Wie beginne ich mit dem Arduino?

Über Anfangsschwierigkeiten von Lernenden und einen einfachen Einstieg in die textuelle Programmierung

Alexander Pusch

In diesem Beitrag werden mögliche Anfangsschwierigkeiten von unerfahrenen Lernenden beim Erlernen des Umgangs mit Mikrocontrollern wie dem Arduino beschrieben. Darauf aufbauend wird ein vielfach mit Studierenden, Lehrkräften und Schüler/inne/n erprobtes Konzept zum Einstieg in das textuelle Programmieren eines Arduino UNO und einer aufsteckbaren LED vorgestellt. Bei diesem Konzept auf Basis des Lernens aus Lösungsbeispielen liegt der Fokus auf dem systematischen Vermeiden des Auftretens typischer Anfangsschwierigkeiten bei gleichzeitig selbstständiger und binnendifferenzierter Erarbeitung grundlegender Programmfunktionen anhand bereitgestellter Materialien.

1 *Welche Anfangsschwierigkeiten und Fehlerquellen gibt es?*

Wie beginnt man mit dem Mikrocontroller Arduino? Wie bringt man die Verwendung Schüler/inne/n oder Studierenden bei? Es gibt viele Webseiten (z.B. https://create.arduino.cc/projecthub), fachdidaktische Artikel (z.B. in den Themenheften der *Plus Lucis* und der *Naturwissenschaften im Unterricht Physik*), Videotutorials und Bücher (z.B. das umfassende Buch zur Sensorik von Karvinen, Karvinen & Valtokari, 2014), die Zugangsmöglichkeiten und Anwendungsbeispiele für Sensorik mit Mikrocontrollern wie dem Arduino vorstellen. Oftmals werden dabei die Schaltungen anhand eines Fotos, eines Schaltplanes oder einer ikonischen Ansicht (vgl. Abb.1) auf einem Steckbrett (*Breadboard*) dargestellt. Nach dem Nachbau der Schaltung wird ein fertiger Programmcode auf den Mikrocontroller geladen, dessen zentrale Bestandteile erläutert werden.

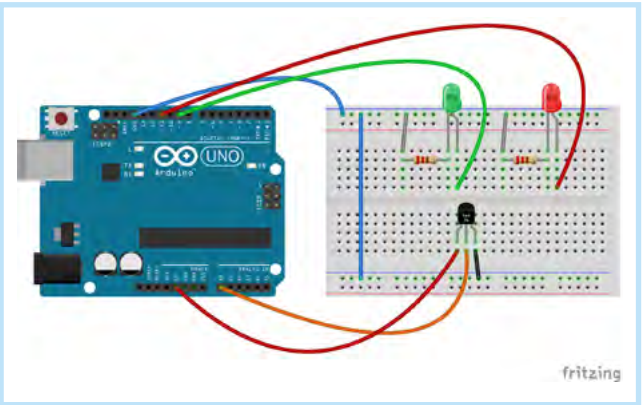

Abb. 1. Exemplarische Schaltung eines Temperatursensors (TMP36) sowie zwei LEDs mit Vorwiderstand an einem Arduino UNO. Schaltpläne dieser Art sind z.B. mit https://fritzing.org/ oder https://www.tinkercad.com/ erstellbar.

Diese Darbietungsweise funktioniert gut für erfahrene Personen, da sie bereits ein großes Erfahrungswissen mit der Thematik gesammelt haben. Noviz/inn/en können bei der beschriebenen Vorgehensweise aber mit vielen Fehlerquellen und individuell verschiedenen Schwierigkeiten konfrontiert sein (Tab. 1, s.a. Pusch, 2019). Gerade aber der allererste Anfang mit der Thematik Mikrocontroller ist schwer und sorgt – wenn er nicht gelingt – schnell für Frustration.

Betrachten wir ein Beispiel für eine sehr einfache Wetterstation: ein Temperatursensor sowie zwei LEDs samt Vorwiderstand, die abhängig von einer bestimmten Temperatur angesteuert werden, sind auf einem Steckbrett verschaltet und mit dem Arduino verbunden (Abb.1).

(1) Um einen Programmcode (*sketch*) auf einen Arduino hochzuladen, muss ein (zunächst unbekanntes) Computerprogramm (z.B. die Arduino-IDE) installiert und bedient werden. In diesem Programm müssen Parameter, wie das verwendete Arduinoboard und der Anschluss des Arduinos (*COM-Port*) eingestellt werden, da sonst die Übertragung des Programmcodes nicht funktioniert.

(2) Da die Funktionsweise von Steckbrettern für Noviz/inn/en nicht intuitiv ist, können sich schnell Fehler durch den falschen Nachbau der Schaltung ergeben. Es ist nicht am Steckbrett ersichtlich, ob die Zeilen oder Spalten leitend miteinander verbunden sind. Auch nicht richtig eingesteckte Verbindungen oder Kurzschlüsse können Probleme verursachen.

(3) Oft fehlt auch das Wissen über die technische Funktionsweise der Bauteile, wie bspw. die der Sperr- und Durchlassrichtung einer LED. Wird die LED falsch herum eingesetzt, wird sie nicht leuchten. Der Temperatursensor TMP36 liefert eine linear von der temperaturabhängige Ausgangsspannung, wenn er an die Versorgungsspannung angeschlossen wird. Welche beiden der drei Beine aber für die Spannungsversorgung und welches für das Messen der Ausgangsspannung verwendet wird, ist nicht auf dem Bauteil, sondern in einem separaten Datenblatt verzeichnet.

(2) + (3) Eine falsche Verschaltung und unpassende Betriebsspannung bzw. fehlender oder unpassender Vorwiderstand können die Bauteile zerstören. Einen möglichen Defekt von Bauteilen können Noviz/inn/en nicht systematisch ermitteln. In seltenen Fällen sind Bauteile sogar defekt, obwohl sie „frisch" aus der Verpackung sind.

(4) Auch beim Vornehmen eigener Anpassungen im Text des Programmcodes können wegen unzureichender Kenntnis von Programmierbefehlen und Syntax weitere, individuell verschiedene Schwierigkeiten liegen. Selbst ganz triviale Fehler wie ein

MNU-Journal – Ausgabe 02.2023 – ISSN 0025-5866 – © Verlag Klaus Seeberger, Neuss

Bereich	Mögliche Fehlerquellen und Schwierigkeiten
(1) Computerprogramm	• Ein (neues) Computerprogramm (hier: Arduino-IDE) installieren und bedienen (Funktionsweise, Einstellungen, Ablaufschritte etc.).
(2) Verschaltung	• Eine elektronische Schaltung auf einem Steckbrett (*Breadboard*) umsetzen und Fehler sowie Wackelkontakte und Kurzschlüsse erkennen und systematisch ermitteln. • Elementare Schaltungstechnik verstehen und anwenden (Reihen- und Parallelschaltung, Spannungsteiler, Vorwiderstand, LED (Kathode & Anode)).
(3) Funktionsweise der Komponenten	• Technische Funktionsweise von Sensoren und Aktuatoren (z.B. Polarität, I2C-Bus (SCL/SDA), Entprellen eines Tasters) verstehen und in der Schaltung umsetzen. • Die physikalische Funktionsweise von Sensoren und Aktuatoren (in Grundzügen) verstehen und korrekt für Messungen bzw. Experimente anwenden. • Defekte ermitteln
(4) Programmiersprache	• Eine Programmiersprache mit eigenen Begriffen, Groß- und Kleinschreibung, Regeln und Syntax verstehen, lernen und anwenden.

Tab. 1. Mögliche Fehlerquellen und Schwierigkeiten von Noviz/inn/en bei der Verwendung von Mikrocontrollern wie dem Arduino.

fehlendes Semikolon als Befehlsabschluss werden manchmal erst nach langer Suche gefunden.
Eine Alternative zu textbasierten Programmieren ist die Verwendung einer Block-strukturierten Programmiersprache (z.B. http://blog.ardublock.com oder https://developers.google.com/blockly/). Diese weist ein eigenes didaktisches Potential auf. Sie stößt aber bei vielen Projekten schnell an die Grenzen und es muss dann auf eine textbasierte Programmiersprache gewechselt werden.

Bei schulischen und universitären Lerngruppen ist erfahrungsgemäß mit dem Auftreten der oben genannten Anfangsschwierigkeiten zu rechnen. Der Zeitbedarf für eine individuelle Unterstützung und Fehlersuche durch die Lehrperson ist dabei selbst bei kleinen Lerngruppen (z.B. 12 Personen) schon sehr hoch. Das Auftreten unverständlicher und unvorhersehbarer Fehler und deren zeitaufwendige Behebung kann bei allen Beteiligten schnell Frustration erzeugen. Daher sollte der erste Einstieg möglichst reibungslos gestaltet werden, indem zu Beginn möglichst viele potenzielle Fehlerquellen vermieden werden, um sukzessive grundlegendes Wissen in den genannten Bereichen aufzubauen. Hierdurch werden Selbstsicherheit und Handlungswissen aufgebaut.

2 *Ein möglichst „reibungsfreier" Einstieg (Best-Practice-Beispiel)*

Nachfolgend wird ein vielfach erprobtes Best-Practice-Beispiel für einen möglichst reibungsfreien Einstieg in die Arbeit mit dem Mikrocontroller Arduino vorgestellt. Hierzu werden ein Arduino UNO samt einer aufgesteckten LED mit angelötetem 220 oder 330 Ohm Vorwiderstand (Abb. 2), ein *Basisprogramm* (Abb. 3) und ein *Handout* (Abb. 4.) benötigt (Download unter https://physikkommunizieren.de/arduino/einstieg/). Die Schaltung kann bereits zusammengesteckt als „fertige" Schaltung herausgegeben werden oder die Lernenden stecken die LED samt angelöteten Vorwiderstand, wie im Handout beschrieben, selber in den Arduino ein.

Das Basisprogramm nimmt die Schaltung rudimentär in Betrieb. Der Programmcode (*sketch*) ist ausführlich an jeder Programmzeile kommentiert (Abb. 3). Er erläutert so die grundlegenden Programmbefehle für die Lernenden. Am Ende des Basisprogramms befinden sich zugehörige *Lernaufgaben* sowie auch *Hilfestellungen samt Lösungsvorschlägen*. Durch die Bearbeitung der Lernaufgaben wird der Programmcode schrittweise um verschiedene Befehle und Funktionen erweitert und es werden einfache Programmieraufgaben und Experimente zum zeitlichen Auflösungsvermögen des Auges sowie zur Codierung und Übertragung von Informationen mittels Morsealphabet angeleitet. Das zusätzliche *Handout* (Abb.4) liefert eine kompakte Schritt-für-Schritt-Anleitung zur technischen Inbetriebnahme des Arduinos mit der Arduino-IDE.

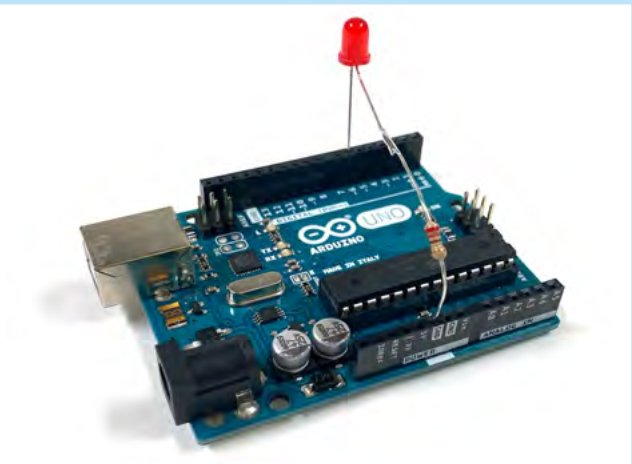

Abb. 2. Die Schaltung ist sehr einfach und besteht aus einem Arduino UNO sowie einer LED samt angelöteten 220 Ohm Vorwiderstand.

Das Lernen anhand fertiger Schaltungen und vor allem mit den kommentierten Basisprogrammen basiert auf dem etablierten fachdidaktischen Prinzip des Lernens aus Lösungsbeispielen (Worked-out-examples, z.B. Renkl, Schworm & Hilbert, 2004). Durch die Bereitstellung aller relevanter Hintergrundinforma-

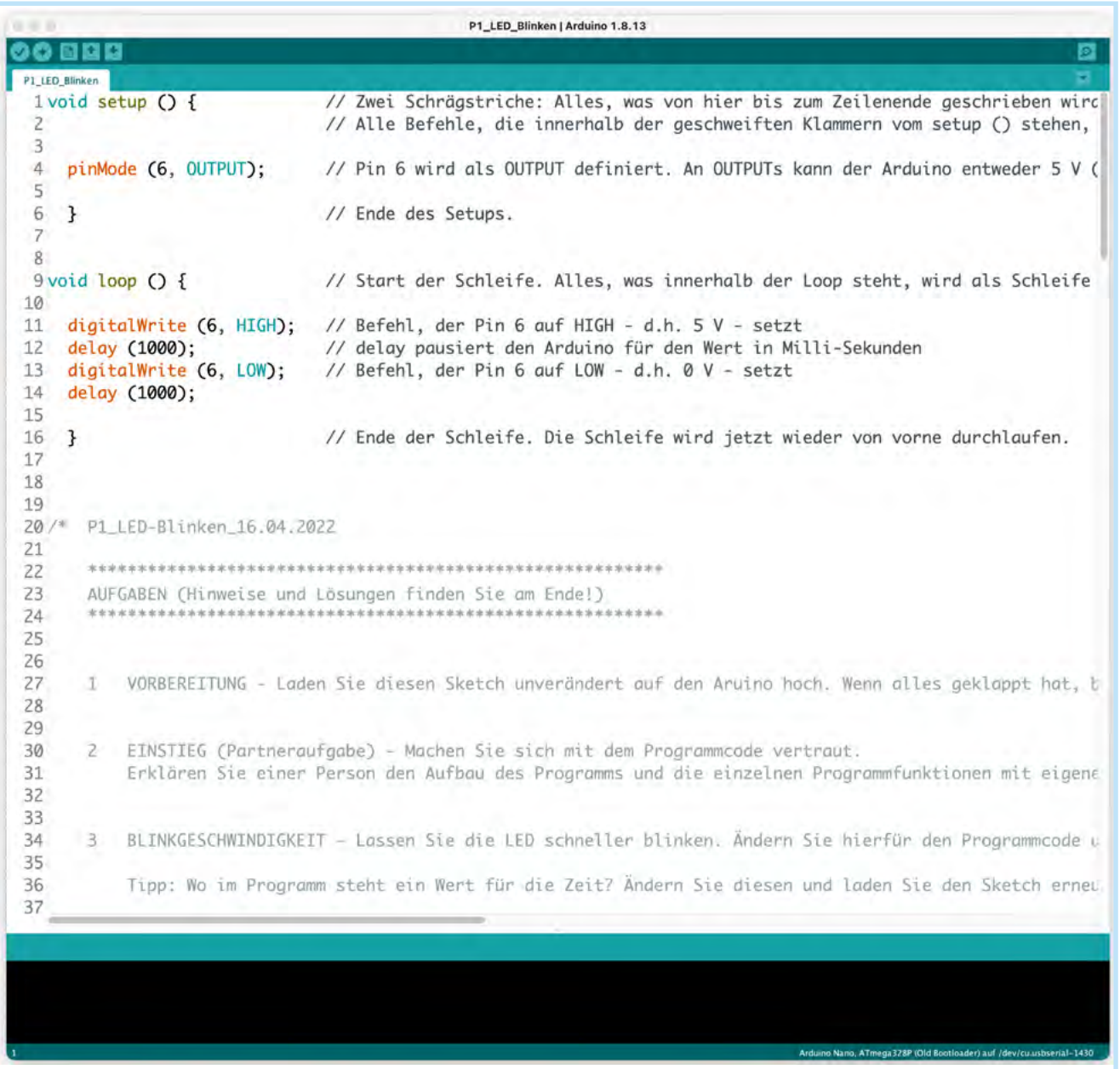

Abb. 3. Das ausführlich kommentierte Basisprogramm samt anschließenden Lernaufgaben und Lösungshinweisen.

tionen kann die wahrgenommene Schwierigkeit und Belastung (*extraneous load*) bei Noviz/inn/en gesenkt werden, sodass mehr kognitive Kapazität für das Erlernen der Inhalte zur Verfügung steht (Renkl, Gruber, Weber, Lerche & Schweizer, 2003). Die Hilfestellungen und Lösungsansätze im kommentierten Basisprogramm greifen die Idee der gestuften Hilfen (z.B. Wodzinski, 2013) auf. Schwächere Lernende können durch die umfangreichen Hilfen die wesentlichen Grundlagen erlernen. Leistungsstärkere Lernende finden im kommentierten Basisprogramm auch vertiefende, offene Aufgabenstellungen.
Die Arbeit mit der einfachen „fertigen" Schaltung und dem kommentierten Basisprogramm verringert das Auftreten vieler der im vorherigen Abschnitt genannten möglichen Fehlerquellen (Tab. 1) und ermöglichen in Kombination mit dem Handout einen weniger schwierigen Einstieg und damit ein einfaches Erlernen der Grundlagen von Arduino.

Die Lernenden sammeln Erfahrung im Umgang mit der Arduino-IDE und der Programmiersprache. Sie lernen die Verwendung einiger wichtiger Befehle, wie dem Schalten der digitalen Ausgänge, Nutzung von Variablen, Delays, Funktionen und Schleifen. Auftretende Fehler und deren Suche beschränken sich auf die Arduino-IDE und die Programmiersprache, sodass hier entsprechendes Wissen in der Fehlersuche und Behebung erworben wird.

3 Nachfolgende Inhalte

Im Anschluss an diesen Einstieg kann die Beschäftigung mit dem Arduino weiter "geöffnet" werden. Es bietet sich an, als nächstes die Schaltung der gesteckten LED samt Vorwiderstand auf einem Steckbrett umzusetzen. Der Betrieb und die Programmierung des Arduinos mittels Arduino-IDE sowie grundlegende

LED BLINKEN MIT ARDUINO

Institut für Didaktik der Physik

MATERIAL

- Arduino UNO (vgl. Abb. 1)
- LED mit 220 Ω Vorwiderstand (vgl. Abb. 1)
- USB-Kabel (vgl. Abb. 1)
- Computer mit installierter Arduino-Software, z.B. der kostenlosen Arduino-IDE von https://www.arduino.cc/en/software
- Das Basisprogramm *„LED_Blinken.ino"*

Abbildung 1: Benötigt wird ein USB-Kabel, der Arduino UNO und eine LED samt angelötetem 220 Ω Vorwiderstand sowie ein Computer samt Arduino-Software (z.B. der kostenlosen Arduino-IDE von https://www.arduino.cc/en/software)

VERSCHALTUNG

Stecken Sie die LED mit Vorwiderstand in Digital-Pin 6 sowie GND. Die Seite mit dem Widerstand kommt in GND, die Seite mit der LED in PIN 6. Andersrum funktioniert die LED nicht. Schauen Sie die folgenden Abbildungen 2 und 3 an.

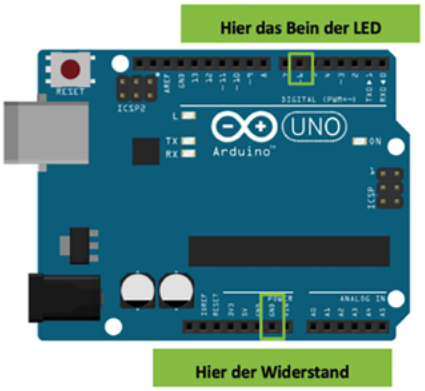

Abbildung 2: So wird die LED samt Vorwiderstand in den Arduino gesteckt.

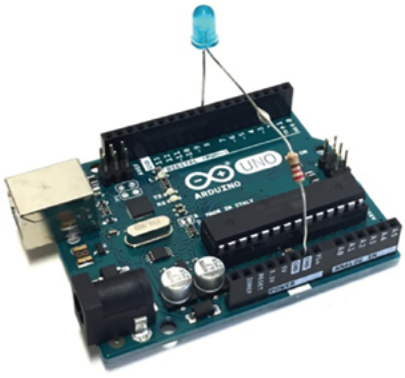

Abbildung 3: So sieht die fertige Schaltung aus.

PROGRAMM HOCHLADEN

- Verbinden Sie den Arduino mit Ihrem Computer per USB-Kabel. Der Arduino wird dann direkt mit Energie versorgt und leuchtet.
- Starten Sie das Programm „Arduino-IDE".
- Öffnen Sie die Datei „LED_Blinken.ino". Sie sehen das Programm im Fenster, es befindet sich aber noch nicht auf dem Arduino (Abb.4).
- Wählen Sie in der Menüleiste unter Werkzeuge das richtige Arduino-Board (hier „Arduino UNO") aus: → **Werkzeuge > Board > Arduino Uno**
- Stellen Sie anschließend den passenden COM-Port für die Kommunikation zwischen Computer und Arduino ein: → **Werkzeuge > Port**. In der Regel wird bei einem der Ports angezeigt, dass ein Arduino verbunden ist – manchmal aber auch nicht. Falls nicht, probieren Sie nach und nach die Ports durch.
- Übertragen Sie das Programm (den so genannten „Sketch") mittels Klick auf den Pfeil (Abb. 5) auf den Arduino. Alternativ geht das auch über das Menu: → **Sketch > Hochladen**. Nach dem erfolgreichem Upload wird das Programm automatisch ausgeführt. Wenn alles geklappt hat, blinkt die LED.
- Bearbeiten Sie nun die Lernaufgaben, die am Ende des Programmes stehen (Abb. 4). Weiter unten stehen auch Hilfestellungen.

Abbildung 4: Programmcode in der Arduino-IDE. Unter dem Programmcode stehen Lernaufgaben und Hilfen.

Abbildung 5: Bedienleiste in der Arduino-Software von arduino.cc. Mit einem Klick auf den Pfeil wird der Sketch auf den Arduino hochgeladen.

WAS TUN, WENN ES NICHT BLINKT?

Prüfen Sie zunächst Folgendes:

- **Ist die LED richtig herum angeschlossen?**
- **Ist die LED in GND und PIN6 gesteckt?**
- **Konnte das Programm hochgeladen werden?**
  → Falls nicht: ist das richtiges Board und der richtige USB-Port eingestellt?
- **Leuchten die LEDs auf dem Arduino?**
  → Falls nicht ist es ein Problem mit der Spannungsversorgung ?
  → anderes Kabel und anderen USB-Port probieren zuletzt: anderen Arduino probieren.

Abb. 4. Das Handout ist eine Schritt-für-Schritt-Anleitung zur Inbetriebnahme.

Programmbefehle für das Ansteuern einer LED sind nun bereits erprobt. Das bereits bekannte Basisprogramm kann auch für die erste Inbetriebnahme der Schaltung auf dem Steckbrett verwendet werden. Hierdurch kommt als weitere Fehlerquelle und mögliche Schwierigkeit nur die Verschaltung auf einem Steckbrett hinzu und fokussiert eine mögliche Fehlersuche auf eben die Verschaltung der Komponenten auf dem Steckbrett.
Daran anschließend kann mit weiteren LEDs und Vorwiderständen beispielsweise eine Ampelschaltung umgesetzt werden. Diese kann im nächsten Schritt mit einem Taster (oder einem anderen Sensor) gesteuert werden. Durch diese schrittweise „Öffnung" können weitere Erfahrungen mit den in Tabelle 1 genannten Bereichen gesammelt werden, die notwendig sind, um auch komplexere Projekte wie in Abbildung 1 erfolgreich bearbeiten zu können.

4 Fazit

Das in diesem Beitrag beschriebene Konzept stellt eine Möglichkeit zur Implementierung des Einstiegs in die Arbeit mit Mikrocontrollern wie dem Arduino in universitäre und schulische Lernprozesse dar. Es wird seit Jahren in universitären Lehrveranstaltungen für angehende Physiklehrkräfte an der WWU Münster und auch in verschiedenen Lehrerfortbildungen für Physik- und NaWi-Lehrkräfte erfolgreich durchgeführt. Typische Anfangsschwierigkeiten, welche Lernenden ohne Vorerfahrungen den Einstieg oft schwierig und wenig motivierend gestalten würden, werden so weitestgehend vermieden. Durch das Bereitstellen von selbstständig zu bearbeiten Lernmaterials können die Lernenden in ihrem eigenen Tempo und mit eigenen Schwerpunkten bearbeiten. Die Lehrperson kann für individuelle Fragen und Rückmeldungen zur Verfügung stehen und bei etwaigen Schwierigkeiten gezielte Hinweise auf die eingegrenzten Auslöser geben.
Von Studierenden wurde der hier beschriebene Einstieg mit der blinkenden LED erfolgreich im Rahmen von Praktika an Schulen getestet. Hierbei wurden Workbooks verwendet, um zusätzliche Informationen bereitzustellen, die Bearbeitung der Lernaufgaben zu strukturieren und die erarbeiteten Ergebnisse zu sichern.

Literatur

Karvinen, K., Karvinen, T. & Valtokari, V. (2014). *Sensoren – messen und experimentieren mit Arduino und Raspberry Pi*. Heidelberg: dpunkt.verlag.

Pusch, A. (2019). Arduino im Physikunterricht. *Physik Journal*, 18(5), 26–29.

Renkl, A., Gruber, H., Weber, S., Lerche, T. & Schweizer, K. (2003). Cognitive Load beim Lernen aus Lösungsbeispielen. *Zeitschrift für Pädagogische Psychologie*, 17, 93–101.

Renkl, A., Schworm, S. & Hilbert, T. S. (2004). Lernen aus Lösungsbeispielen: Eine effektive, aber kaum genutzte Möglichkeit, Unterricht zu gestalten. In J. Doll & M. Prenzel (Hg.), *Studien zur Verbesserung der Bildungsqualität von Schule: Lehrerprofessionalisierung, Unterrichtsentwicklung und Schülerförderung* (S. 77–92). Münster: Waxmann.

Girwidz, R. & Watzka, B. (Hg.) (2018). *Naturwissenschaft im Unterricht Physik „Arduino, Raspberry Pi & Co."*. 167(29). Hannover: Friedrich Verlag.

Verein zur Förderung des physikalischen und chemischen Unterrichtsund des Fachausschusses Physik & Schule der Österreichischen Physikalischen Gesellschaft (2018). *Plus Lucis. Themenheft "Arduino"*. 1/2018 *https://www.pluslucis.org/ZeitschriftenArchiv/2018-1_PL.pdf*

Wodzinski, R. (2013). Lernen mit gestuften Hilfen. *Physik Journal*, 12, 45-49.

Dr. Alexander Pusch lehrt und forscht am Institut für Didaktik der Physik an der Westfälischen Wilhelms-Universität Münster. Alle genannten Inhalte finden Sie unter https://physikkommunizieren.de/arduino/einstieg/ zum Download.

MNU-Journal – Ausgabe 02.2023 – ISSN 0025-5866 – © Verlag Klaus Seeberger, Neuss