

Aufbau eines Sketches

```
Blinken | Arduino 1.8.8

Blinken

1 /*
2  Blinken - Das 'Hello World' des Arduinos
3  Einfaches Blinken der eingebauten LED im Sekundentakt
4
5  Version 1.0
6  23.12.2018 Der Hobbyelektroniker
7  Der Code ist Public Domain und kann ohne Einschränkungen frei verwendet werden
8
9  */
10
11 // ==> Einbindung von externen Programmodulen (Bibliothek, Daten usw.)
12
13 // ==> Definition von Konstanten und Makros
14 const int eingebauteLed_Pin = LED_BUILTIN; // Das ist beim Arduino UNO der Pin 13
15
16 // ==> Globale Variablen
17
18 // ==> Eigene Funktionen
19
20
21 // ==> setup() und loop() sind in jedem Programm obligatorisch!
22
23 // Dieser Code wird beim Start genau ein Mal ausgeführt
24 void setup() {
25   // Pins arbeiten standardmässig als Eingänge
26   // Deshalb müssen die Ausgänge hier definiert werden
27   pinMode(eingebauteLed_Pin, OUTPUT);
28 }
29
30 // Das ist das eigentliche Programm. Es läuft in einer endlosen Schleife
31 // und kann nicht unterbrochen werden.
32 void loop() {
33   digitalWrite(eingebauteLed_Pin, HIGH); // LED einschalten (5V an Ausgang legen)
34   delay(1000); // 1000 Millisekunden (= 1 Sekunde) warten
35   digitalWrite(eingebauteLed_Pin, LOW); // LED ausschalten (0V an Ausgang legen)
36   delay(1000); // 1000 Millisekunden (= 1 Sekunde) warten
37 }

Kompilieren abgeschlossen.
Der Sketch verwendet 930 Bytes (2%) des Programmspeicherplatzes. Das Maximum sind 32256 Bytes
Globale Variablen verwenden 9 Bytes (0%) des dynamischen Speichers, 2039 Bytes für lokale Var

2 Arduino/Genuino Uno auf /dev/cu.usbmodem141101
```

Kommentare sind wichtig, damit wir später noch wissen, was wir wann und warum gemacht haben. Ausserdem könnte es sein, dass irgendwann mal jemand anderes mit unserem Code arbeiten möchte.

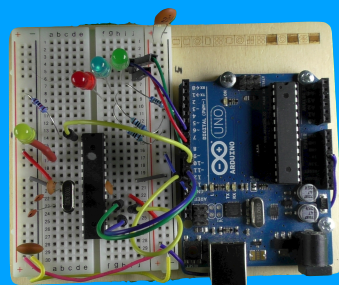
Wenn unsere Programme grösser werden, arbeiten wir mit externen Modulen.

Wenn immer möglich, arbeiten wir mit verständlichen Bezeichnungen anstelle von Nummern.

Daten lassen sich in Variablen speichern. Eigene Funktionen helfen uns, das Programm übersichtlich zu halten. Das brauchen wir jetzt noch nicht, später wird das aber sehr wichtig!

Dieser Programmteil wird nur beim Starten einmal ausgeführt. Darin werden alle Vorbereitungen getroffen, die für den Programmablauf notwendig sind. Diese Funktion ist obligatorisch und kann nicht weggelassen werden!

Hier läuft das eigentliche Programm ab. Diesen Teil sollte man möglichst schlank halten und die eigentliche Funktionalität in selbstgeschriebene Funktionen verpacken. Momentan fehlt uns dazu das notwendige Wissen, deshalb steht hier das ganze Programm.



Der schnelle Einstieg in Arduino & Co.
Programmierung

