Die Fernbedienung steuert verschiedene Lauflichter. Sie leuchten jeweils solange, bis eine beliebige andere Taste auf der Fernbedienung gedrückt wird.

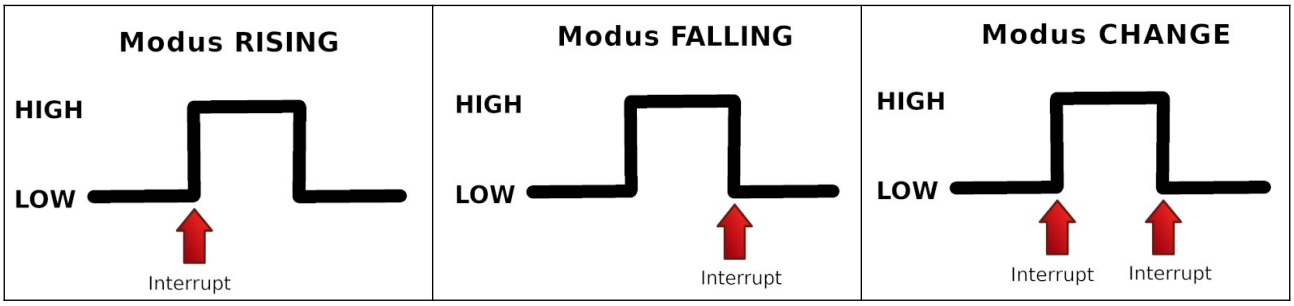
Taste 1: Lauflicht vor
Taste 2: Lauflicht hin und zurück
Taste 3: nacheinander blinkende LEDs

Dem Infrarot-Empfänger wird der Interrupt-Pin 2 zugeordnet (`attachInterrupt`). Wenn die zugeordnete Taste der Fernbedienung betätigt wird, startet sie das entsprechende Lauflicht, ein erneuter Druck auf eine beliebige Taste löst den Interrupt aus. Der normale Programmablauf wird unterbrochen und die festgelegte Methode (Interrupt-Service-Routine) wird ausgeführt. Anschließend wird das Programm normal fortgesetzt.

```
attachInterrupt(digitalPinToInterrupt(InterruptPin), Schalten, FALLING);
```

Es gibt verschiedene Ereignisse, die den Interrupt auslösen können:

RISING	der Interrupt wird ausgelöst wenn sich der Status von LOW zu HIGH ändert
FALLING	der Interrupt wird ausgelöst wenn sich der Status von HIGH zu LOW ändert
CHANGE	der Interrupt wird ausgelöst wenn sich der Status ändert



Der Empfänger muss zwingend am Pin 2 oder Pin 3 angeschlossen werden.

Zusätzlich muss die Variable, die eine Statusänderung anzeigt, als `volatile` definiert werden. Variable werden im RAM und temporär im Speicherregister gespeichert, bearbeitet oder verändert.

Es kann vorkommen, dass der Zustand der Variablen im Flashspeicher und im SRAM durch eine neue Wertzuweisung für kurze Zeit nicht übereinstimmen.
Das Schlüsselwort `volatile` weist das Programm an, die Variable immer aus dem Flashspeicher und nicht vom SRAM zu laden. Damit wird garantiert, dass der jeweils aktuelle Wert geladen wird.

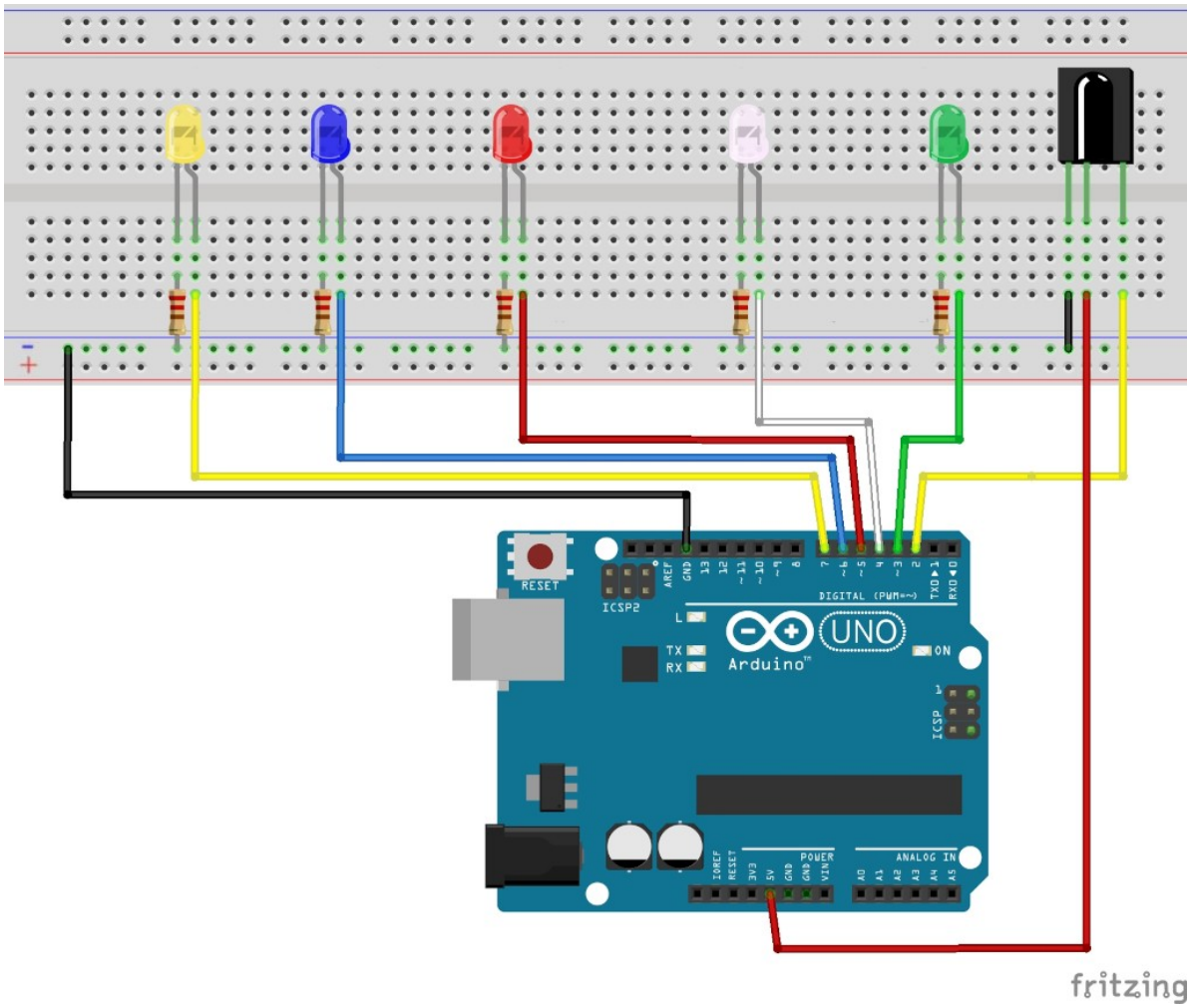
Außerdem sind bei der Programmierung einer Interrupt-Methode einige Regeln zu beachten:

- Der Programmteil muss so kurz wie möglich sein.
- Verwende kein `delay()`.
- Benutze auch kein `Serial.print()`.

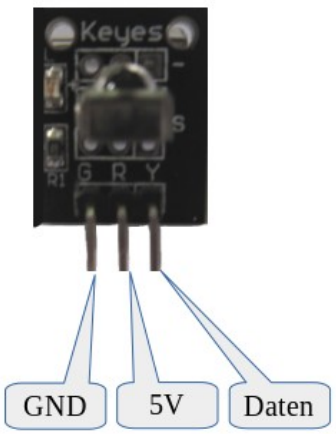
Benötigte Bauteile

- 5 LEDs
- 3 Widerstände 220 Ω (gelb, rot und grüne LEDs)
- 2 Widerstände 100 Ω (blaue und weiße LEDs)
- Fernbedienung
- Leitungsdrähte

Baue die Schaltung auf.



Achte auf die Pinbelegung der Infrarotempfänger.



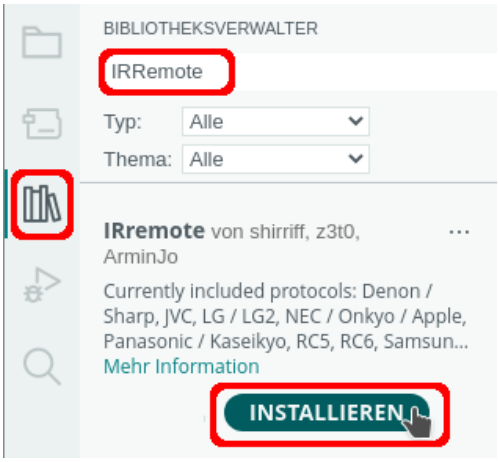
Keyes-Empfänger

VS1838B

Achte darauf, dass die Batterie richtig eingelegt wurde. Der Minus-Pol zeigt nach oben.

Benötigte Bibliothek:

Sketch → Bibliothek einbinden → Bibliotheken verwalten



Suche die Bibliothek IRremote ...

… und klicke auf installieren

Die Fernbedienung sendet beim Druck auf die Tasten einen Zahlencode.

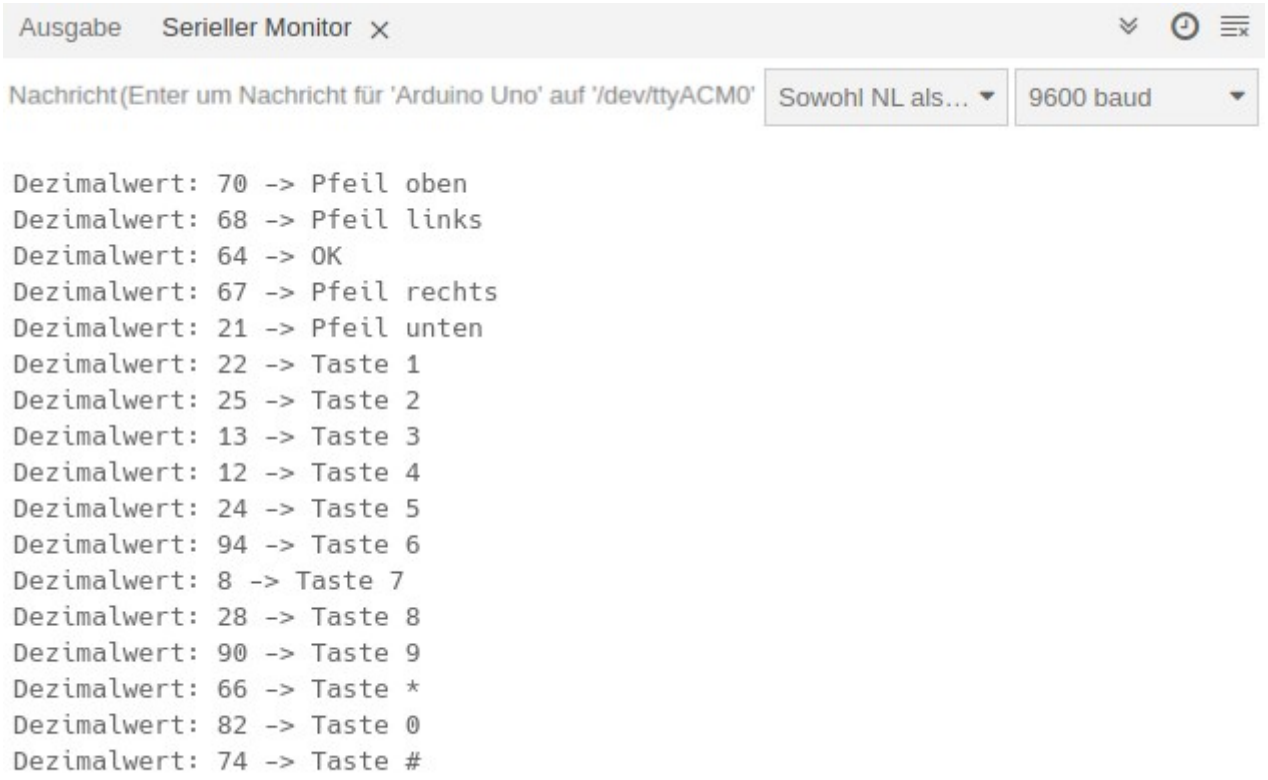
Die Tastencodes beziehen sich auf die **Keyes-Fernbedienung**. Die Tastencodes anderer Fernbedienungen kannst du mit Hilfe des Seriellen Monitors herausfinden.

Die Tastencodes der Keyes Fernbedienung (hexadezimal und dezimal).

oben	links	rechts	unten	OK	1	2	3	4
0x46 70	0x44 68	0x43 67	0x15 21	0x40 64	0x16 22	0x19 25	0xD 13	0xC 12
5	6	7	8	9	*	0	#	
0x18 24	0x5E 94	0x8 8	0x1C 28	0x5A 90	0x42 66	0x52 82	0x4A 74	

Die Tastencodes kannst du mit folgendem Programm herausfinden. Sie werden im Seriellen Monitor angezeigt.

```
// benötigte Bibliothek einbinden
#include "IRremote.hpp"
// der Pin, an dem der Infrarot-Empfänger angeschlossen ist
int EmpfaengerPin = 2;
void setup()
{
  // Seriellen Monitor starten
  Serial.begin(9600);
  // Infrarot-Empfänger starten
  IrReceiver.begin(EmpfaengerPin);
}
void loop()
{
  // decode() -> Daten lesen
  if (IrReceiver.decode())
  {
    // kurzes delay, damit nur ein Tastendruck gelesen wird
    delay(200);
    // resume -> nächsten Wert lesen
    IrReceiver.resume();
    /*
      der Empfänger empfängt zwischendurch Signale,
      die nicht ausgewertet werden können
      es sollen deshalb nur die korrekt erkannten Tasten ausgewertet werden
      die Dezimalwerte der korrekten erkannten Tasten liegen zwischen > 0 und < 95
      es wird abgefragt, ob das empfangene Kommando decodedIRData.command
      zwischen 0 und (&&) 95 liegt
    */
    if (IrReceiver.decodedIRData.command > 0 && IrReceiver.decodedIRData.command < 95)
    {
      Serial.print("Dezimalwert: ");
      // IrReceiver.decodedIRData.command = Wert der gedrückten Taste
      Serial.print(IrReceiver.decodedIRData.command);
      Serial.print(" -> ");
      // Werte abfragen und anzeigen
      if (IrReceiver.decodedIRData.command == 22) Serial.println("Taste 1");
      if (IrReceiver.decodedIRData.command == 25) Serial.println("Taste 2");
      if (IrReceiver.decodedIRData.command == 13) Serial.println("Taste 3");
      if (IrReceiver.decodedIRData.command == 12) Serial.println("Taste 4");
      if (IrReceiver.decodedIRData.command == 24) Serial.println("Taste 5");
      if (IrReceiver.decodedIRData.command == 94) Serial.println("Taste 6");
      if (IrReceiver.decodedIRData.command == 8) Serial.println("Taste 7");
      if (IrReceiver.decodedIRData.command == 28) Serial.println("Taste 8");
      if (IrReceiver.decodedIRData.command == 90) Serial.println("Taste 9");
      if (IrReceiver.decodedIRData.command == 82) Serial.println("Taste 0");
      if (IrReceiver.decodedIRData.command == 66) Serial.println("Taste *");
      if (IrReceiver.decodedIRData.command == 74) Serial.println("Taste #");
      if (IrReceiver.decodedIRData.command == 68) Serial.println("Pfeil links");
      if (IrReceiver.decodedIRData.command == 67) Serial.println("Pfeil rechts");
      if (IrReceiver.decodedIRData.command == 70) Serial.println("Pfeil oben");
      if (IrReceiver.decodedIRData.command == 21) Serial.println("Pfeil unten");
      if (IrReceiver.decodedIRData.command == 64) Serial.println("OK");
    }
  }
}
```



Binde die benötigte Bibliothek ein und definiere die Variablen.

```
#include "IRremote.hpp"

// Pin für den Auslöser des Interrupts
int InterruptPin = 2;

// Array mit 5 Elementen und den zugehörigen Ports
int LED[5] = {3, 4, 5, 6, 7};

// Anzahl der LEDs feststellen
int LEDMax = sizeof(LED) / sizeof(LED[0]);

// Variable im Flash-Speicher ablegen
volatile bool Status = true;

int Leuchtdauer = 100;
```

Der setup-Teil. Beachte die Kommentare

```
void setup()
{
  Serial.begin(9600);

  // auf serielle Verbindung warten
  while (!Serial) {;}
```

```
  // Zufallsgenerator starten
  randomSeed(analogRead(0));

  // Empfänger starten

  IrReceiver.begin(InterruptPin);

  for (int i = 0; i < LEDMax; i++)
  {
    pinMode(LED[i], OUTPUT);
  }

  // Funktion Schalten() dem Interrupt-Pin zuordnen
  attachInterrupt(digitalPinToInterrupt(InterruptPin), Schalten, FALLING);
}
```

Die Methoden für die verschiedenen Lauflichter.

```
void Schalten()
{
  Status = false;
}

void LEDBlinken(int LEDNummer, int Anzahl)
{
  for (int i = 0; i <= Anzahl; i ++)
  {
    digitalWrite(LEDNummer, HIGH);
    delay(Leuchtdauer);
    digitalWrite(LEDNummer, LOW);
    delay(Leuchtdauer);
  }
}

void LauflichtHin()
{
  for (int i = 0; i < LEDMax; i ++)
  {
    digitalWrite(LED[i], HIGH);
    delay(Leuchtdauer);
    digitalWrite(LED[i], LOW);
  }
}
```

```
void LauflichtHinUndHer()
{
  for (int i = 0; i < LEDMax; i++)
  {
    digitalWrite(LED[i], HIGH);
    delay(Leuchtdauer);
    digitalWrite(LED[i], LOW);
  }
  // und wieder zurück
  for (int i = LEDMax - 1; i >= 0; i--)
  {
    digitalWrite(LED[i], HIGH);
    delay(Leuchtdauer);
    digitalWrite(LED[i], LOW);
  }
}

void LauflichtMitBlinken()
{
  for (int i = 0; i < LEDMax; i++)
  {
    int Anzahl = random(1, 5);
    /* aktuelle LED i einschalten
      → Funktion LEDBlinken aufrufen
    */
    LEDBlinken(LED[i], Anzahl);
  }
}
```

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  // Daten lesen
  if (IrReceiver.decodedIRData.address == 0)
  {
    if (IrReceiver.decode())
    {
      delay(200);
      Status = true;
      /*
        solange der Status true ist, wird die jeweilige while-Schleife
        ausgeführt, ein weiterer Druck auf eine Taste der Fernbedienung löst
        den Interrupt aus
        -> Status wird zu false, die while-Schleife wird nicht erneut
        ausgeführt
      */
```

```
      switch (IrReceiver.decodedIRData.command)
      {
        // Taste 1: Lauflicht vor
        case 22:
          while (Status) LauflichtHin();
          break;

        // Taste 2:  LEDs leuchten vor und zurück
        case 25:
          while (Status) LauflichtHinUndHer();
          break;

        // Taste 3: LEDs blinken nacheinander
        case 13:
          while (Status) LauflichtMitBlinken();
          break;
      }
      // nächsten Wert lesen
      IrReceiver.resume();
    }
  }
}
```

Jetzt fehlt noch die durch den Interrupt ausgelöste Funktion Schalten(). Sie ändert den Zustand von Status auf false.

```
void Schalten()
{
  // verhindern, dass durch ein evtl. gesendetes Signal
  // das Lauflicht gestoppt wird
  if (IrReceiver.decodedIRData.address == 0)
  {
    if (IrReceiver.decode())
    {
     if (IrReceiver.decodedIRData.command != 0) Status = false;
    }
  }
}
```