|  |  |
| --- | --- |
| **Arbeitsblattsammlung GDC Ostwürttemberg** | |
| Modul | 1.3 |
| Thema | Programmieren |
| Zielalter | Ab 12 |
| Methodik | Informationstext, Programm |



**„Blink“ – die blinkende Leuchtdiode**

Ein Arduino-Programm (Sketch) besteht grundsätzlich aus zwei Teilen:

1. Der erste Teil, das **setup***,* wird einmalig beim Start des Arduino-Mikrocontrollers ausgeführt. In diesem Teil werden Vorbereitungen für den restlichen Programmablauf getroffen. Hier wird beispielsweise festgelegt, welche Eingänge oder Ausgänge, die sogenannten Pins, des Mikrocontrollers genutzt werden sollen. In der Programmiersprache des Arduino wird dies so dargestellt:

*void setup()*

In der vorliegenden Aufgabe wird der PIN2 als Ausgang festgelegt. In der Programmiersprache des Arduino wird dies so dargestellt:

*pinMode(2, OUTPUT)*

1. Der zweite Teil, der so genannte **loop-Teil**, wird nach einmaliger Ausführung des **setup***-*Teils kontinuierlich wiederholend Zeile für Zeile abgearbeitet. In der Programmiersprache des Arduino wird dies so dargestellt:

*void loop()*

Für die ständige Wiederholung des Blinksignals wird folgendes benötigt:

1. der Ausgang eine Spannung erhält (High-Signal an PIN2),
2. das Signal eine Weile geschaltet bleibt (einige Millisekunden warten)
3. am Ausgang dann die Spannung abgeschaltet wird (LOW-Signal an PIN2)
4. der Zustand eine Weile bestehen bleibt (einige Millisekunden warten).

In der Programmiersprache des Arduino wird dies so dargestellt:

*digitalWrite(2, HIGH);*

*delay(5000);*

*digitalWrite(2, LOW);*

*delay(5000);*

Zum Schluss müssen geschweifte Klammern um die gesamten Programmteile gesetzt werden. Diese Klammern geben dem Programmleser wichtige Hinweise zu dem Programmablauf. Das ist vergleichbar mit den Klammern in Mathematik. Das gesamte Programm sieht nun wie folgt aus:

*void setup() {*

*pinMode(LED\_BUILTIN, OUTPUT);*

*}*

*void loop() {*

*digitalWrite(LED\_BUILTIN, HIGH);*

*delay(1000);*

*digitalWrite(LED\_BUILTIN, LOW);*

*delay(1000);*

*}*

Nun lässt sich das Programm testen, in dem es in den Arduino übertragen und ausgeführt wird.

Praxistipp: Wenn das Programm beim Kompilieren nicht läuft, liegt dies oft daran, dass nach Befehlen das Semikolon fehlt oder geöffnete geschweifte Klammern nicht wieder geschlossen wurden.