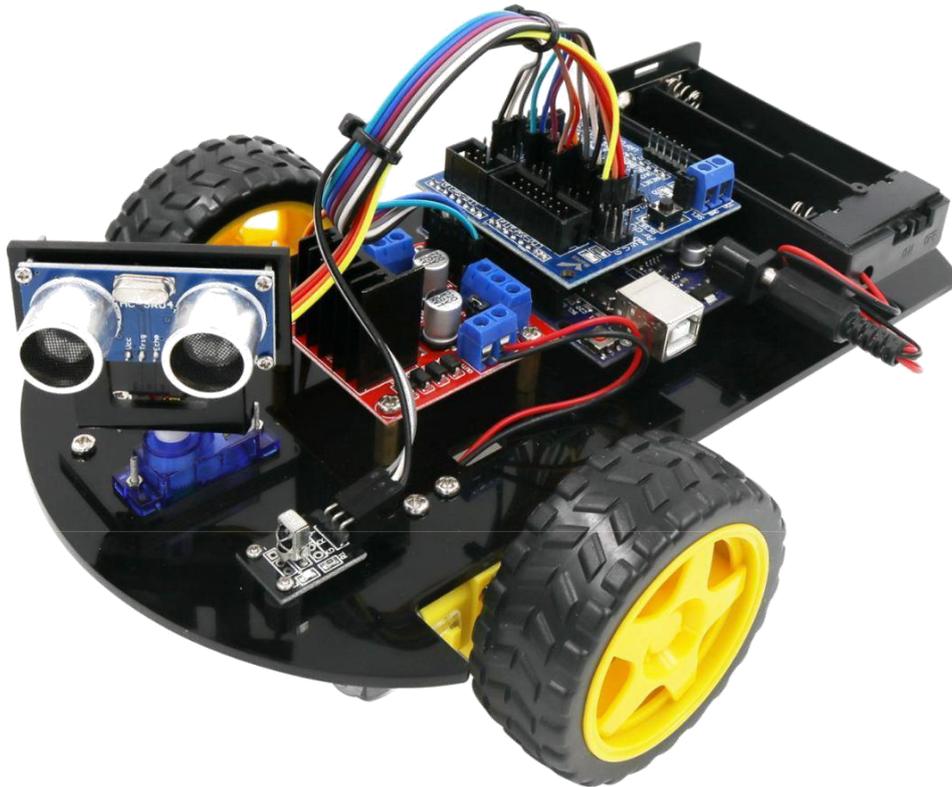


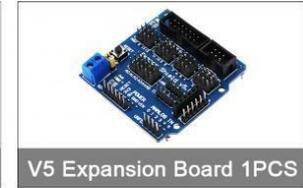
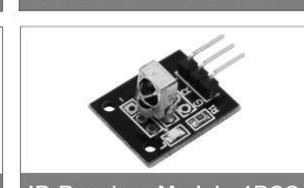
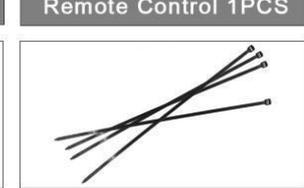
Anleitung yourDroid 2WD Smart Car Roboter



Inhaltsverzeichnis

1. Aufbau	2
2. Arduino IDE installieren	14
3. Bibliotheken hinzufügen und den seriellen Monitor öffnen	20
4. Blink Beispiel	26
5. Servo	31
6. Ultraschallsensor	32
7. Infrarot-Fernbedienung	36
8. L298N Motortreiber	40
9. Auto mit Infrarotfernbedienung steuern	44
10. Auto mit Hinderniserkennung	47

Packliste

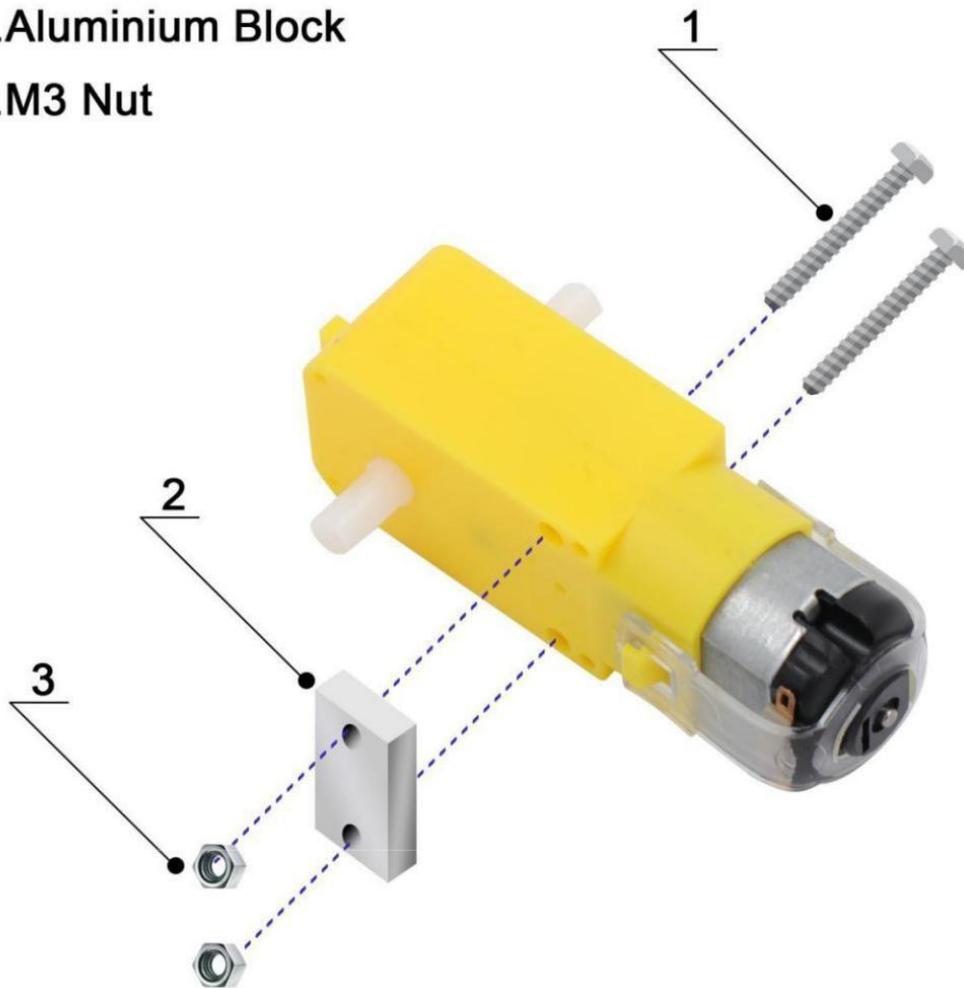
 <p>UNO R3 with Cable 1PCS</p>	 <p>V5 Expansion Board 1PCS</p>	 <p>Universal Wheel 1PCS</p>	 <p>Ultrasonic Sensor 1PCS</p>
 <p>Acrylic Chassis 1PCS</p>	 <p>Servo Motor(SG90) 1PCS</p>	 <p>L298N Motor Driver Board 1PCS</p>	 <p>F-F Dupont Wire 20PIN</p>
 <p>Cell Box 1PCS</p>	 <p>Tires 2PCS DC Motor 2PCS</p>	 <p>IR Receiver Module 1PCS</p>	 <p>Remote Control 1PCS</p>
 <p>Ultrasonic Holder 1PCS</p>	 <p>Screw Kit 1 Set</p>	 <p>Screwdriver 1PCS</p>	 <p>Bunding Belt 2PCS</p>

1. Aufbau

1.M3*30mm

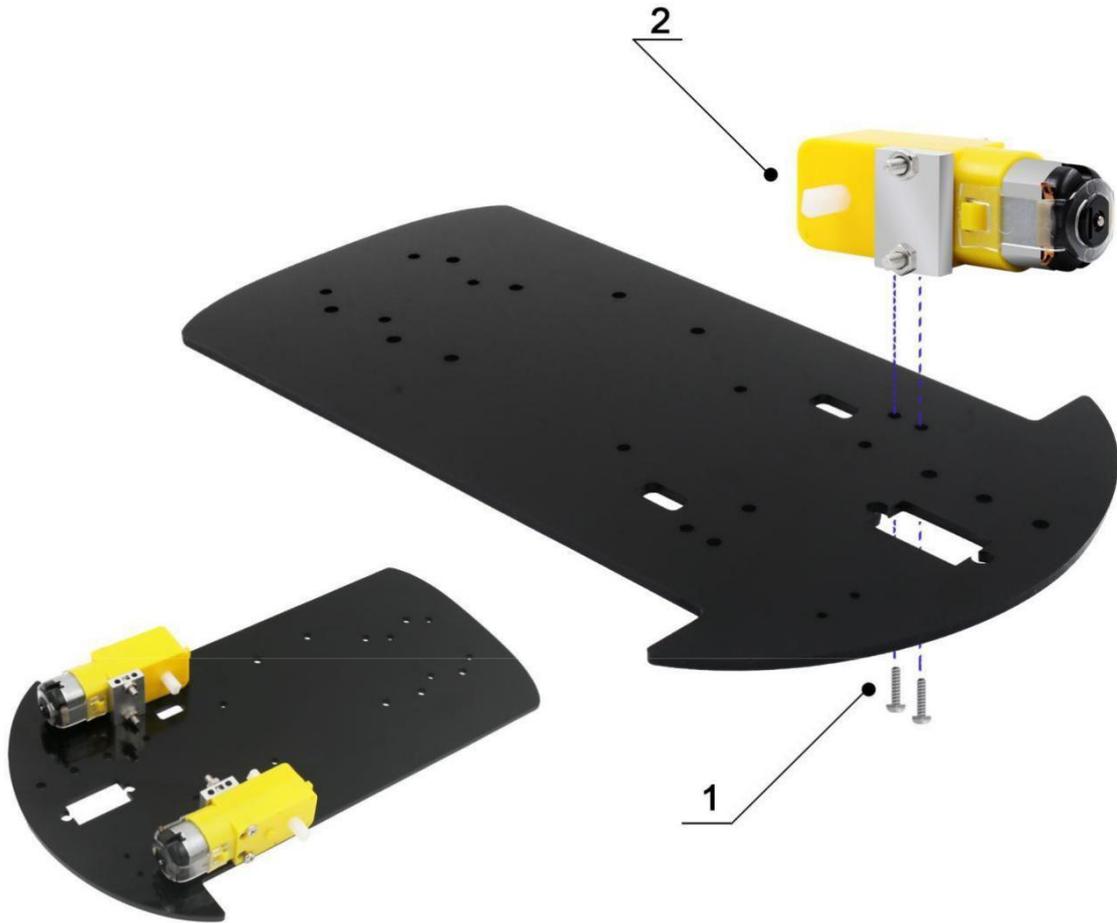
2.Aluminium Block

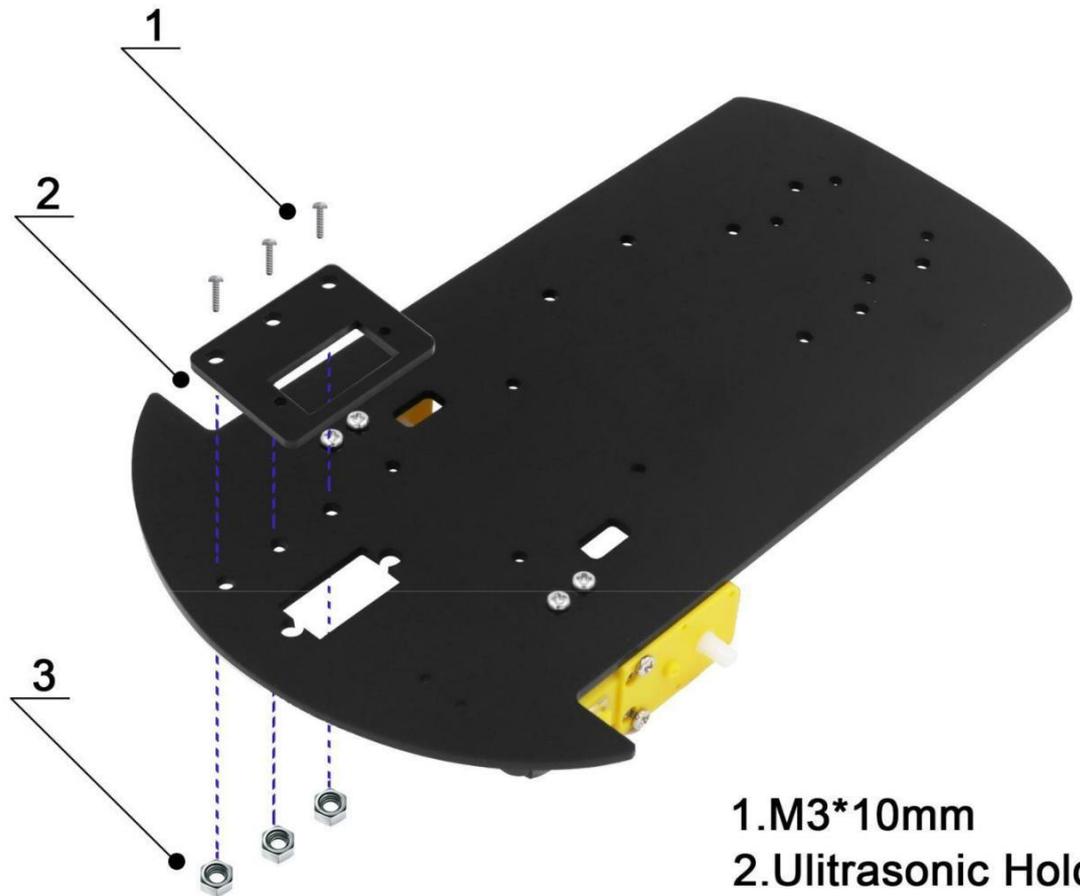
3.M3 Nut



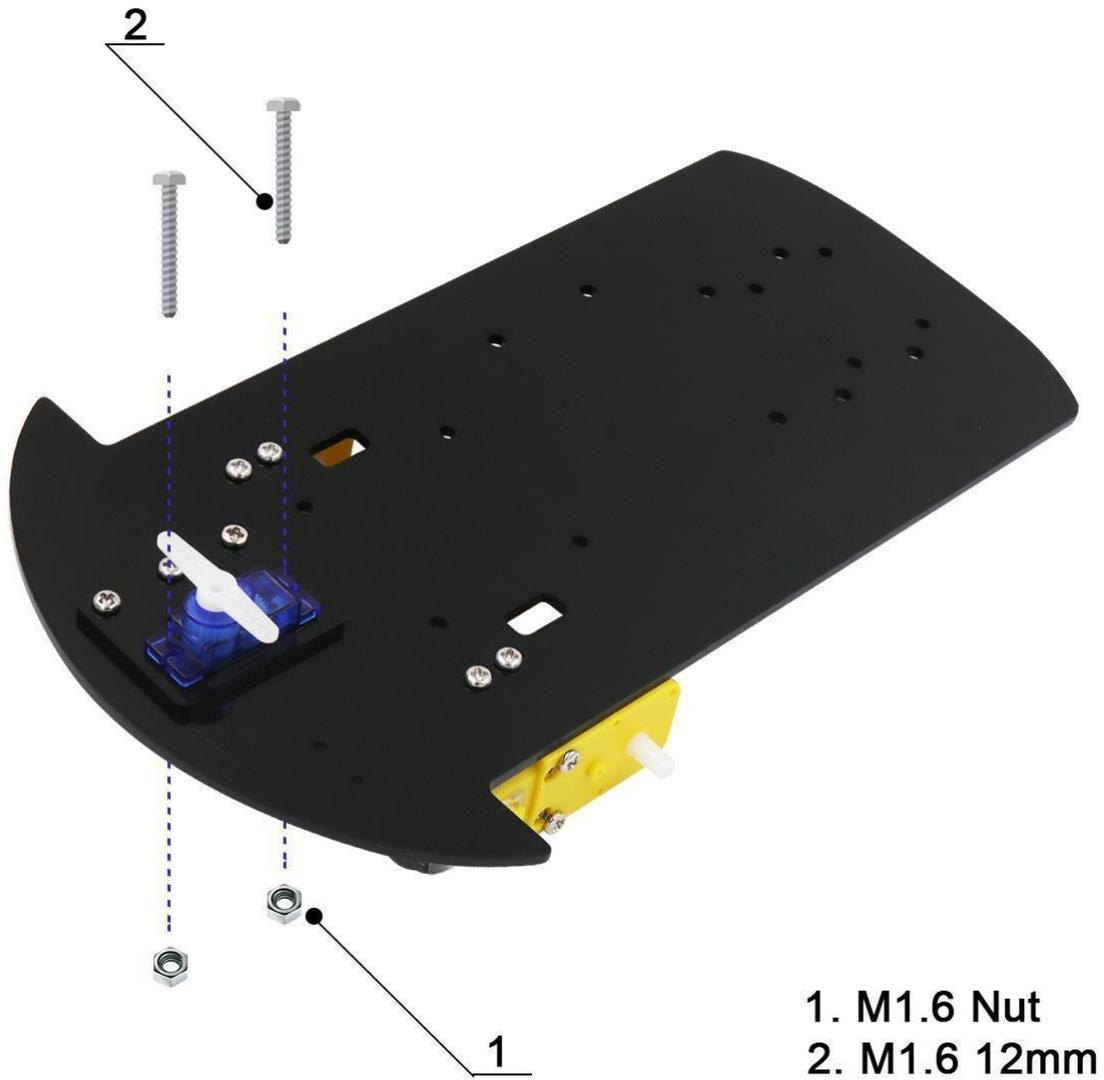
1.M3*6MM

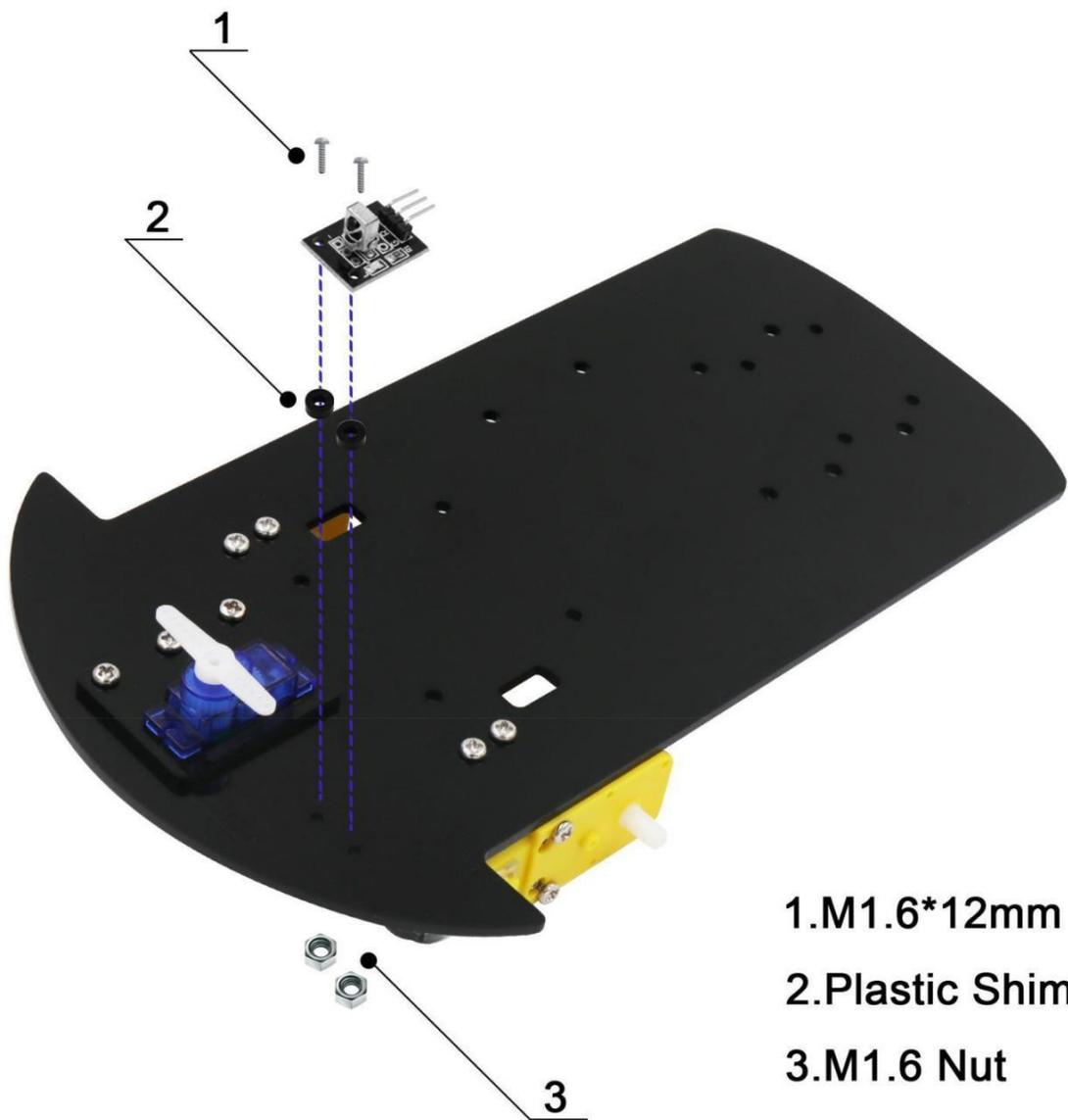
2.Motor with Aluminium Block

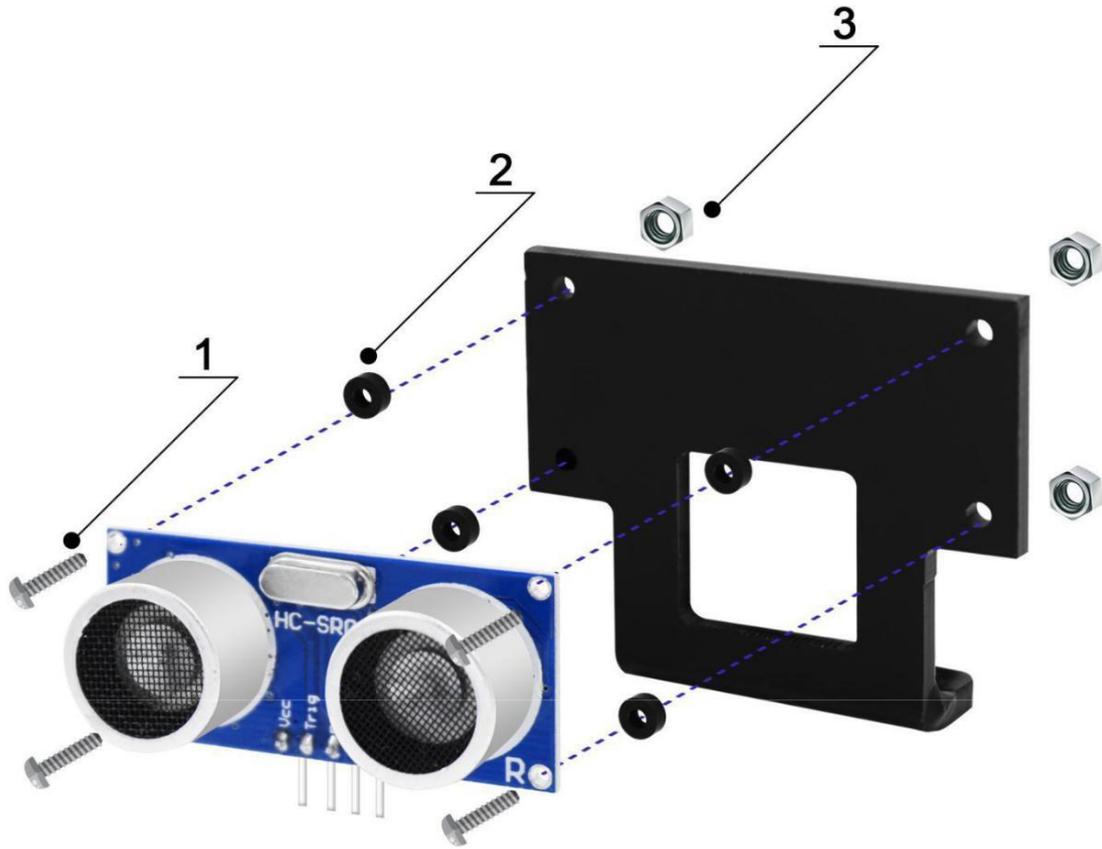




- 1.M3*10mm
- 2.Ultrasonic Holder
- 3.M3 Nut

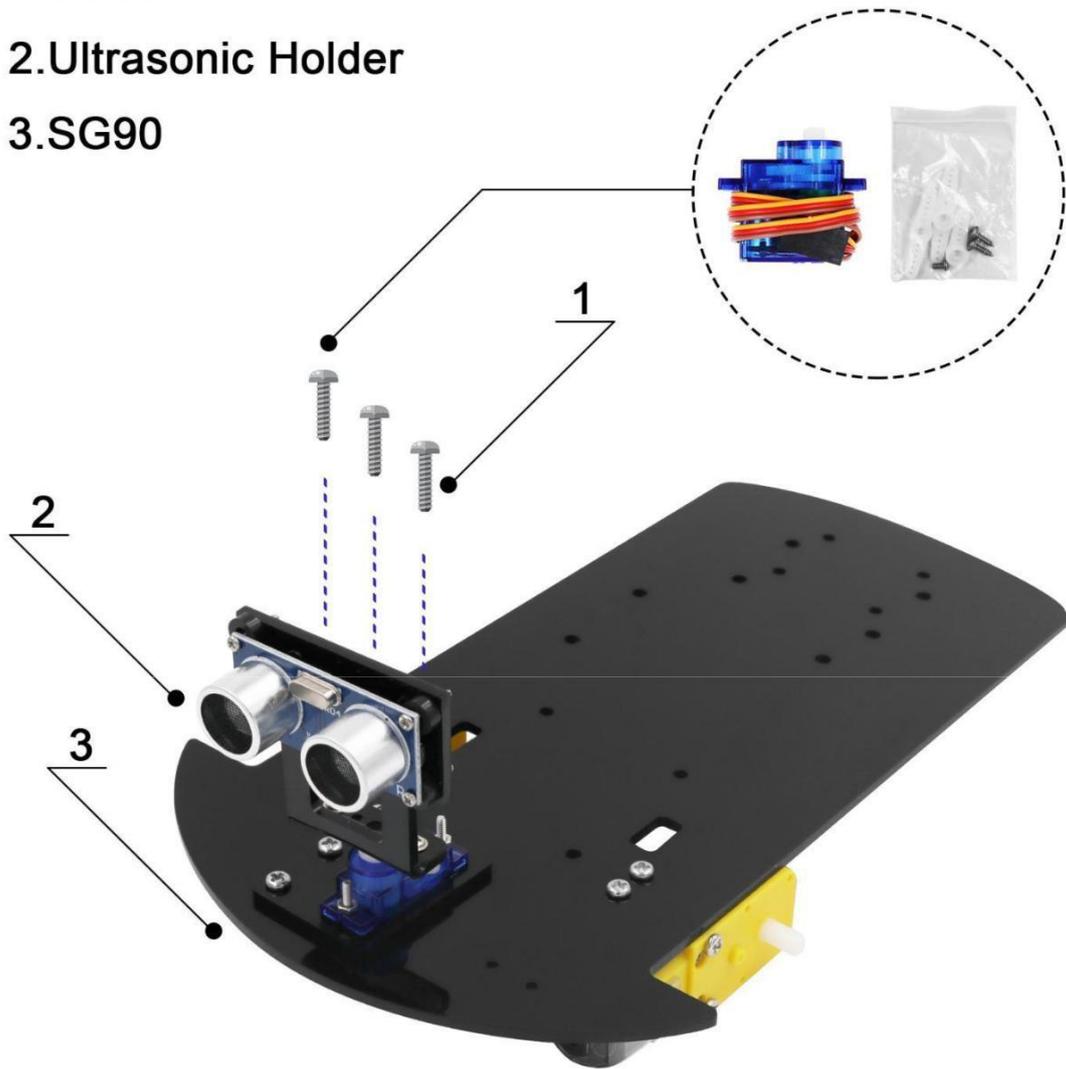






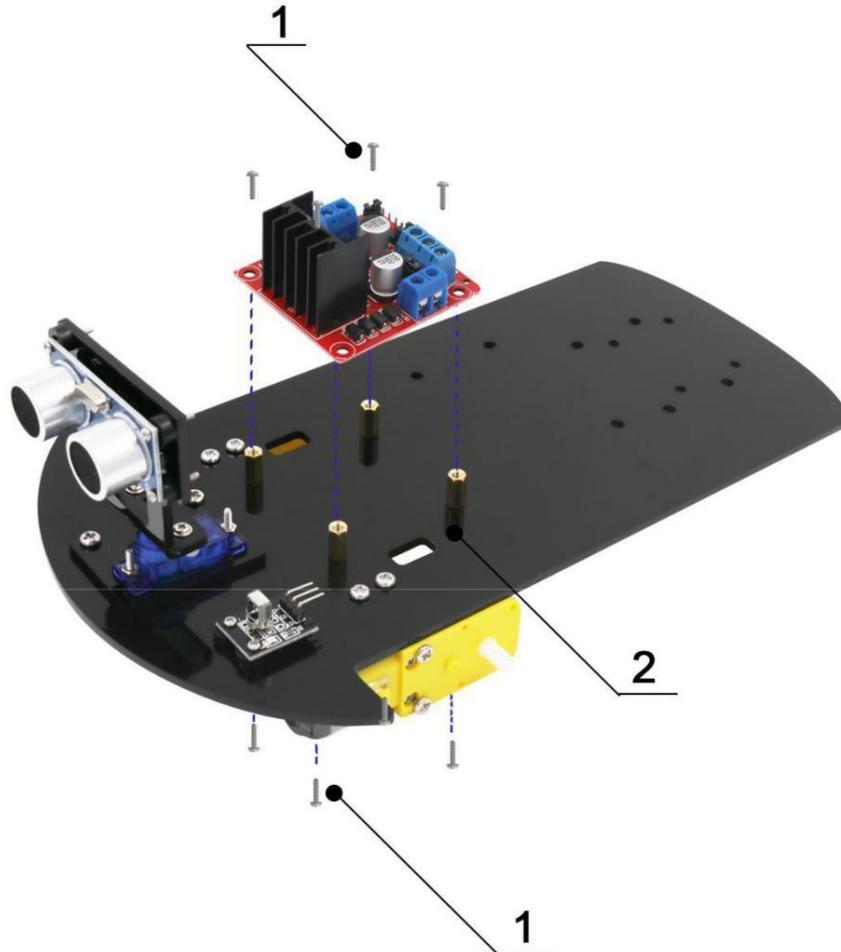
- 1.M1.6*12mm
- 2.Plastic Shim
- 3.M1.6 Nut

- 1.SG90 Screws
- 2.Ultrasonic Holder
- 3.SG90



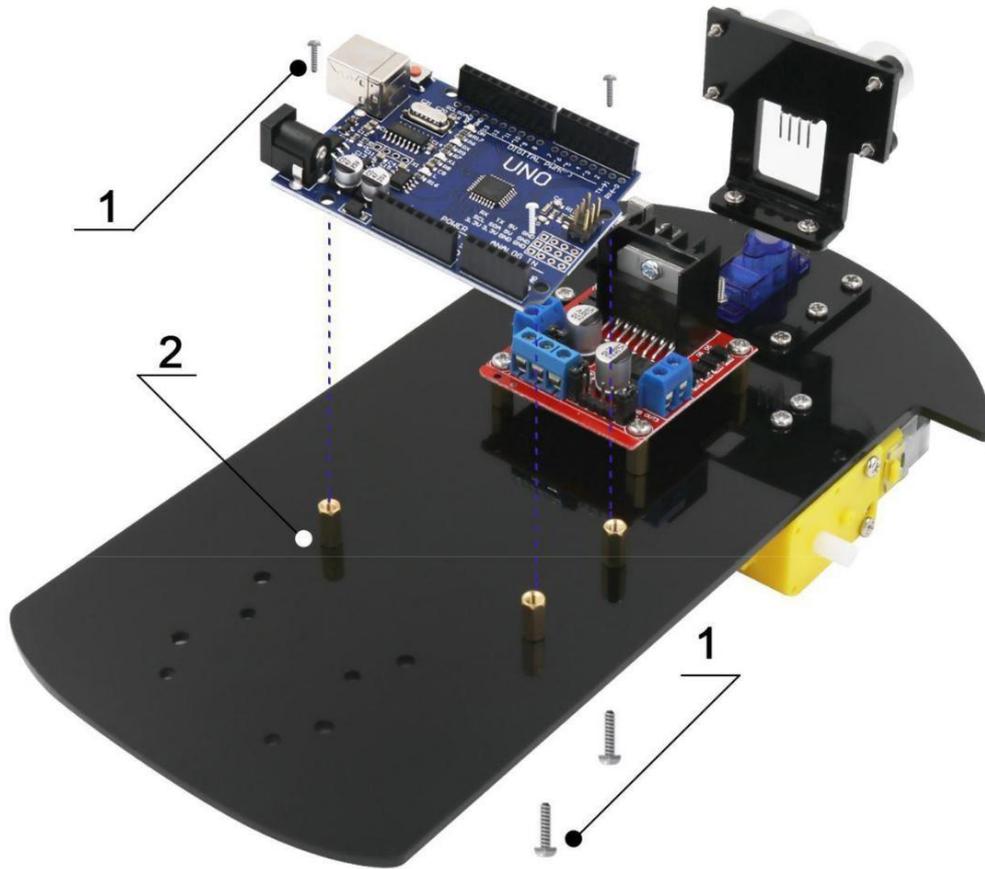
1.M3*6mm

2.Copper Cylinder M3*8mm

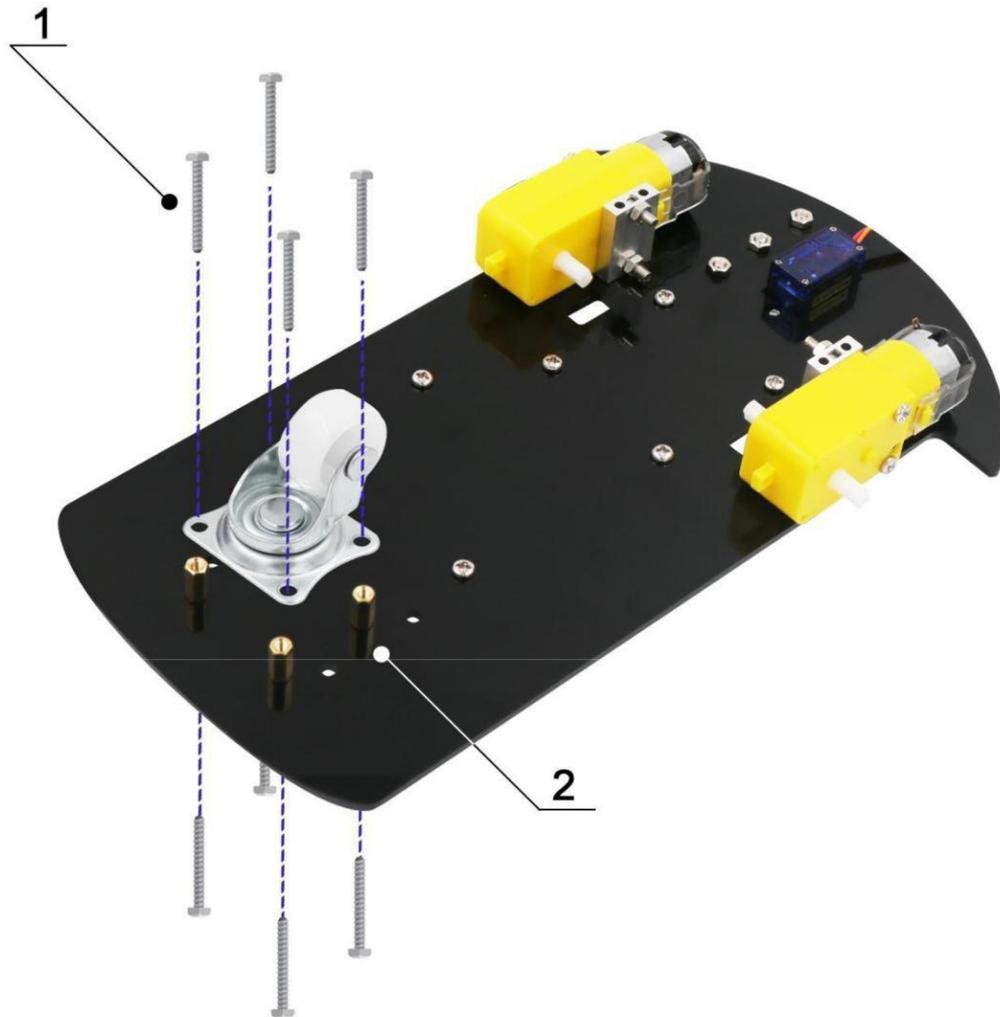


1.M3*6mm

2.Copper Cylinder M3*8mm

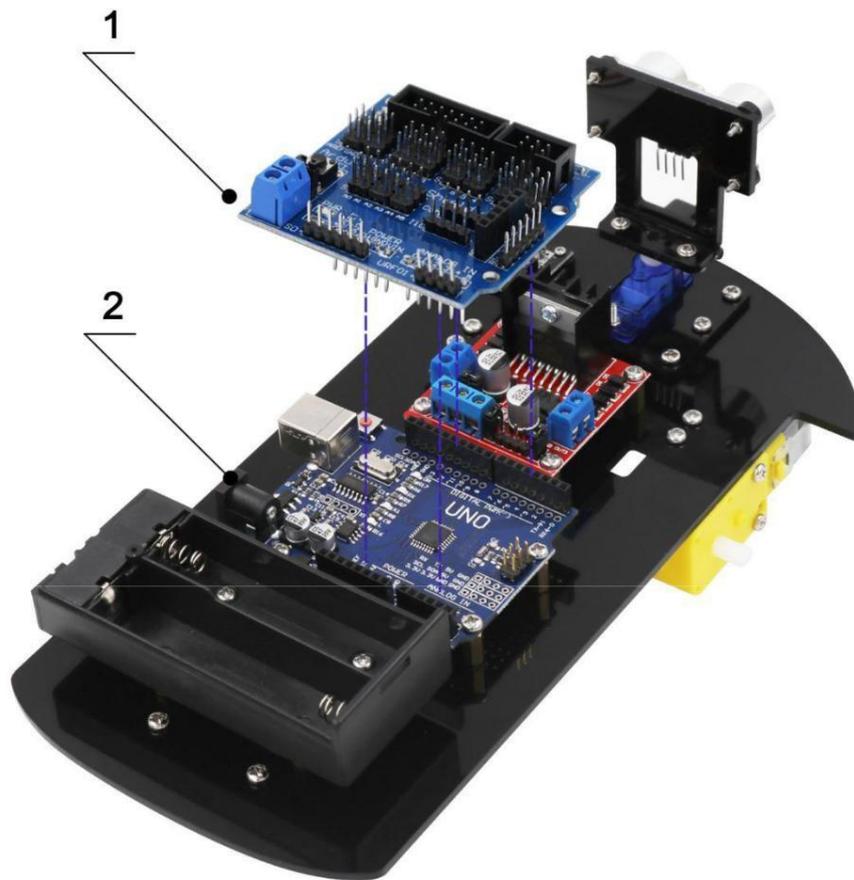


- 1.M2*6mm
- 2.Copper Cylinder M3*8mm



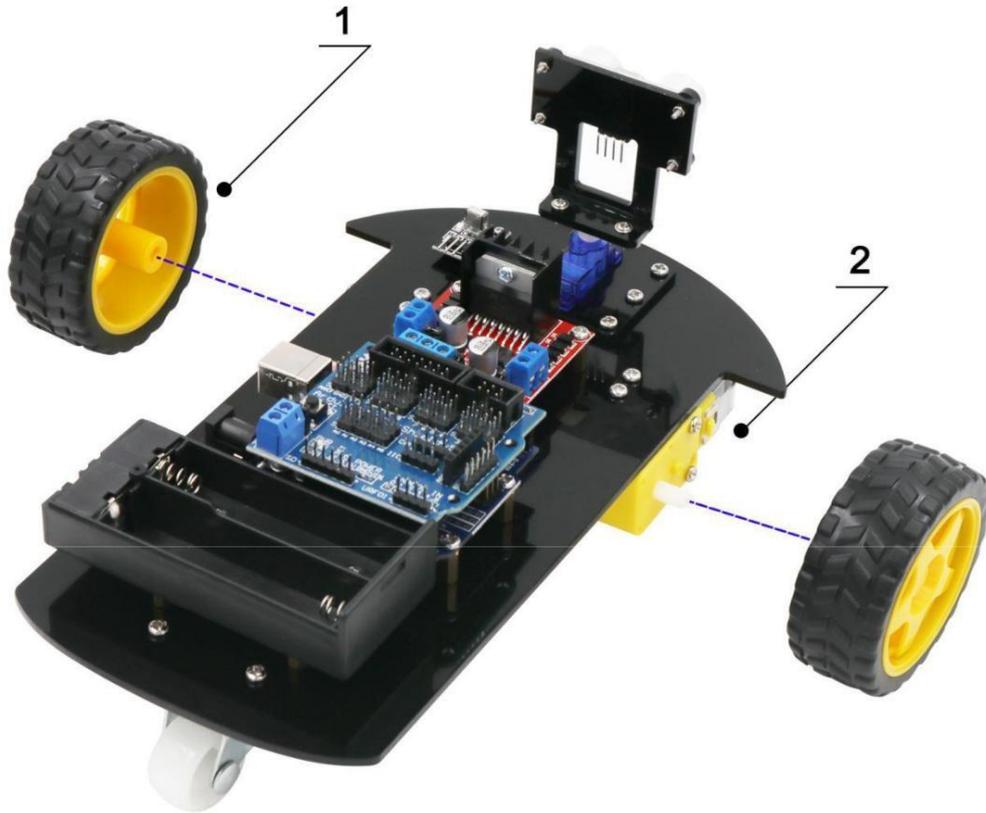
1.V5 Expansion Board

2.UNO R3



1. Tire

2. DC Motor



2. Arduino IDE installieren

Die Arduino Integrated Development Environment ist die Entwicklungsumgebung und Software der Arduino Plattform.

In dieser Lektion lernen Sie Ihren Computer einzurichten, um Arduino Boards nutzen zu können.

Die Arduino Software erlaubt es Ihnen Arduinos und andere Entwicklungsplatinen zu programmieren. Die Software ist verfügbar für Windows, Mac und Linux. Der Installationsprozess ist für alle Plattformen unterschiedlich.

Arduino IDE herunterladen

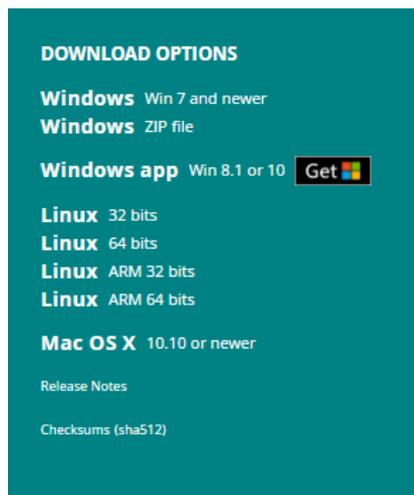
Schritt 1: <https://www.arduino.cc/en/software> aufrufen



The screenshot shows the Arduino IDE 1.8.19 download page. On the left, there is a description of the software and a link to the 'Getting Started' page. On the right, there is a 'DOWNLOAD OPTIONS' section with links for Windows (Win 7 and newer), Windows app (Win 8.1 or 10), Linux (32 bits, 64 bits, ARM 32 bits, ARM 64 bits), and Mac OS X (10.10 or newer). There are also links for Release Notes and Checksums (sha512).

Auf dieser Webseite finden Sie immer die aktuelle Version. Die aktuelle Version ist vermutlich neuer als die aus dieser Anleitung.

Schritt 2: Passenden Installer herunterladen



The screenshot shows the 'DOWNLOAD OPTIONS' section of the Arduino IDE download page. It lists the following options:

- Windows** Win 7 and newer
- Windows** ZIP file
- Windows app** Win 8.1 or 10 
- Linux** 32 bits
- Linux** 64 bits
- Linux** ARM 32 bits
- Linux** ARM 64 bits
- Mac OS X** 10.10 or newer

There are also links for Release Notes and Checksums (sha512).

„Win 7 and newer“ auswählen, falls Sie nicht die App aus dem Microsoft Store installieren möchten.

Schritt 3: Download Starten

Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **58.947.663** times — impressive! Help its development with a donation.

JUST DOWNLOAD

CONTRIBUTE & DOWNLOAD

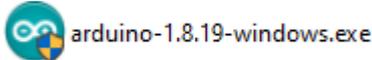


Learn more about [donating to Arduino](#).

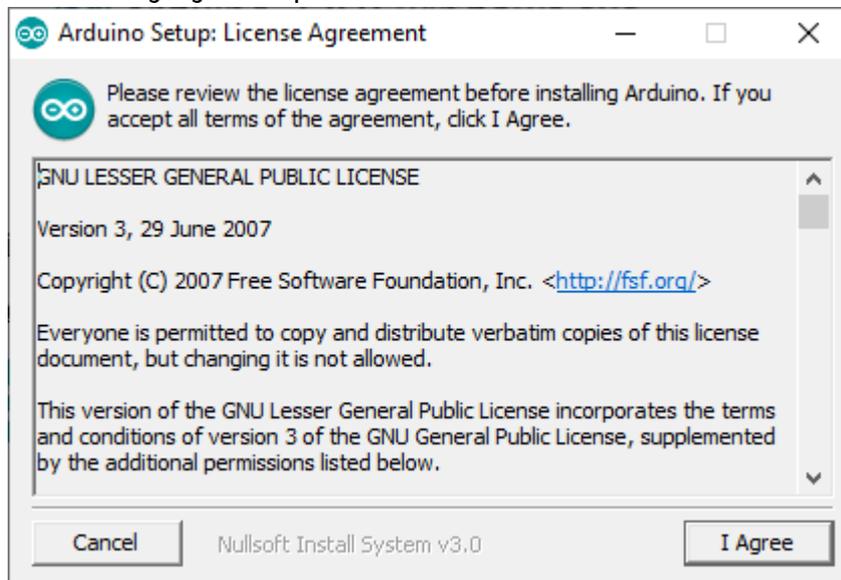
„Just Download“ auswählen.

Installation unter Windows

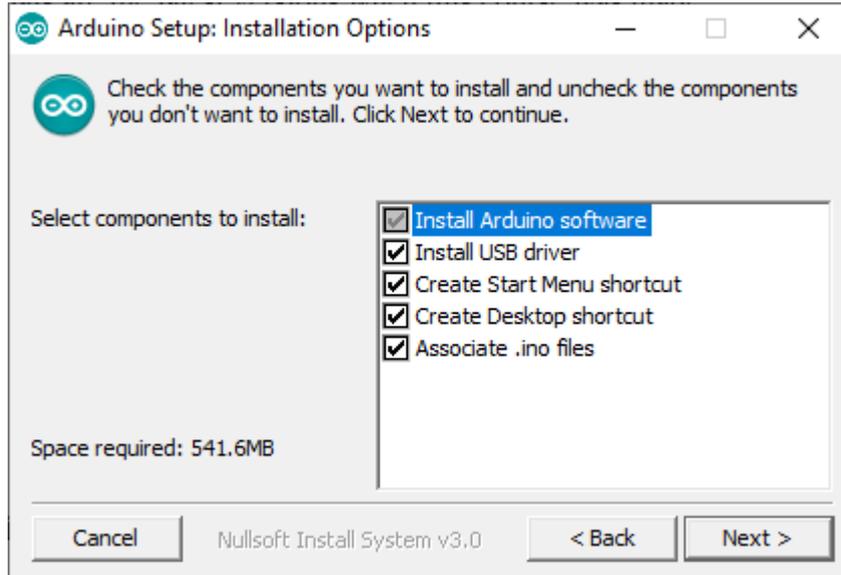
1. Installationsdatei starten



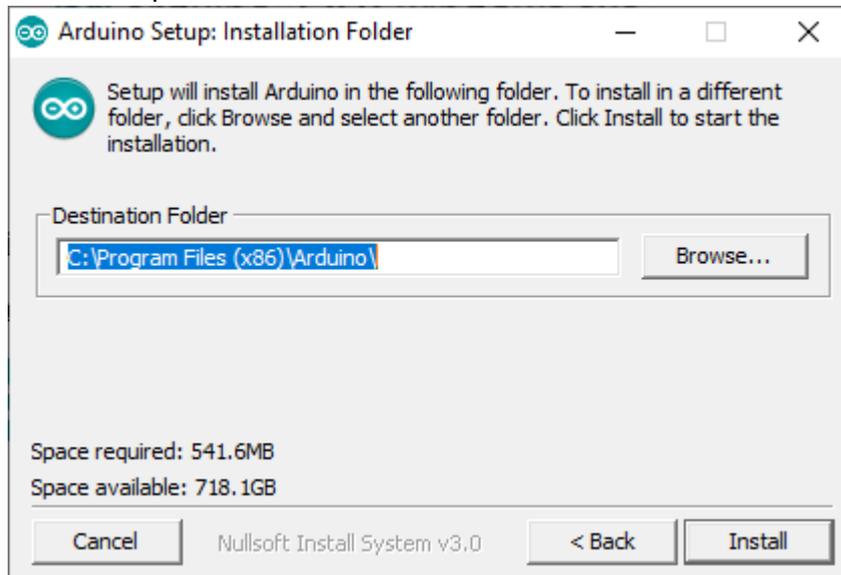
2. Lizenzbedingungen akzeptieren



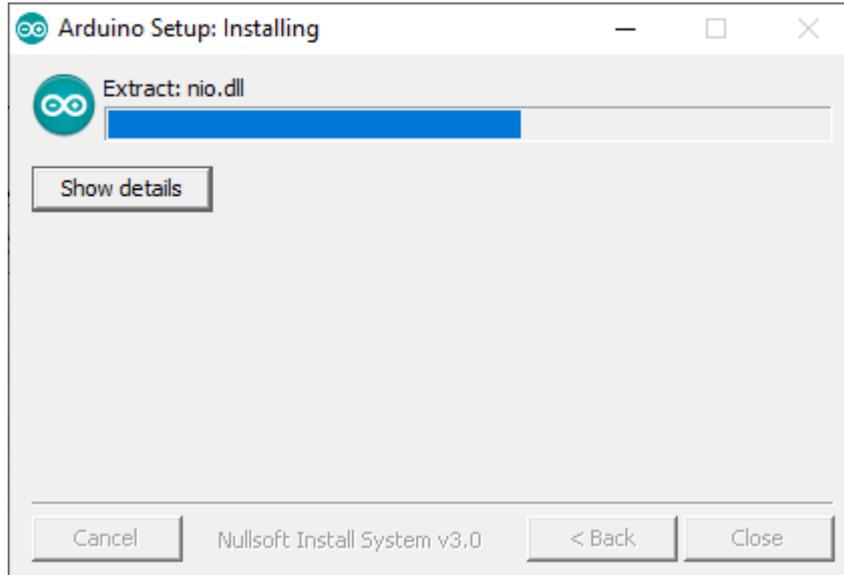
3. Auf „Next“ klicken



4. Installationspfad auswählen oder „Install“ drücken



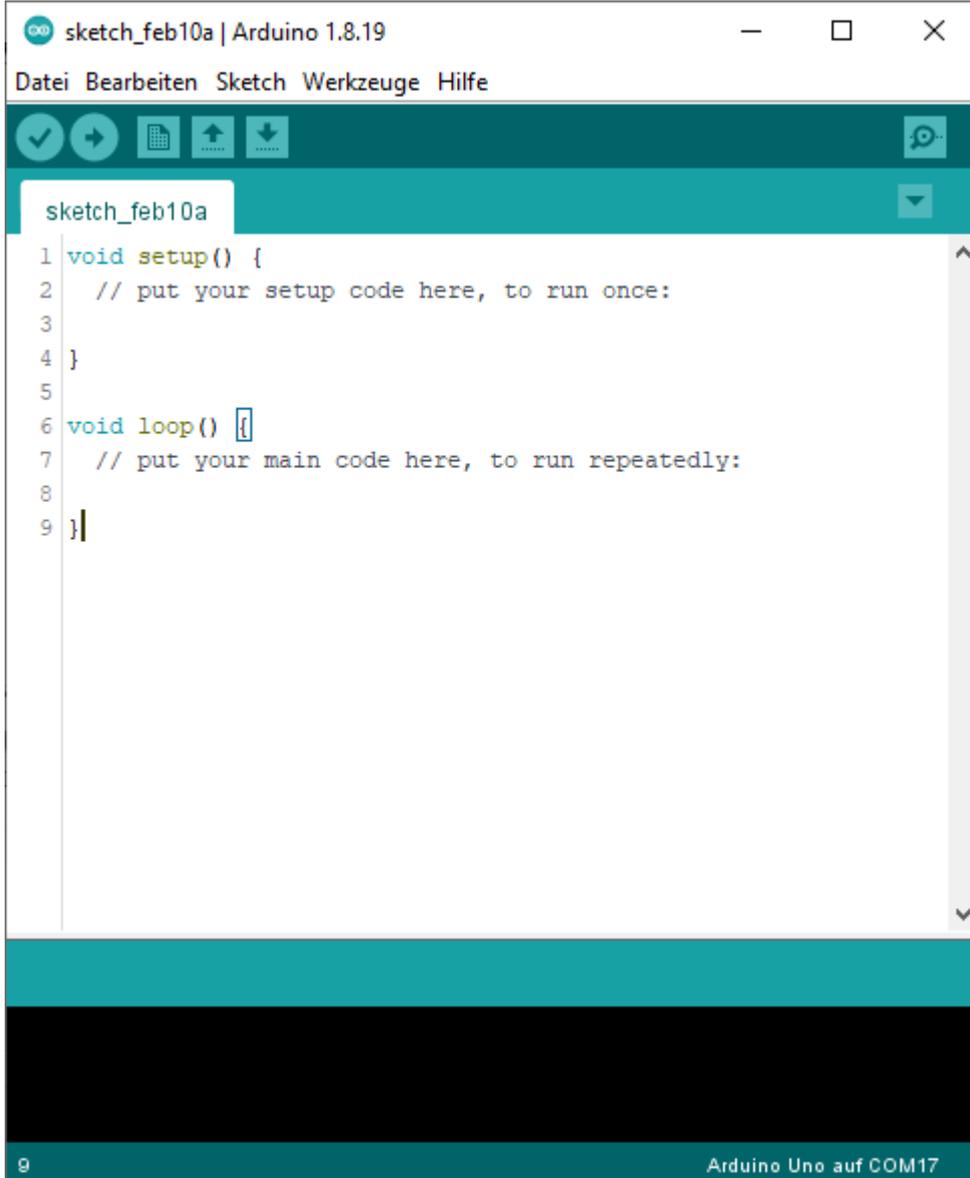
5. Installation abwarten



6. Arduino USB-Treiber installieren lassen



7. Zurück zum Desktop und die Verknüpfung der Arduino IDE doppelklicken



```
sketch_feb10a
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

Arduino Uno auf COM17

8. Das war die Installation unter Windows.

Installation unter Linux

1. Package extrahieren
2. Install Script starten
Entweder im entpackten Archiv die „install.sh“ starten oder im Terminal zum Dateipfad navigieren und „./install.sh“ eingeben
3. Die Arduino IDE kann nun gestartet werden. Auf dem Desktop finden Sie die Verknüpfung.

Empfohlen: Wenn Sie ein Ubuntu System nutzen, können Sie stattdessen die Installation über das Software-Center von Ubuntu durchführen.

Arduino Installation unter MacOS

1. Die Installationsdatei ist im Zip-Format. Wenn Sie Safari nutzen, wird der Inhalt automatisch entpackt. Die Installationsdatei doppelklicken, falls noch nicht bereits installiert, werden Sie aufgefordert die Java Runtime Library zu installieren.
2. Die Arduino IDE kann nun gestartet werden.

3. Bibliotheken hinzufügen und den seriellen Monitor öffnen

Installieren zusätzlicher Arduino Bibliotheken

Sobald Sie sich mit der Arduino IDE vertraut gemacht haben und die integrierten Beispiele ausprobiert haben, können Sie mit Bibliotheken mehr Potenzial aus Ihrem Arduino Board schöpfen.

Was sind Bibliotheken?

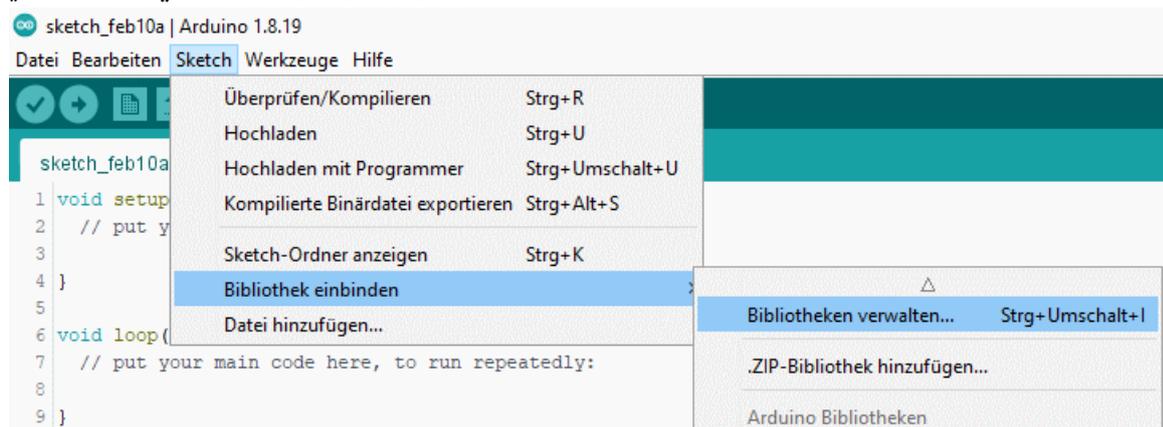
Bibliotheken sind Codesammlung um Sensoren, Module, Displays oder Funktionen wesentlich einfacher programmieren zu können. Zum Beispiel vereinfacht die integrierte LiquidCrystal.h Bibliothek die Kommunikation zu LCD Displays, um Zeichenfolgen unkompliziert ausgeben können. Es gibt tausende zusätzliche Bibliotheken im Internet kostenfrei zum Herunterladen. Die integrierten Bibliotheken besitzen Beispiele, die Sie unter „Datei – Beispiele“ aufrufen können. Zusätzliche Bibliotheken müssen heruntergeladen und installiert werden.

Wie installiere ich weitere Bibliotheken?

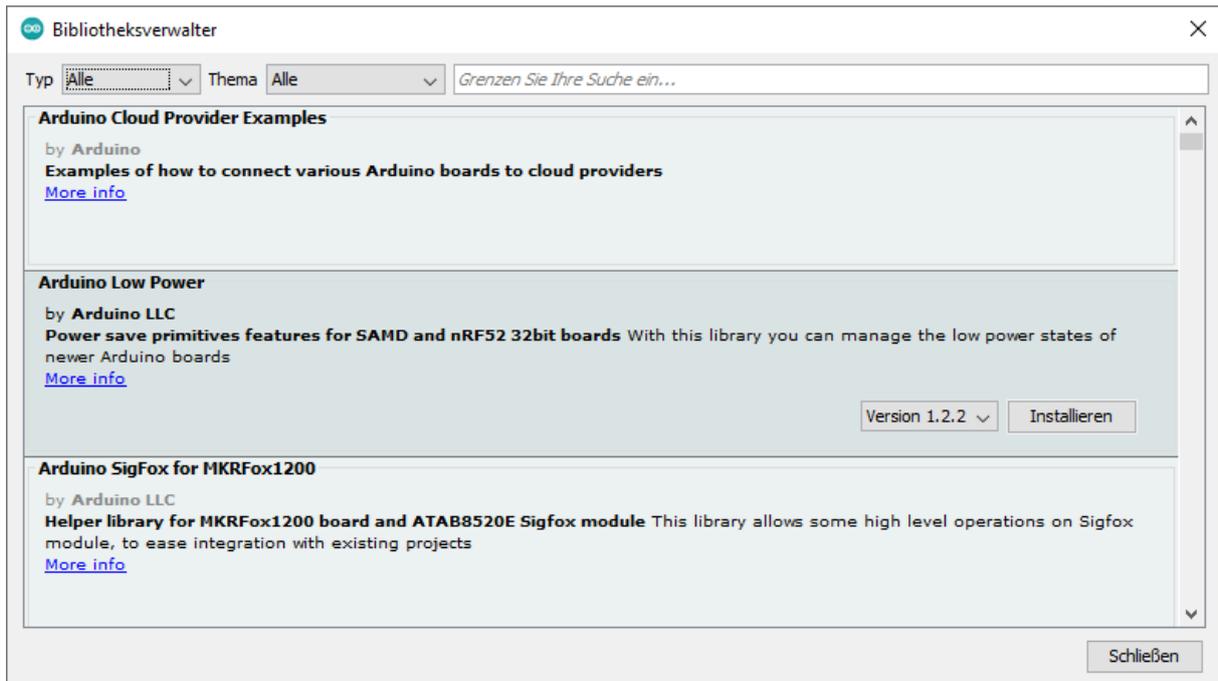
Für die Installation von weiteren Bibliotheken gehen wir hier auf die verschiedenen Möglichkeiten ein.

Bibliotheken mit der Arduino IDE installieren

Seit Version 1.8.0 gibt es den praktischen Library Manager mit dem Sie im Handumdrehen Bibliotheken suchen, herunterladen und installieren können. Öffnen Sie die Arduino IDE und klicken Sie im im Reiter „Sketch“ auf **„Bibliotheken verwalten...“**



Es öffnet sich der Library Manager, mit dem Sie Bibliotheken installieren, aktualisieren oder entfernen können. Gelegentlich werden Sie in der Arduino IDE unten rechts eine Meldung erhalten, dass für installierte Bibliotheken Aktualisierungen verfügbar sind.

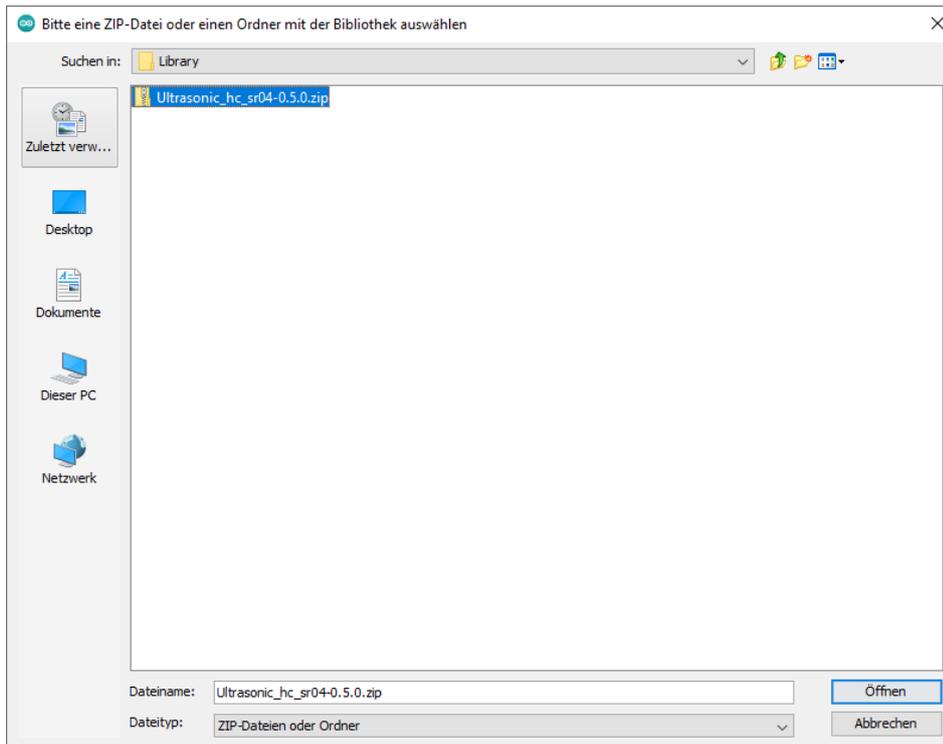


.zip Bibliothek importieren

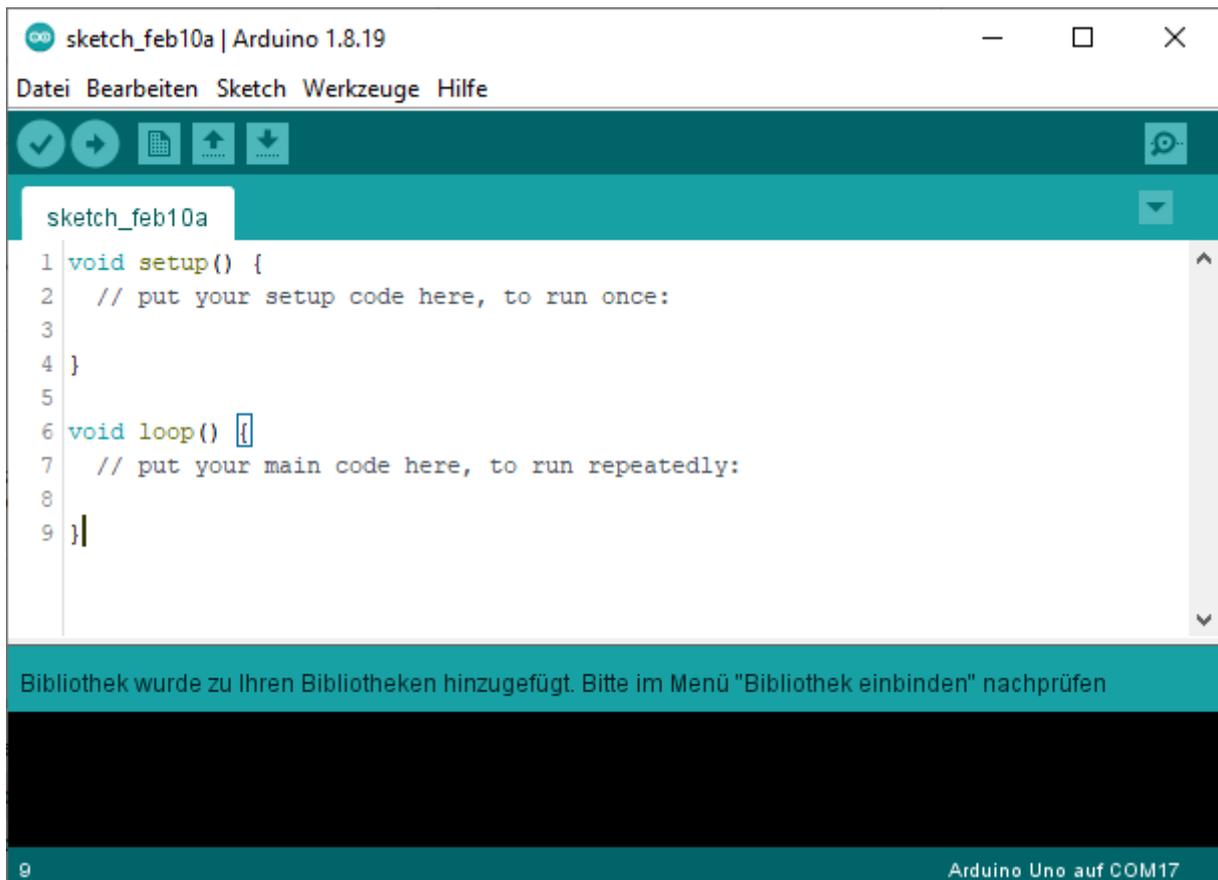
Es kann vorkommen, dass eine Bibliothek nicht im Bibliotheksverwalter zu finden ist. In diesem Fall muss man auf eine externe Quelle zurückgreifen und die Bibliothek importieren. Hierzu in der Arduino IDE unter „Sketch“ zu „Bibliothek einbinden“ navigieren und **„.ZIP Bibliothek hinzufügen...“** anklicken.



Sie werden aufgefordert mit dem Dateixplorer eine ZIP Bibliothek zu öffnen. Navigieren Sie zum entsprechenden Dateipfad und wählen Sie eine ZIP Bibliothek aus.



Wenn die Bibliothek erfolgreich eingebunden wurde, erscheint im Hauptfenster folgende Nachricht:



Nun Können Sie, wie aufgefordert, unter „Sketch -> Bibliotheken einbinden“ nach unten scrollen um zu schauen ob die Bibliothek aufgelistet wird.

Hinweis: Die Beispiele unter „Datei – Beispiele“ tauchen erst nach einem Neustart der Arduino auf.

Bibliotheken manuell installieren

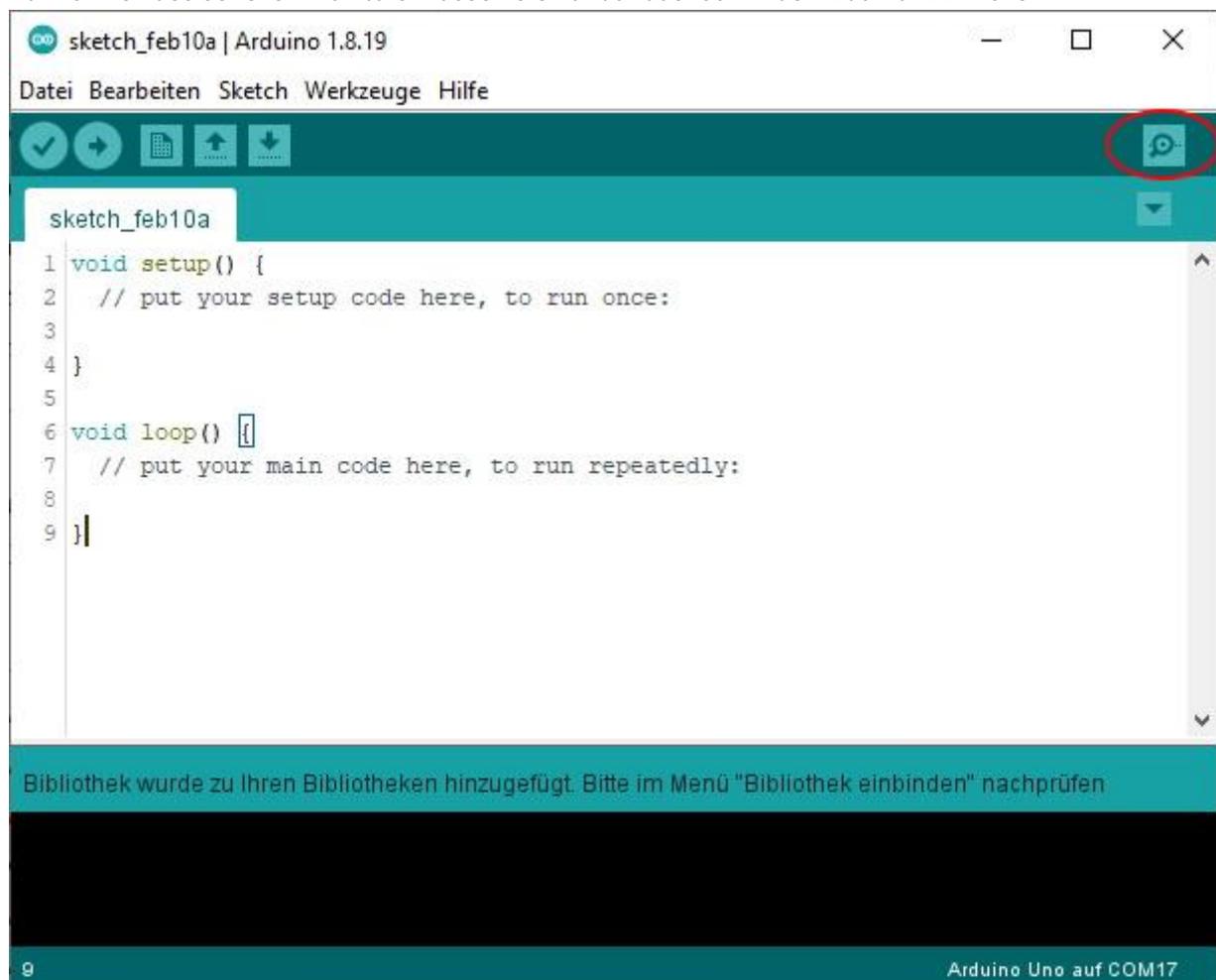
Um Bibliotheken manuell zu installieren, müssen Sie die .ZIP Datei entpacken und den Ordner nach „Dokumente/Arduino/Libraries“ kopieren. Wichtig ist, dass der Ordner wie die Bibliothek heißt und eine .cpp und .h Datei enthält. Nach einem Neustart befindet sich die Bibliothek nun ebenfalls unter „Sketch – Bibliotheken einbinden“ in der Liste.

Serieller Monitor

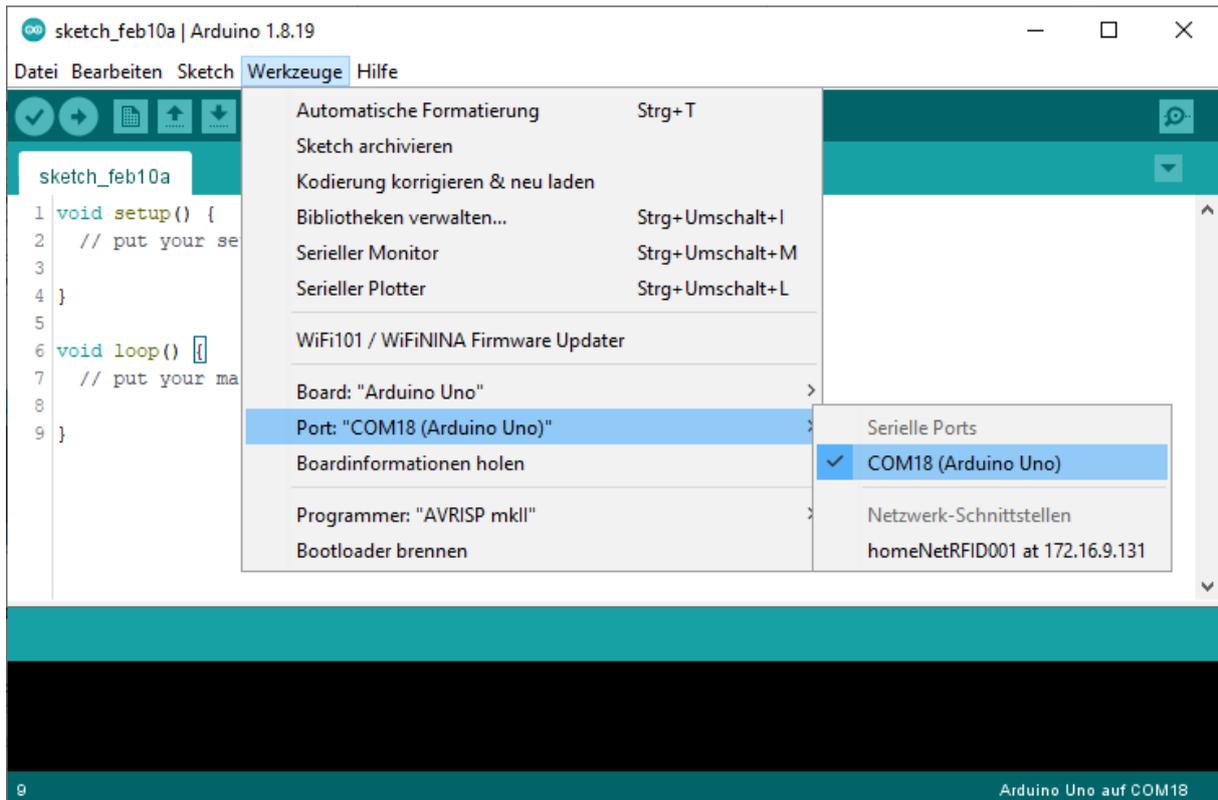
Der serielle Monitor ist ein wichtiger Teil der Arduino IDE, um mit angeschlossenen Microcontrollern kommunizieren und sich auf Fehlersuche begeben zu können. Mit diesem Werkzeug können Sie ganz schnell Messwerte ausgeben lassen, dem Arduino Befehle erteilen oder Programmfunktionen auf Fehler überprüfen.

Eine Verbindung herstellen

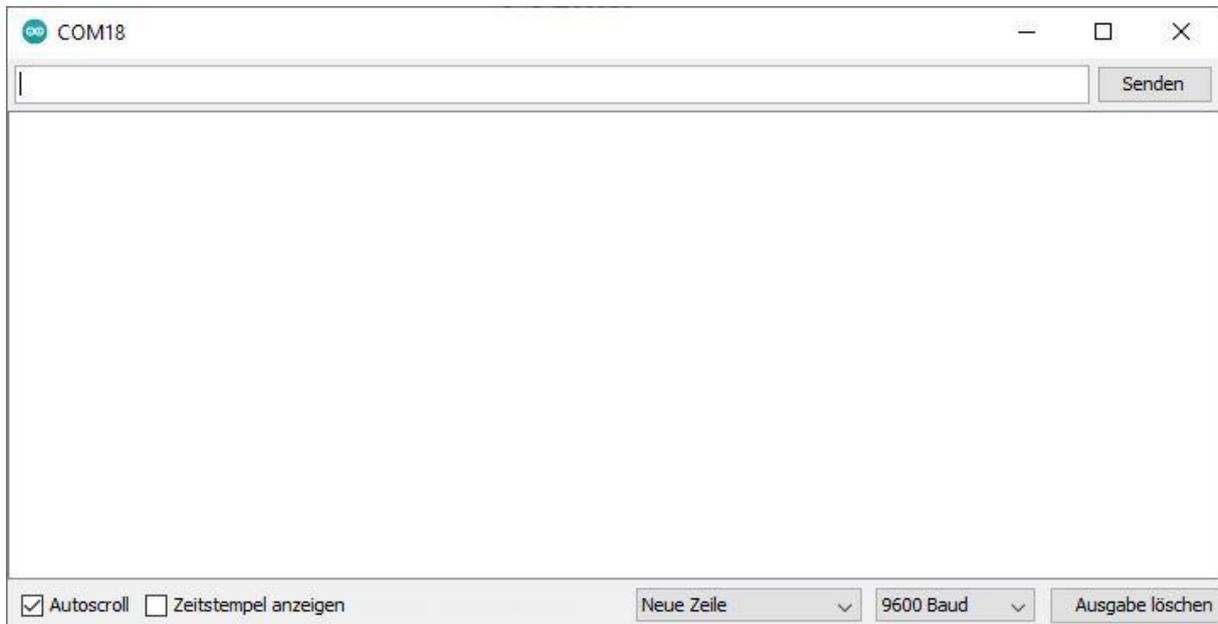
Zum Öffnen des seriellen Monitors müssen Sie nur auf das Icon in der Arduino IDE klicken



Die Port-Auswahl vom seriellen Monitor funktioniert wie beim Hochladen von Arduino Codes.

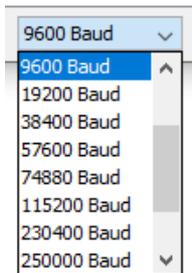


Einmal geöffnet sieht er ungefähr so aus:



Einstellungen des seriellen Monitors

Der serielle Monitor der Arduino IDE hat nur wenige Einstellungen, die aber für die meisten Anwendungen ausreichen dürften. Die wichtigste Einstellung ist die baud rate.



Mit der baud rate wird die **Datenrate in Bit pro Sekunde** für die serielle Datenübertragung festgelegt. Wenn die baud rate falsch eingestellt ist, werden die Daten nicht korrekt angezeigt und die Kommunikation wird nicht funktionieren. **Autoscroll** ist sehr praktisch, kann aber bei Bedarf ausgeschaltet werden.

Vorteile vom seriellen Monitor

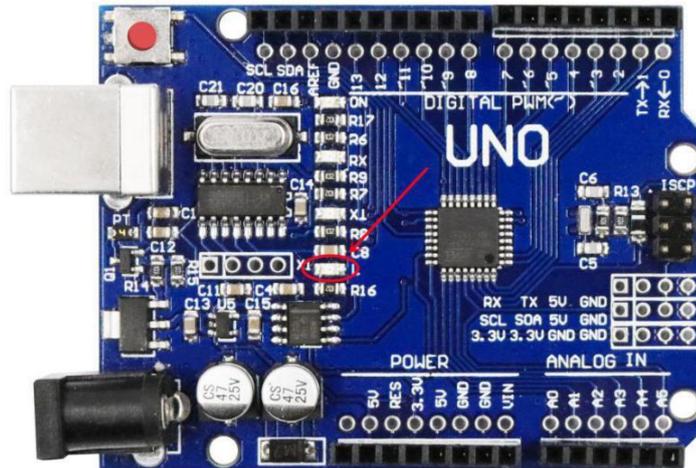
Der serielle Monitor ist eine gute und sehr einfache Möglichkeit eine serielle Verbindung mit Ihrem Arduino herzustellen. Wenn Sie sowieso in der Arduino IDE programmieren, liegt es nahe die integrierten Werkzeuge zu verwenden.

Nachteile vom seriellen Monitor

Durch die dürftigen Einstellungsmöglichkeiten eignet sich der serielle Monitor nicht für fortgeschrittene Anwendungen. Die Grenzen bereits sind schnell erreicht, wenn Ihre Anwendung eine nicht aufgeführte baud rate verlangt.

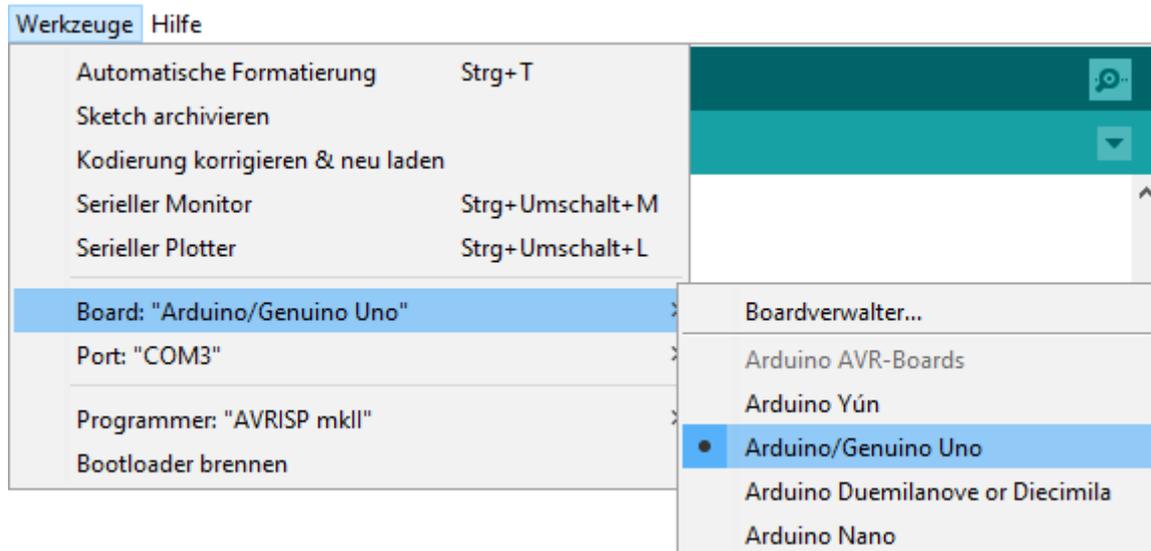
4. Blink Beispiel

In diesem Kapitel zeigen wir Ihnen durch das beliebte Blink Beispiel, wie Sie das Arduino UNO kompatible Board über die offizielle Software programmieren. Auf dem Arduino UNO ist eine kleine LED angebracht, die wir steuern möchten. Standardmäßig wird das Board bereits mit dem Blink Sketch ausgeliefert, also sollte die LED bereits blinken sobald es angeschlossen wird.



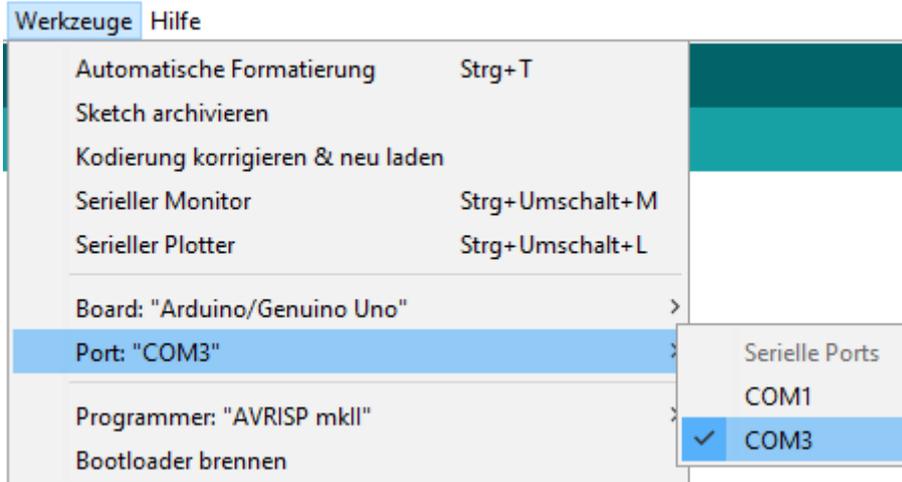
Nachdem wir die Arduino-Software installiert haben, schließen wir das Arduino UNO Board über USB an den Computer an.

Nun öffnen wir die Arduino-Software und stellen sicher, dass bei "Werkzeuge - Board" der Arduino UNO ausgewählt ist. und der serielle Port richtig eingestellt ist. Unter Windows ist es meistens Port "COM 3" oder höher, denn Port 1-2 sind für das System reserviert.



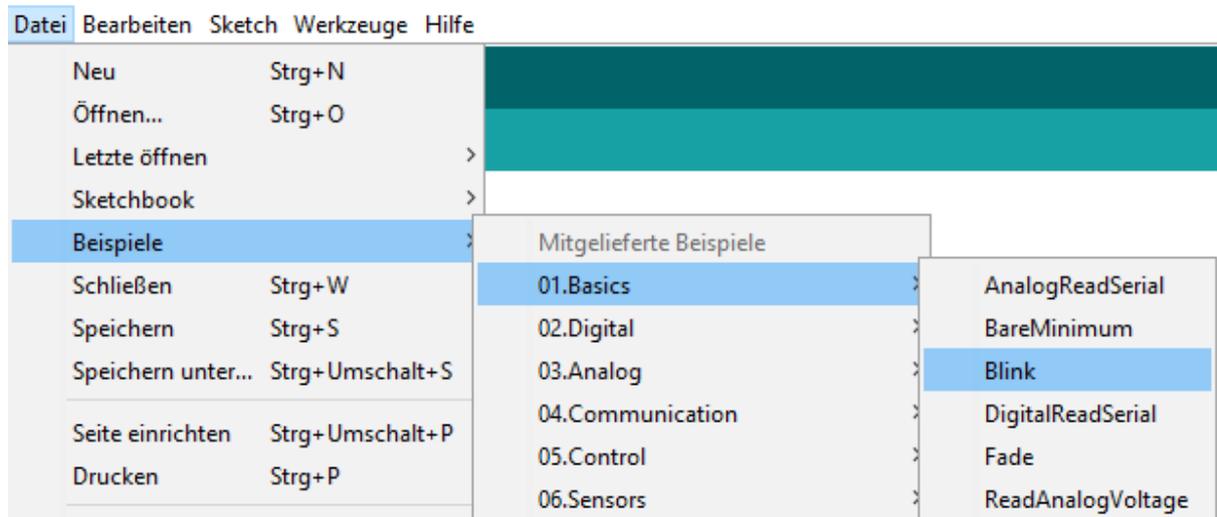
Als nächstes stellen wir den richtigen Port ein. Unter Windows ist es meistens Port "COM 3" oder höher, denn Port 1-2 sind in der Regel für das System reserviert.

Achtung: Der richtige Port auf Ihrem System wird vermutlich nicht derselbe wie auf den Screenshots sein!



Die Arduino-Software liefert einige Programmbeispiele mit, für dieses Projekt benutzen wir einfach das "Blink"-Beispiel.

Für dieses Beispiel klicken wir oben links auf "Datei" und wählen unter dem Reiter "Beispiele - 01.Basics - Blink" aus.



Dann öffnet sich ein neuer Sketch, welcher hier nochmal mit deutschen Kommentaren versehen ist:

```

/*
  Blink
  Schaltet eine LED für 1 Sekunde an und für eine Sekunde aus.
*/

// Die Setup-Funktion wird einmalig nach jedem Start oder Reset ausgeführt
void setup() {
  // digitaler pin 13 wird als Output deklariert
  pinMode(13, OUTPUT);
}

// Die Loop-funktion wird immer und immer wieder ausgeführt
void loop() {
  digitalWrite(13, HIGH); // Schaltet die LED an (HIGH ist die Spannungsversorgung)
  delay(1000);           // 1000 ms (= 1 Sekunde) Verzögerung
  digitalWrite(13, LOW); // Schaltet die LED aus indem die Spannungsversorgung auf LOW gesetzt wird
  delay(1000);           // 1 Sekunde Verzögerung
}

```

}

Hinweis: Die grau hinterlegten **Kommentare nach // haben keine Programmfunktion**. Sie dienen nur der Erklärung des Programms. Zwischen /* und */ befinden sich Block-Kommentare, die sich über mehrere Zeilen erstrecken können.

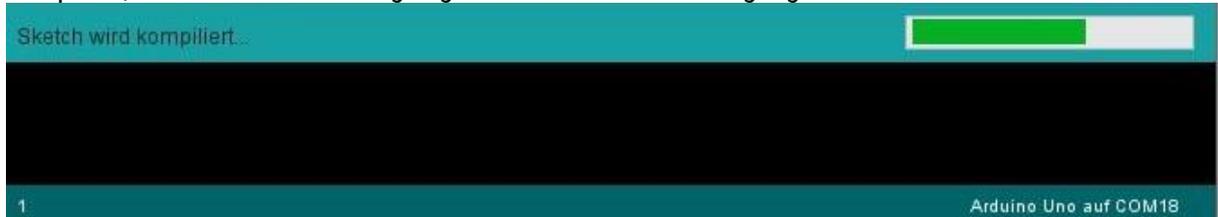
Jeder Arduino Sketch benötigt eine **„Setup“-Funktion**, die einmalig beim Start ausgeführt wird. In unserem Beispiel wird hier nur die Anweisung erteilt, dass der Pin 13 als Ausgang deklariert wird. Weitere Anweisungen können Sie zwischen den geschweiften Klammern hinzufügen.

Die **„Loop“-Funktion** ist ebenfalls Pflicht und wird nach dem Setup immer wieder ausgeführt. In der Loop-Funktion wird die LED angeschaltet (Pin 13 auf HIGH), eine Pause von 1000ms eingelegt und die LED ausgeschaltet (Pin 13 auf LOW).

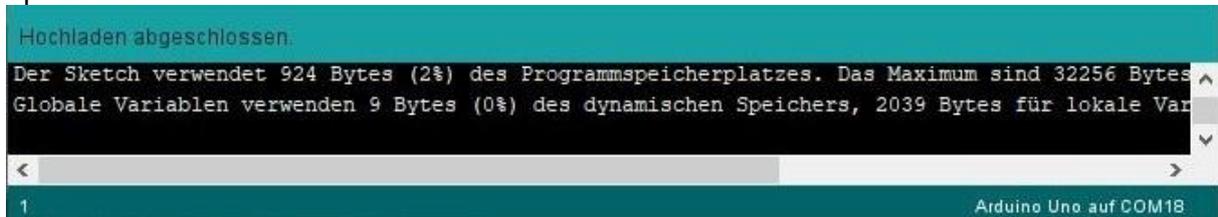
Dieses kleine Programm laden wir nun auf unser Board, indem wir auf die Schaltfläche klicken:



Im unteren Bereich können Sie den Fortschritt des Vorgangs verfolgen. Der Sketch wird zunächst kompiliert, damit der Code in ein geeignetes Format zur Übertragung konvertiert wird.



Im Anschluss wird der Sketch hochgeladen, die anderen LEDs auf dem Board sollten während dem Upload flackern.



Fehlerbehebung

Die folgende Fehlermeldung bedeutet, dass das Board nicht angeschlossen ist.



Mögliche Fehlerursachen und Lösungen

- **Board nicht angeschlossen.** Manchmal hilft es auch einen anderen USB-Port auszuprobieren.

- **Treiber nicht richtig installiert.** In diesem Fall müssen Sie im Geräte-Manager Ihr Arduino finden und im Fenster „Eigenschaften“ den Treiber deinstallieren und anschließen neu installieren.
- **Falscher Port ausgewählt.** In der Arduino IDE unter „Werkzeuge“ den Port erneut überprüfen.
- **Falsches Board ausgewählt.** In der Arduino IDE unter „Werkzeuge“ das Board erneut überprüfen.



```

expected ';' before ')' token
^
exit status 1
expected ';' before ')' token
  
```

Fehlermeldungen kopieren

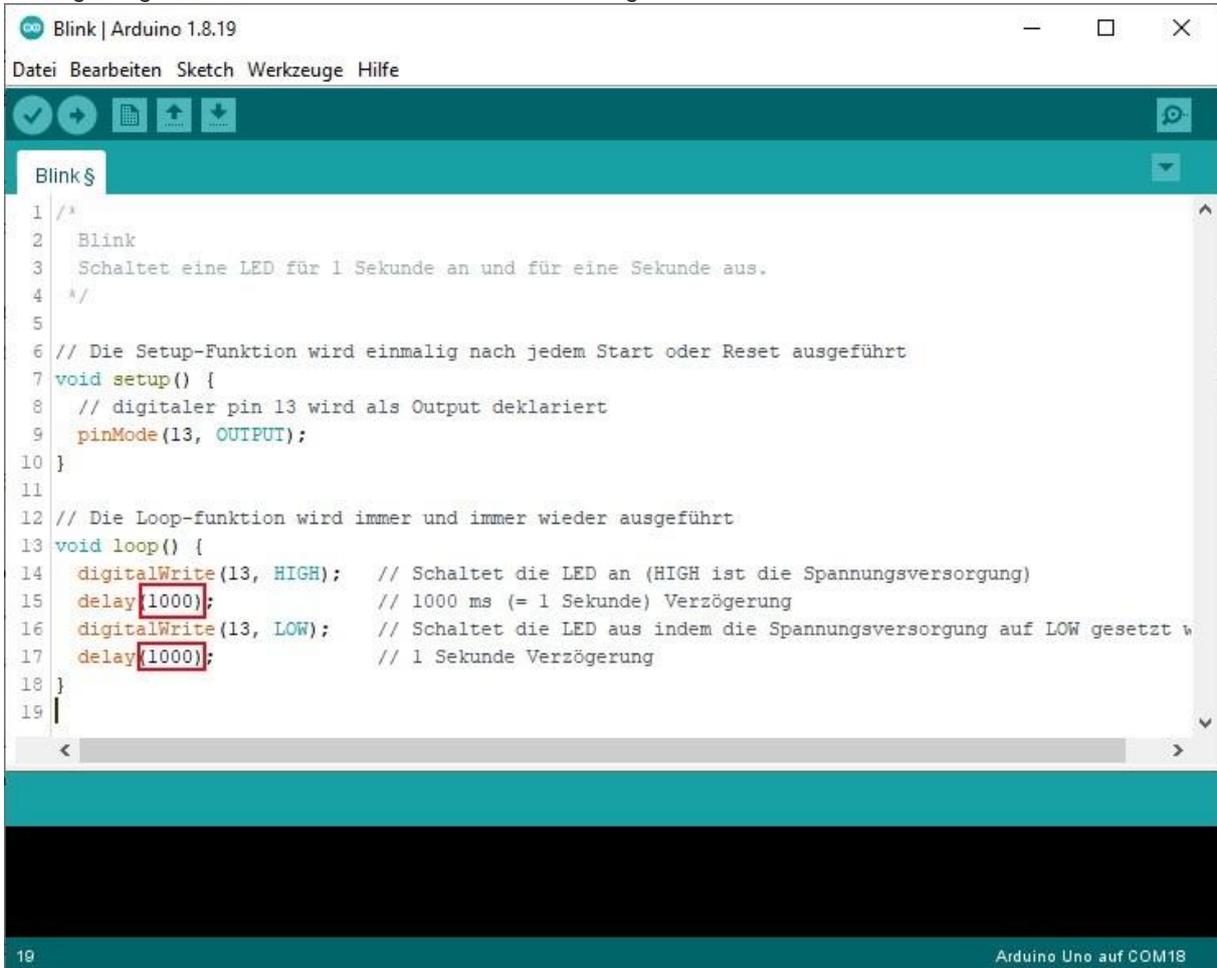
29 Arduino Uno auf COM18

Diese Fehlermeldung wird jeder früher oder später erhalten. In der Programmiersprache von Arduino muss hinter jeder Zeile, also jedem Befehl ein Semikolon „;“ stehen. Andernfalls bricht die Kompilierung ab und erzeugt diese Fehlermeldung.

Lösung: Die entsprechende Zeile wird in der Arduino rot hinterlegt und kann korrigiert werden.

Ergebnis

Wenn alles funktioniert hat, blinkt unsere LED im 1-Sekunden-Takt. Als nächstes können Sie die Verzögerungen ändern und die LED schneller oder langsamer blinken lassen.



```
1 /*
2  Blink
3  Schaltet eine LED für 1 Sekunde an und für eine Sekunde aus.
4  */
5
6 // Die Setup-Funktion wird einmalig nach jedem Start oder Reset ausgeführt
7 void setup() {
8   // digitaler pin 13 wird als Output deklariert
9   pinMode(13, OUTPUT);
10 }
11
12 // Die Loop-funktion wird immer und immer wieder ausgeführt
13 void loop() {
14   digitalWrite(13, HIGH); // Schaltet die LED an (HIGH ist die Spannungsversorgung)
15   delay(1000); // 1000 ms (= 1 Sekunde) Verzögerung
16   digitalWrite(13, LOW); // Schaltet die LED aus indem die Spannungsversorgung auf LOW gesetzt w
17   delay(1000); // 1 Sekunde Verzögerung
18 }
19 |
```

Ändern Sie den Delay von 1000 auf 500, um die LED alle 500 ms blinken zu lassen. Nach der Änderung den Sketch erneut uploaden und Sie müssten sehen, dass die LED doppelt so schnell blinkt.

5. Servo

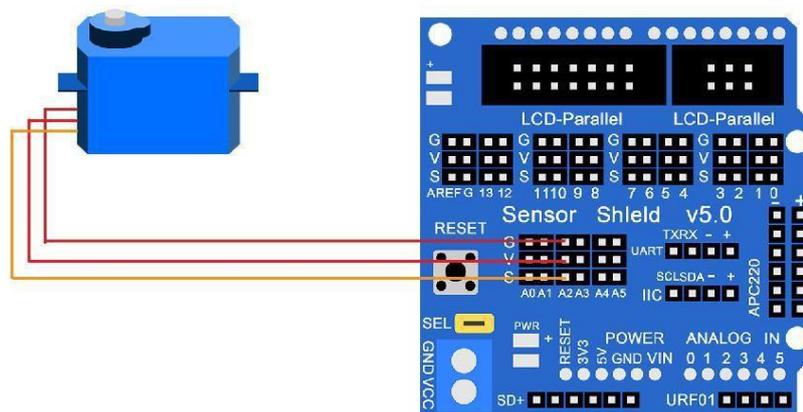
In diesem Kapitel zeigen wir anhand der Servo.h Bibliothek wie man einen Servomotor mit einem Arduino kompatiblen Board ansteuert. Servomotoren sind sehr vielseitig einsetzbar, sie sind beliebt in RC-Modellen, Robotern, Automaten und bringen Bewegung in sämtlich DIY-Elektronik Projekte.

Funktionsweise

Ein Servomotor (kurz Servo) ist ein Elektromotor, der sehr genau angesteuert werden kann. Neben der exakten Winkelposition kann auch die Drehgeschwindigkeit und Beschleunigung gesteuert werden. Der SG90 Micro Servo besteht aus einem kleinen DC-Motor, einem Getriebe und einer Steuerplatine mit Potentiometer.

Die Steuerplatine setzt die Signale in genaue Befehle um und über das Potentiometer wird die Position überprüft. Die üblichen Hobbyservos besitzen drei Pins zur Ansteuerung: GND, VCC und PWM. Gelegentlich findet man einen vierten Pin vor, dieser dient als Feedback-Pin, um die Position des Potentiometers über einen Microcontroller abzurufen.

Anschlussplan

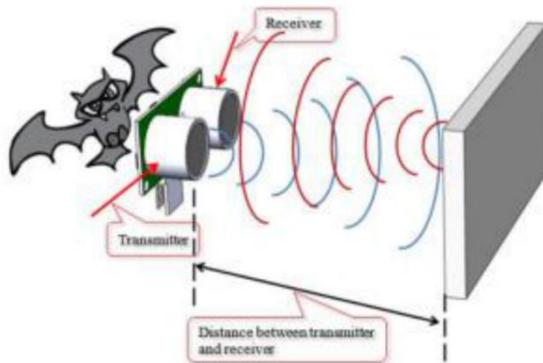


Servomotor	Arduino
Gelb (PWM-Signalleitung)	Pin 2
Rot (VCC)	5V
Braun (GND)	GND

Code

Laden Sie Code für Lektion 5 Servo auf den Arduino. Ohne die Servo.h Bibliothek funktioniert der Code nicht, wie Sie Bibliotheken installieren, können Sie in Lektion 2 nachlesen. Wenn Sie Probleme beim Hochladen haben, schlagen Sie in Lektion 3 nach.

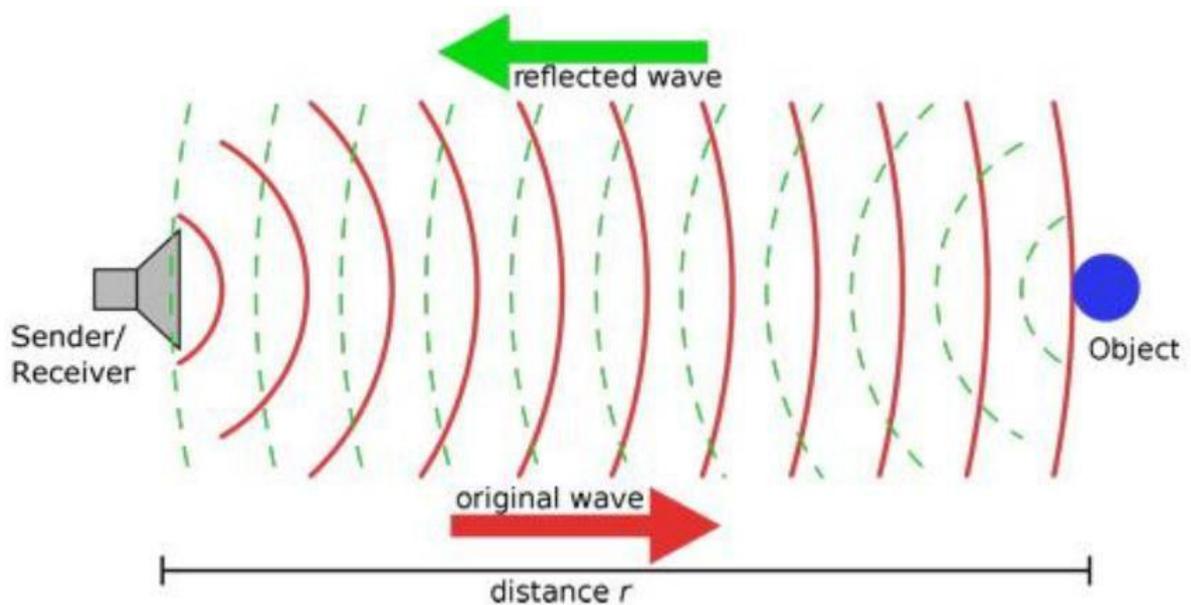
6. Ultraschallsensor



Ultraschallsensoren sind perfekt für Elektronik-Projekte, bei denen Abstände gemessen oder Hindernisse erkannt werden sollen. Der HC-SR04 ist ein günstiger, einfach zu nutzender Ultraschallsensor mit einer Messreichweite von 2 cm bis 400 cm und ca. 3mm Genauigkeit. Er besteht aus einem Ultraschall-Sender und Ultraschall-Empfänger auf einer Steuerplatine. Zur einfachen und schnellen Programmierung wird die HC-SR04 Bibliothek für Arduino verwendet.

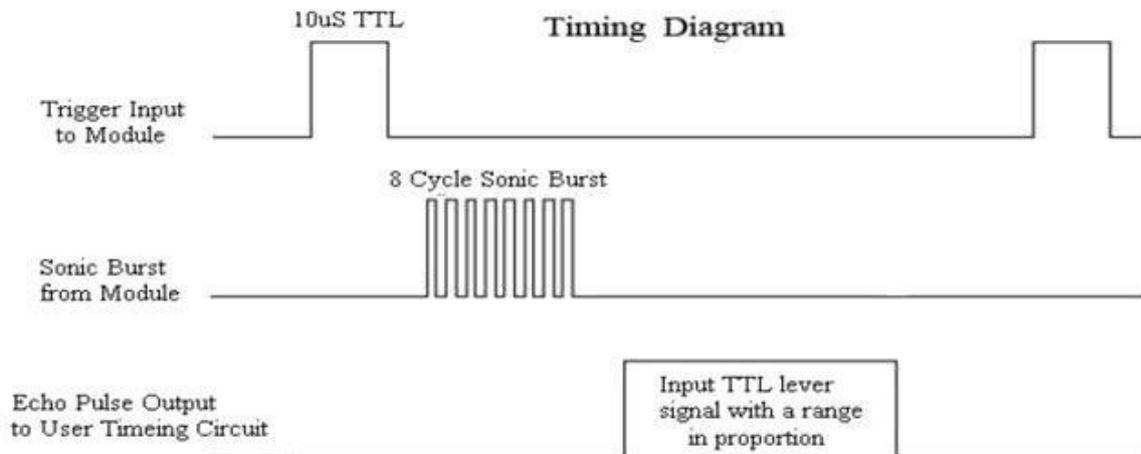
Funktionsweise

Das HC-SR04 Ultraschallmodul misst selbstständig die Entfernung mit einer Messzeit von 10ms Intervallen. Pro Intervall werden acht 40KHz Schallimpulse ausgesendet, um die reflektierten Schallimpulse aufzufangen und aus der vergangenen Zeit zwischen Senden und Empfangen die Entfernung zu berechnen.

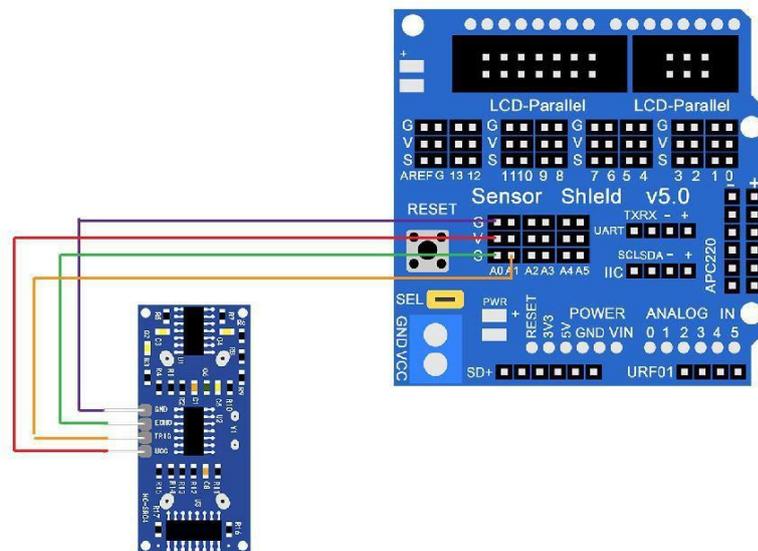


Der Echo-Pin des HC-SR04 sendet nach der Messung ein HIGH Signal. Die Dauer des HIGH-Pegels bestimmt die Entfernung nach folgender Formel:

$$\text{Entfernung} = ((\text{Dauer_High_Pegel}) * (\text{Sonic: } 340\text{m/s})) / 2$$



Anschlussplan



Ultraschallsensor	Arduino
Echo	Pin 7
Trig	Pin 8
VCC	5V
GND	GND

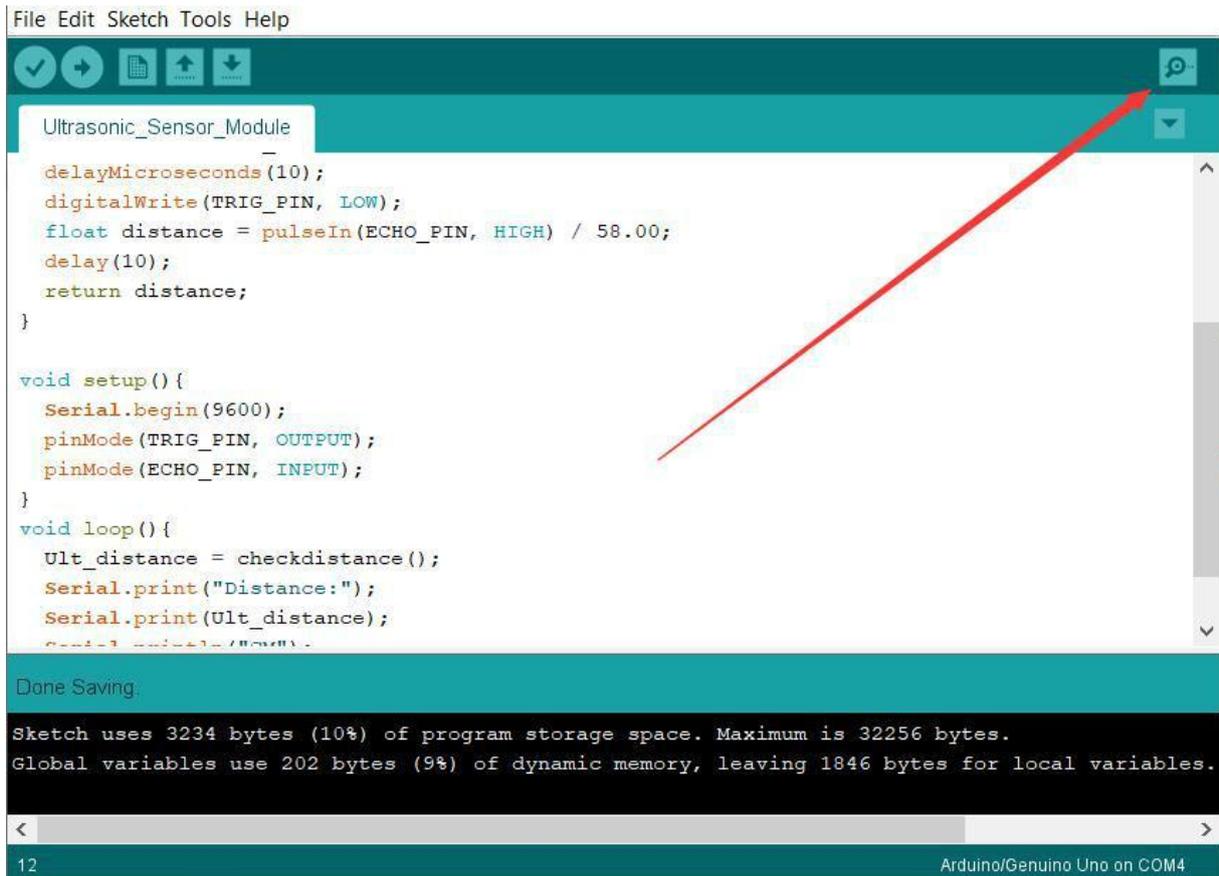
Code

Durch Verwendung der HC-SR04 Bibliothek ist der Code kurz und knapp. Die Bibliothek wird am Anfang vom Code eingebunden, um die Funktionen des Sensors einfach abrufen zu können.

Ohne die <HC-SR04> Bibliothek funktioniert der Code nicht, wie Sie Bibliotheken installieren, können Sie in Lektion 2 nachlesen. Wenn Sie Probleme beim Hochladen haben, schlagen Sie in Lektion 3 nach.

Ergebnis

Nachdem der Code hochgeladen wurde, können Sie die gemessene Distanz im seriellen Monitor auslesen.



```
File Edit Sketch Tools Help
[Icons] [Search Icon]
Ultrasonic_Sensor_Module [Dropdown]
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
float distance = pulseIn(ECHO_PIN, HIGH) / 58.00;
delay(10);
return distance;
}

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}

void loop() {
  Ult_distance = checkdistance();
  Serial.print("Distance:");
  Serial.print(Ult_distance);
  Serial.print("\n");
}
```

Done Saving.

Sketch uses 3234 bytes (10%) of program storage space. Maximum is 32256 bytes.
Global variables use 202 bytes (9%) of dynamic memory, leaving 1846 bytes for local variables.

< 12 Arduino/Genuino Uno on COM4 >

COM4

Send

```
Distance:2CM
Distance:3CM
Distance:3CM
Distance:3CM
Distance:3CM
Distance:3CM
Distance:5CM
Distance:6CM
Distance:8CM
Distance:11CM
Distance:14CM
Distance:18CM
Distance:23CM
Distance:27CM
Distance:31CM
```

Autoscroll Show timestamp

Newline 9600 baud Clear output

7. Infrarot-Fernbedienung

In diesem Kapitel zeigen wir Ihnen, wie sie den Infrarotempfänger und die Infrarotfernbedienung mit Ihrem Arduino UNO kompatiblen Board verwenden. Infrarotfernbedienungen sind aus dem Alltag nicht wegzudenken und funktionieren auch super in der DIY-Elektronik, um Projekte zu steuern.

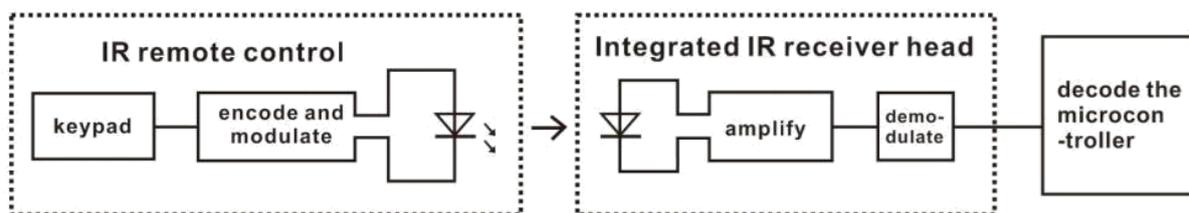
Wie funktioniert eine Infrarot-Fernbedienung?

IR-Fernbedienungen senden binäre Impulse als Signal im unsichtbaren Infrarotbereich aus. Um Störsignale von anderen Quellen zu vermeiden, werden die Signale vormoduliert und erst dann mit Hilfe einer IR-LED in einer bestimmten Frequenz gesendet. Der IR-Empfänger besitzt in der Regel eine IR-Fotodiode, die Signale auf einer bestimmten Frequenz empfängt und Störungen von anderen Lichtquellen filtert. Hinter der IR-Fotodiode befindet sich dann ein Signalverstärker und Demodulator, um das Signal weiterverarbeiten zu können.

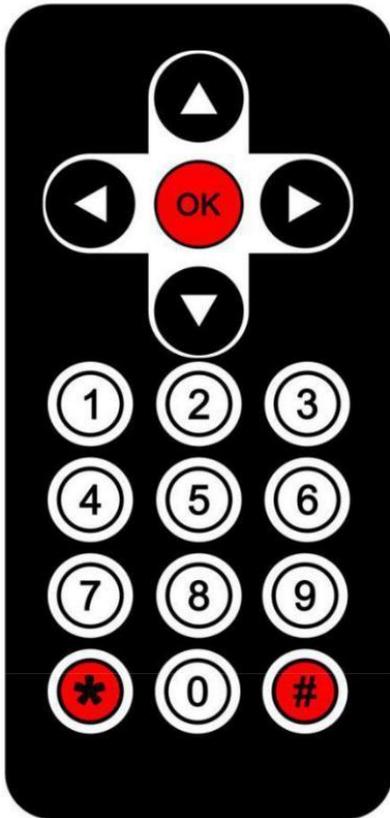
Das Infrarotsignal wird von einem Kodierchip auf der Fernbedienung kodiert. Das Zeitintervall zwischen den Impulsen des Signals wird verwendet, um zu unterscheiden, ob es sich um 0 oder 1 handelt. Wenn das Verhältnis zwischen hohem Pegel und niedrigem Pegel etwa 1:1 ist, wird das Signal als 0 betrachtet.

Die Kodierung der Fernbedienung setzt sich aus den Signalen 0 und 1 zusammen, wobei der Code für jede Taste gleichbleibt. Mit Hilfe der verschiedenen Codes wird die gedrückte Taste der Fernbedienung unterschieden. Wenn eine Taste auf der Fernbedienung gedrückt wird, sendet sie das entsprechende Infrarotsignal aus.

Anschließend wird das gesendete Infrarotsignal mit dem Infrarotempfänger aufgefangen. Der Empfänger demoduliert das Signal und sendet es an unseren Arduino UNO. Dieser kann dann auf Grund der verschiedenen Datencodes erkennen, welche Taste gedrückt wurde.

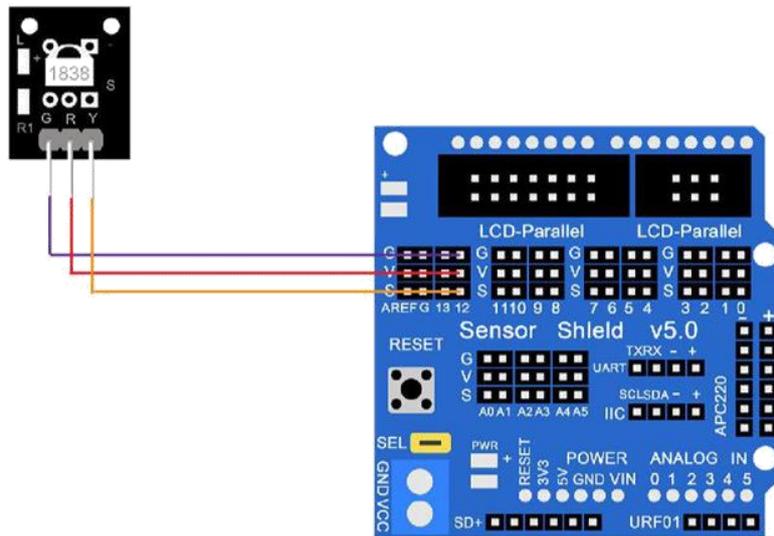


Tasten-Codes der Fernbedienung



 FF629D	 FF22DD	 FFC23D
 FFA857	 FF02FD	 FF6897
 FF9867	 FFBD4F	 FF30CF
 FF18E7	 FF7A85	 FF10EF
 FF38C7	 FF5AA5	 FF4AB5
 FF42BD	 FF52AD	

Anschlussplan



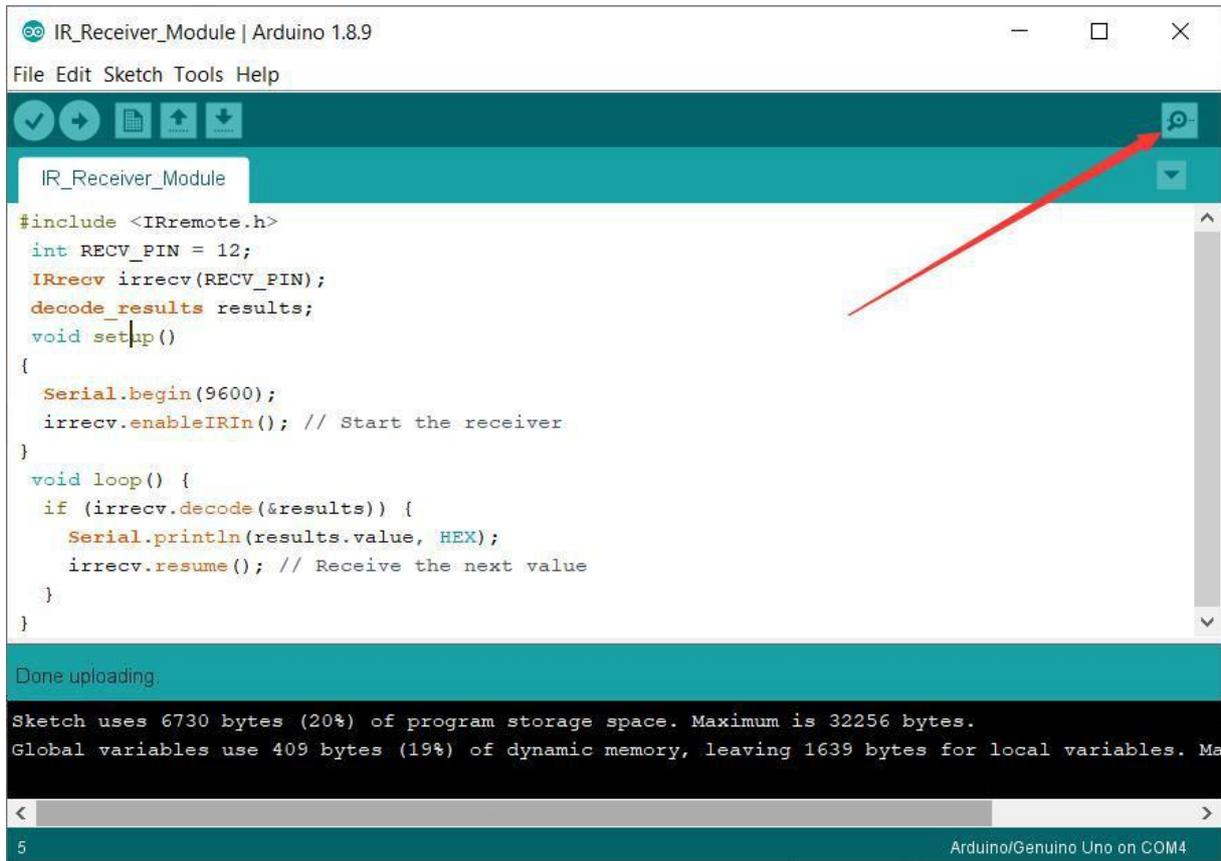
IR-Empfänger	Arduino UNO
S oder Y(Signal)	11
- oder G(GND)	GND
+ oder R(VCC)	5V

Hinweis: Je nach Charge sind die Pins anders beschriftet.

Code

Wir benutzen eine Fernbedienung mit 17 Tasten und einer Reichweite von maximal 8 Metern. In diesem Beispiel werden die Signale der Fernbedienung dekodiert, um die Tasten frei programmieren zu können. Wenn Sie die Tasten zu lange gedrückt halten, entstehen Fehleingaben.

Nachdem der Code hochgeladen wurde, können Sie die gesendeten Codes der Fernbedienung im seriellen Monitor auslesen.



```

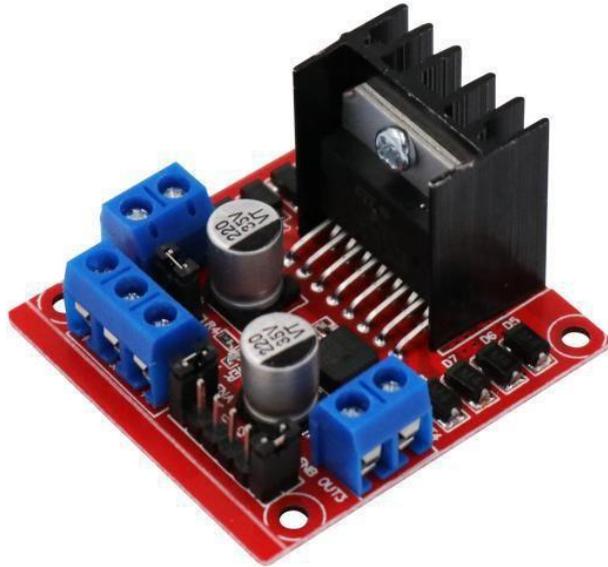
IR_Receiver_Module | Arduino 1.8.9
File Edit Sketch Tools Help
IR_Receiver_Module
#include <IRremote.h>
int RECV_PIN = 12;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}
void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
}
Done uploading.
Sketch uses 6730 bytes (20%) of program storage space. Maximum is 32256 bytes.
Global variables use 409 bytes (19%) of dynamic memory, leaving 1639 bytes for local variables. Max
5 Arduino/Genuino Uno on COM4
  
```



```

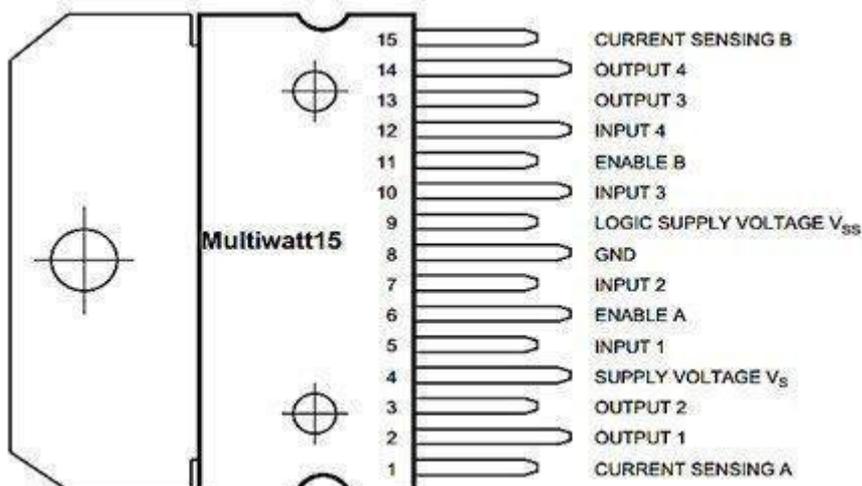
COM9 (Arduino/Genuino Uno)
发送
FF5AA5
FF5AA5
FF5AA5
FF5AA5
FF38C7
FF38C7
FF10EF
FFFFFFF
FF42BD
FF4AB5
FF4AB5
FFFFFFF
FF4AB5
自动滚屏 没有结束符 9600 波特率 Clear output
  
```

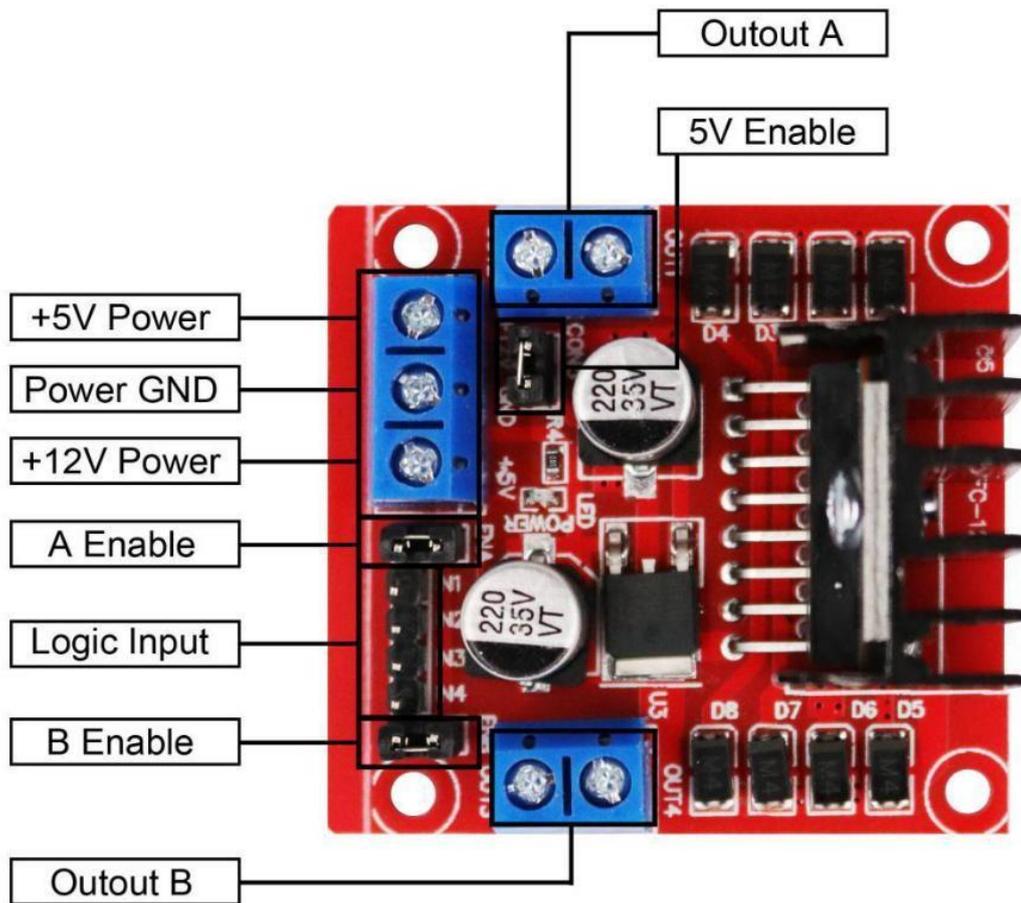
8. L298N Motortreiber



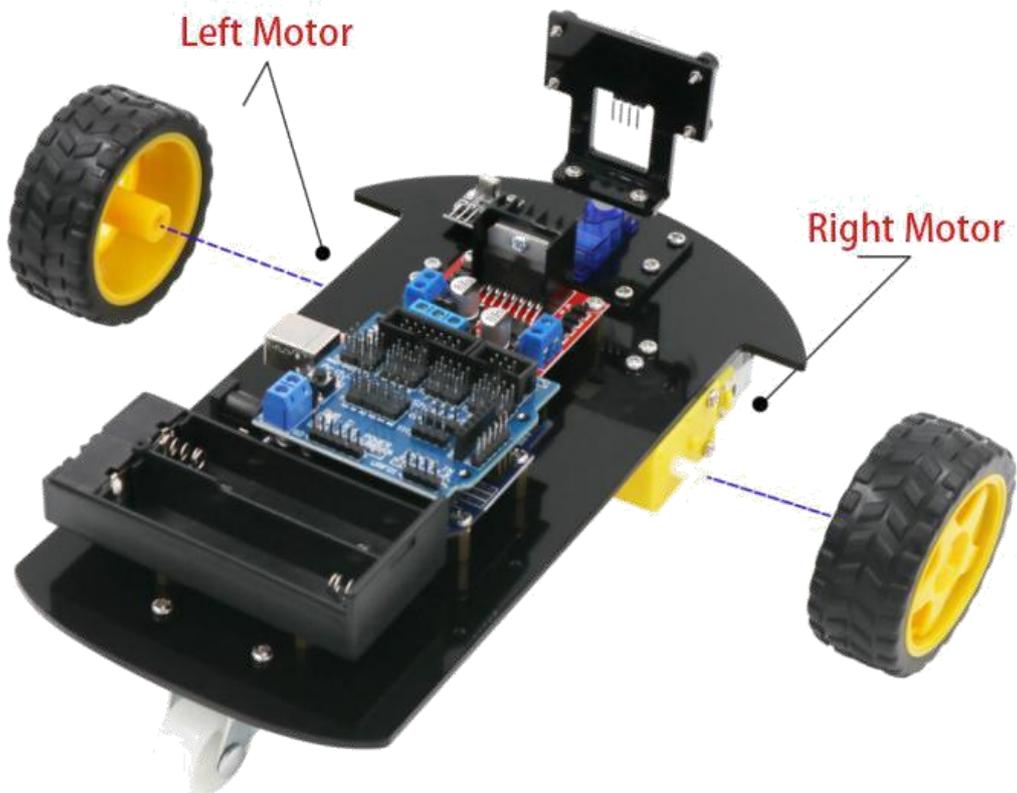
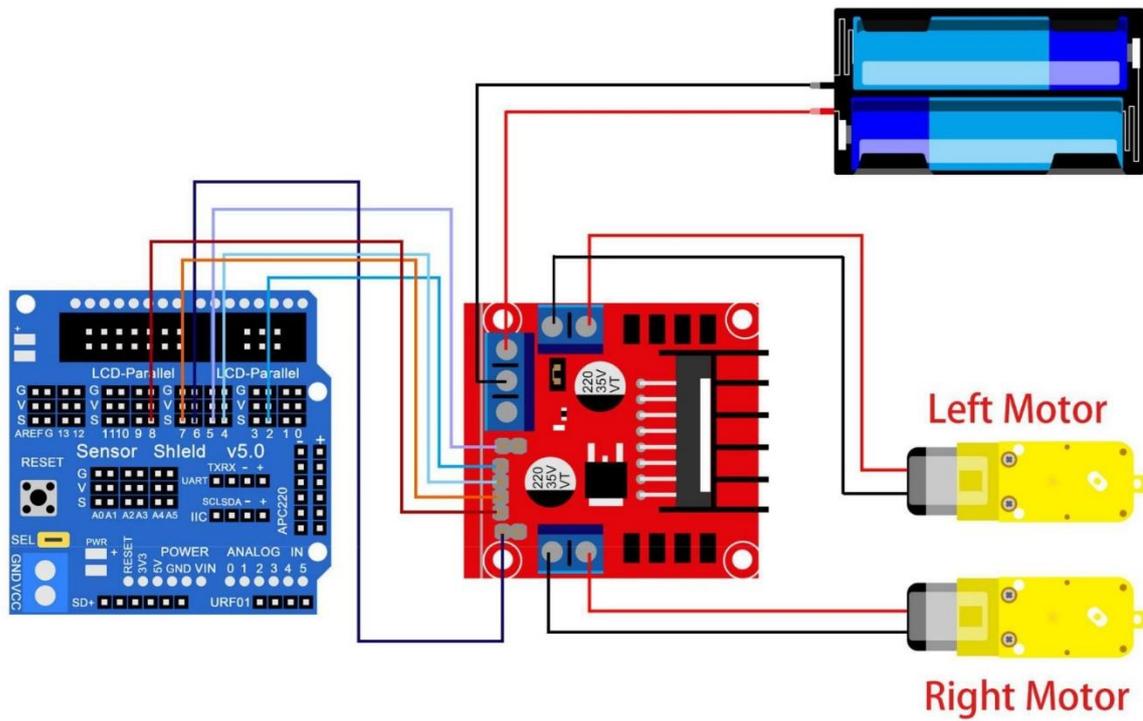
Mit dem L298N Modul lassen sich zwei DC Motoren unabhängig voneinander und in beide Richtungen steuern. Das L298N Modul besteht aus einer doppelten H-Brücke und eignet sich ideal für Robotik-Projekte und ferngesteuerte Autos. Mit dem L298N Modul können Sie auch einen Schrittmotor steuern, darauf gehen wir in dieser Anleitung aber nicht ein.

Pinbelegung





Anschlussplan



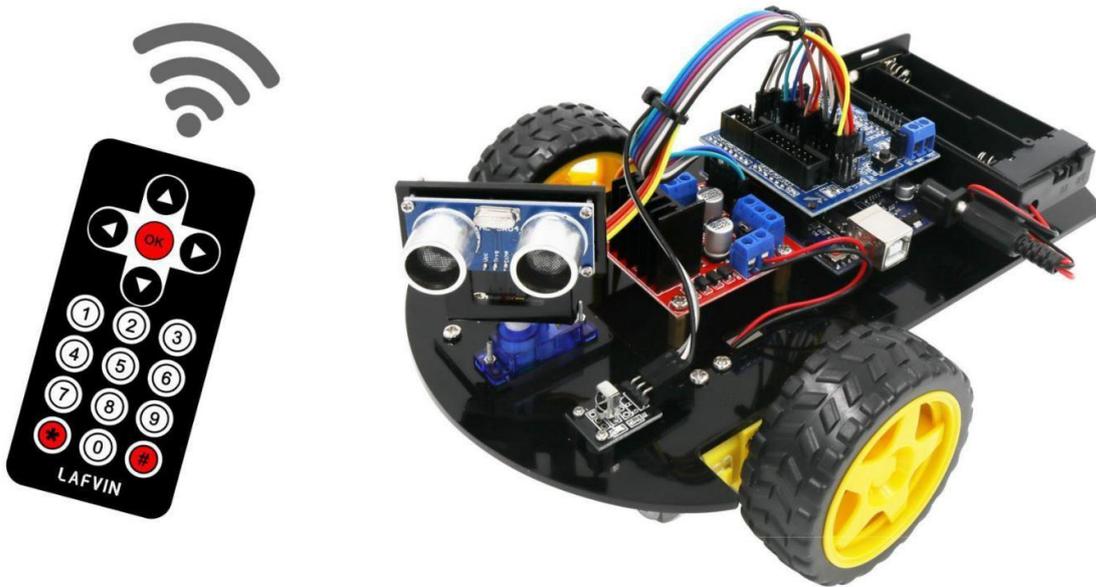
Code

Nach der Verkabelung können Sie den L2098N Code öffnen und hochladen. Sollten Sie Probleme beim Hochladen haben, können Sie in Lektion 3 nachschlagen.

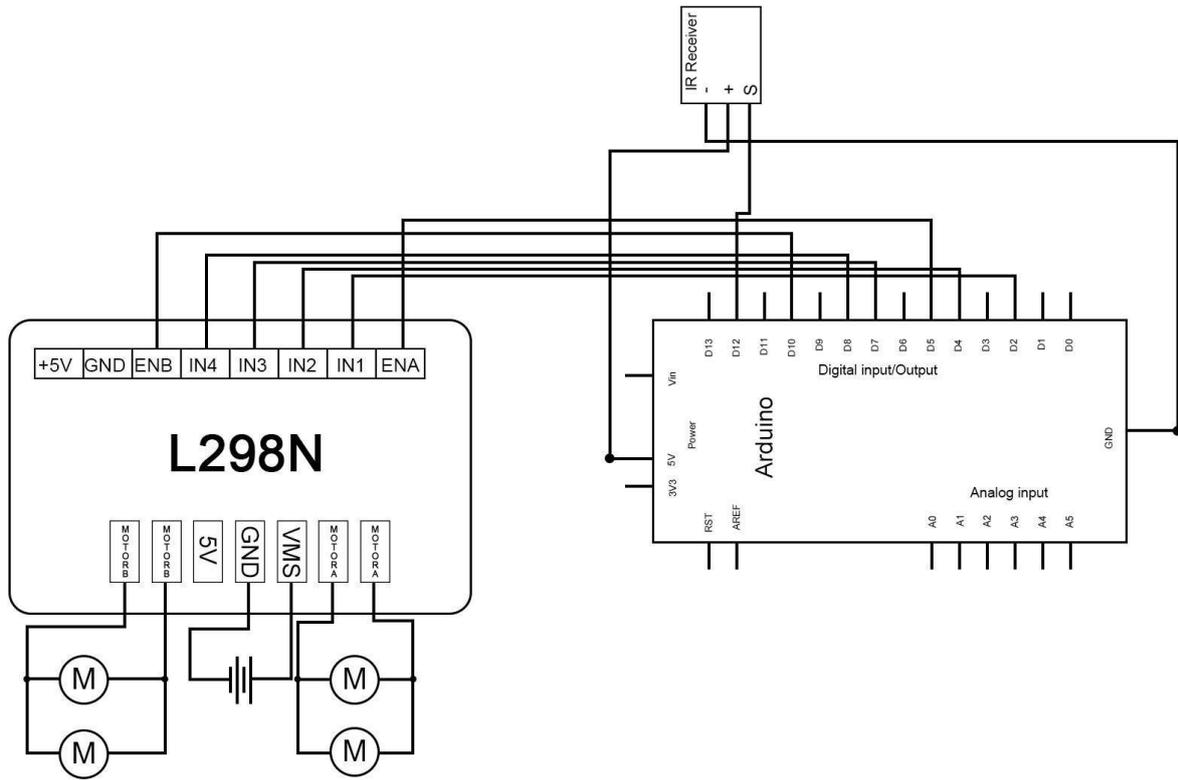
Wenn alles funktioniert hat, sollten die Motoren 2 Sekunden im Uhrzeigersinn mit einer Geschwindigkeit von 200 (PWM-gesteuert) drehen und für 2 Sekunden anhalten. Danach wieder für 2 Sekunden gegen den Uhrzeigersinn mit einer Geschwindigkeit von 100.

9. Auto mit Infrarotfernbedienung steuern

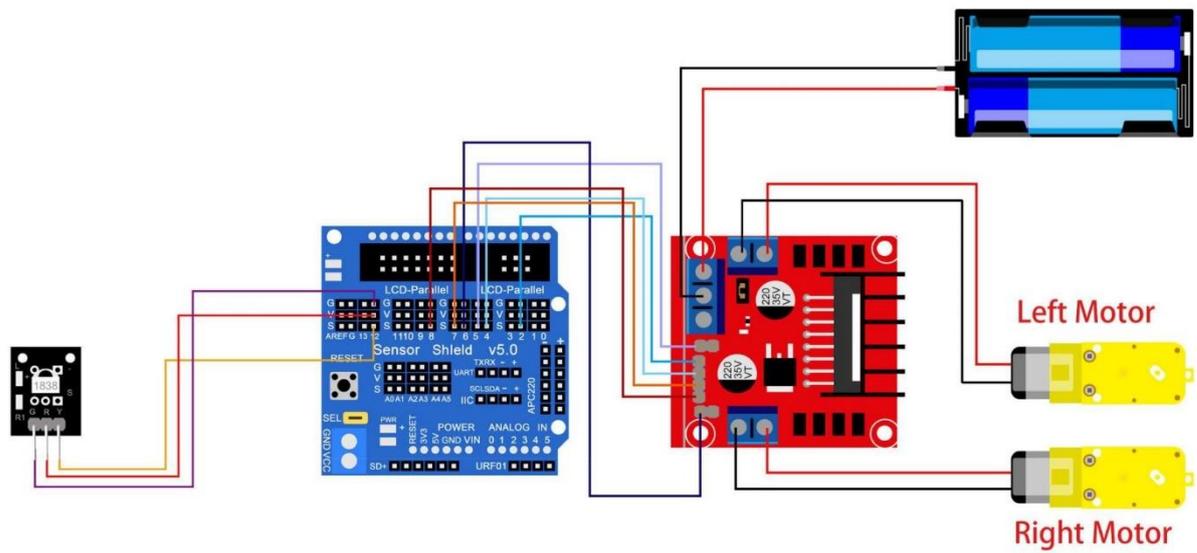
In diesem Kapitel steuern wir den Roboter mit dem IR-Empfänger und der IR-Fernbedienung. Die Fernbedienung sendet ein Infrarotsignal, welches vom IR-Empfänger aufgefangen und anschließend vom Arduino analysiert wird. Der Arduino entschlüsselt das Signal und setzt die entsprechenden Befehle für die Motoren um.



Schaltplan



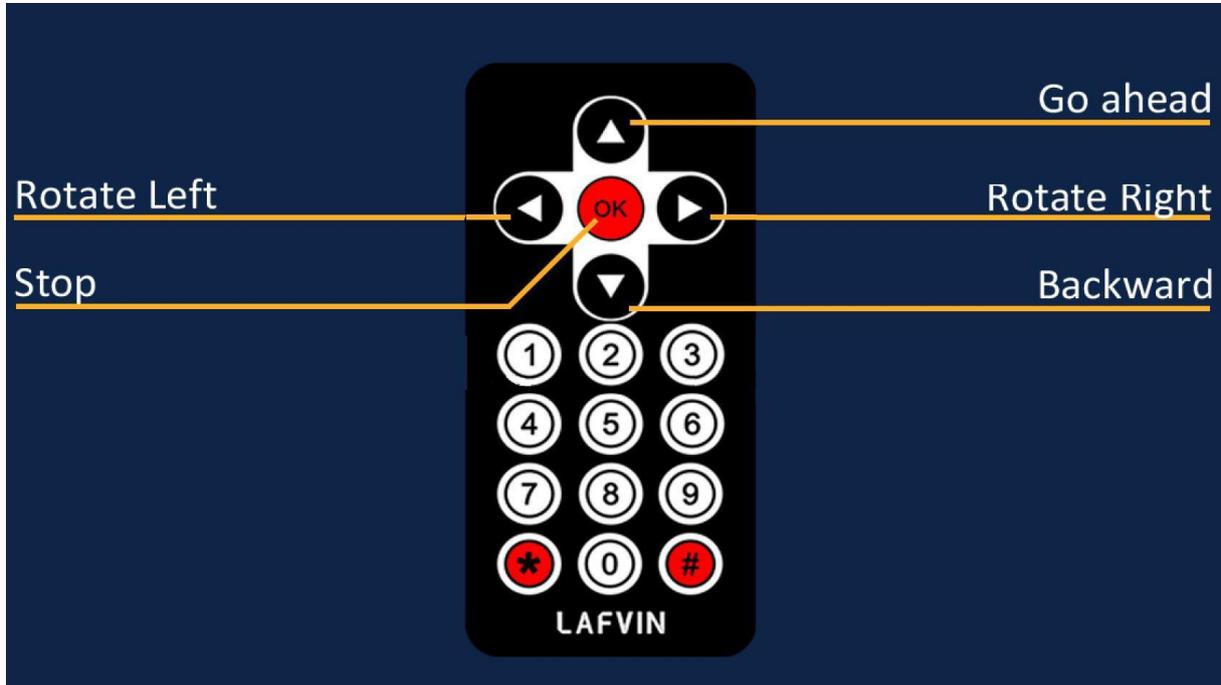
Anschlussplan



Code

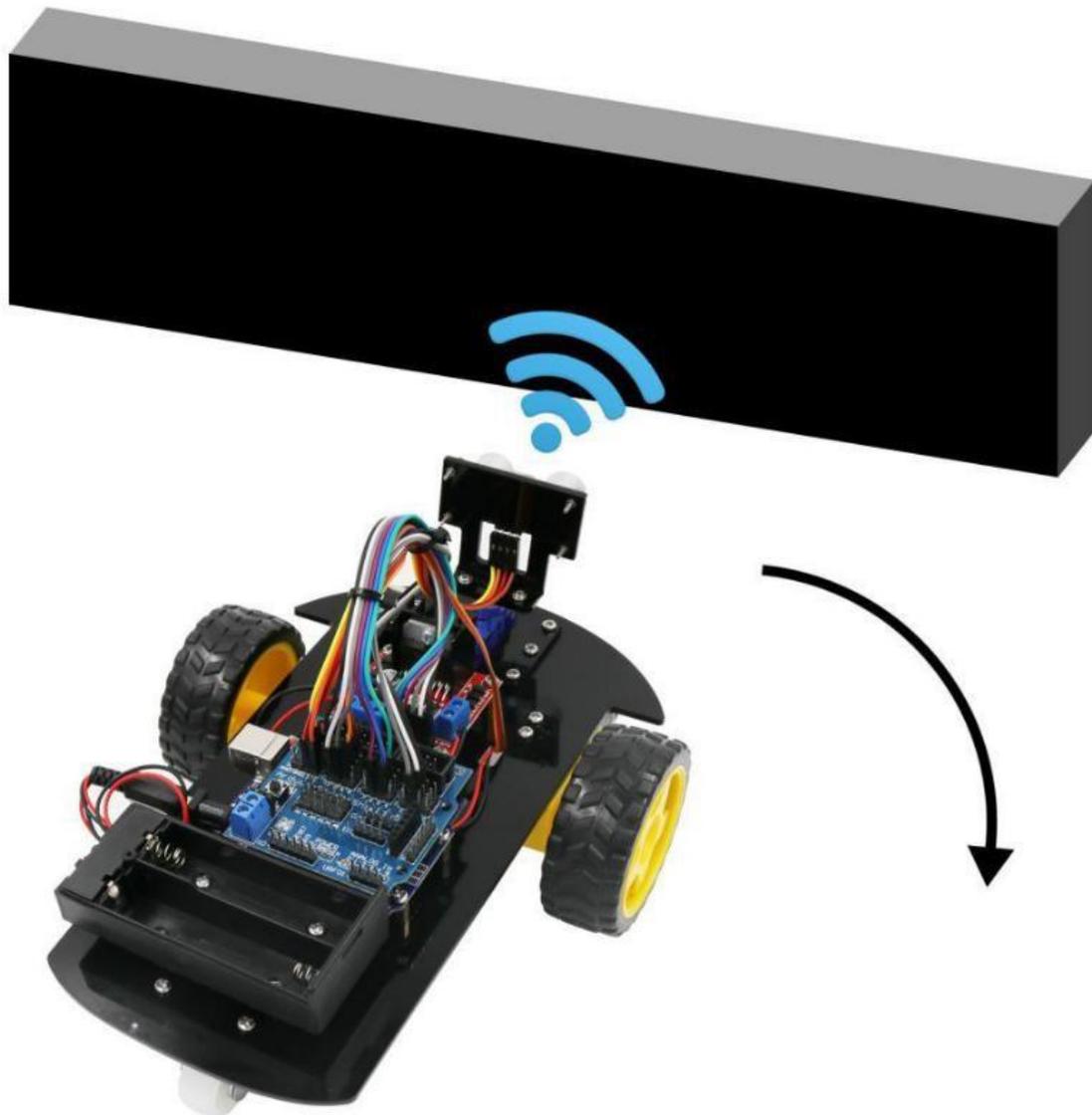
Nach dem Verkabeln können Sie den Code aus Lektion 9 hochladen. Sollten Sie Probleme beim Hochladen haben, können Sie in Lektion 3 nachschlagen.

Nun können Sie den Roboter mit der Fernbedienung steuern.



10. Auto mit Hinderniserkennung

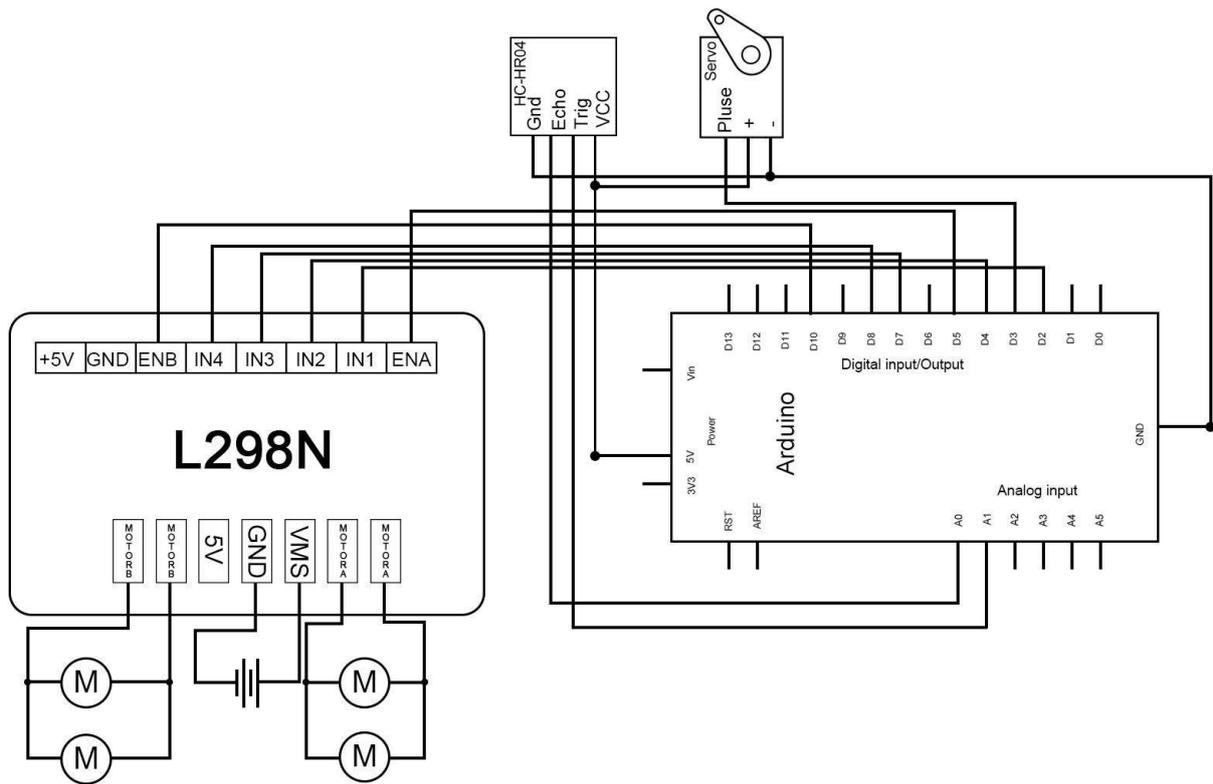
In diesem Kapitel wird das Roboter Auto komplett durch den Arduino gesteuert. Der Ultraschallsensor erkennt Hindernisse und kommuniziert mit dem Arduino UNO. Der Arduino verarbeitet das Signal und korrigiert die Position des Autos. Anschließend fährt das Auto weiter, bis es wieder ein Hindernis erkennt.



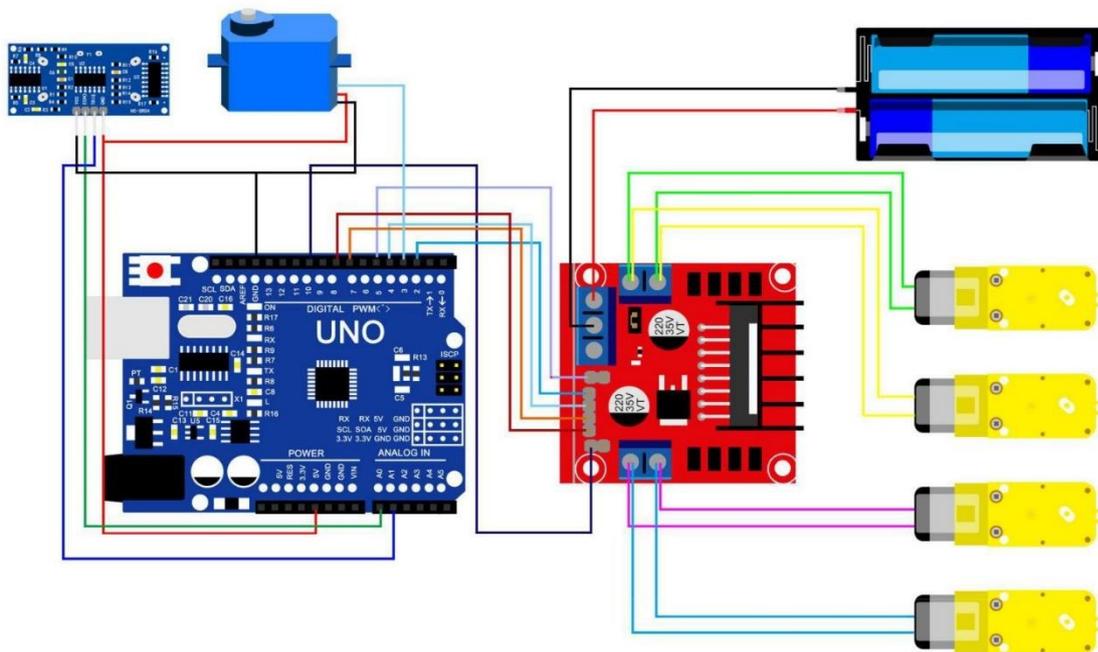
Funktionsweise

1. Das Ultraschallmodul erkennt die Entfernung zu Hindernissen und sendet ein Signal an den Arduino UNO.
2. Der Arduino UNO verarbeitet das Signal der Entfernung und vergleicht es mit den hinterlegten Daten.
3. Der Arduino UNO steuert die Motoren entsprechend an und korrigiert ggf. die Position des Autos.

Schaltplan



Anschlussplan



Code

Nach dem Verkabeln können Sie den Code aus Lektion 10 hochladen. Ohne die <HC-SR04> und <Servo> Bibliotheken funktioniert der Code nicht, wie Sie Bibliotheken installieren, können Sie in Lektion 2 nachlesen. Wenn Sie Probleme beim Hochladen haben, schlagen Sie in Lektion 3 nach.

Wenn alles funktioniert hat, fährt Ihr 2WD Roboter herum und weicht Hindernissen aus. Der Roboter erkennt, ob sich ein Hindernis vor ihm befindet und überprüft mit dem Servo ob er links oder rechts weiterfahren soll.

Hinweis: Die Hindernisse sollten >1cm hoch sein, damit der Ultraschallsensor sie erkennt.

Anweisungen

