Infokarte: Snap4Arduino

Ein Arduino-Projekt erstellen

Um ein neues Arduino-Projekt in Snap4Arduino zu erstellen, wird das Programm geöffnet. Die Sprache lässt sich über das „Zahnrad"-Symbol einstellen. Snap4Arduino erzeugt automatisch ein neues Projekt. Ein vorhandenes Projekt öffnet man über das Datei-Symbol und die Auswahl „Öffnen…" (Abb. 1). Projekte von einer externen Quelle können geladen werden, indem man im selben Menü auf „Importieren…" klickt und das entsprechende Projekt auswählt (Abb. 2). Nachdem der Arduino mit dem Computer per USB-Kabel verbunden ist, muss er noch mit Snap4Arduino eingerichtet werden. Hierzu klickt man oben links auf „Arduino" und anschließend auf „Mit Arduino verbinden" (Abb. 3).

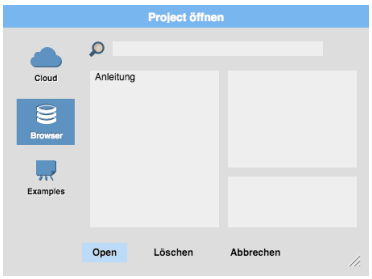


Abb. 1

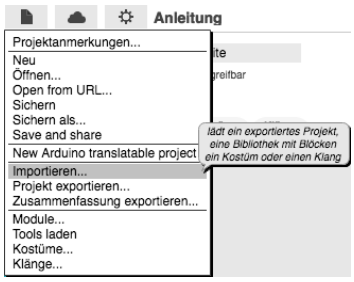


Abb. 2

Abb. 3

Falls neu angelegt, sollte das Projekt danach mit eigenem Namen gesichert werden (Abb. 4, Abb. 5).

Abb. 4

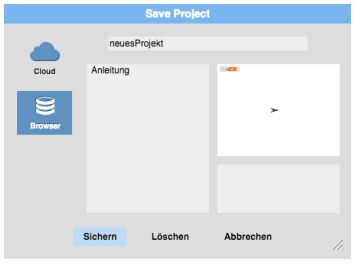


Abb. 5

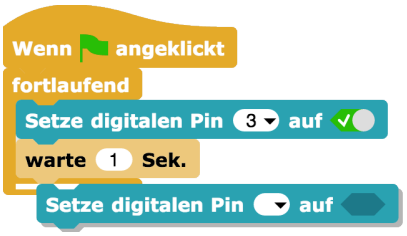


Abb. 6

Snap4Arduino kommuniziert mit dem Arduino über die Blöcke in der Kategorie „Arduino". Zur Erstellung eines Programms werden außerdem Blöcke aus den Kategorien „Steuerung", „Operatoren" und „Variablen" benötigt (Abb. 3). Die anderen Kategorien dienen ausschließlich dazu, Sprites auf der Bühne (das weiße Feld mit dem Pfeil in der Mitte) zu manipulieren. Ein Snap4Arduino-Programm beginnt typischerweise mit dem Block 

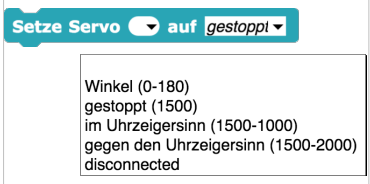
. Daran anschließend folgt die Erstellung der Blöcke des eigentlichen Programms per Drag & Drop aus den entsprechenden Kategorien (Abb. 6).

Über die Funktion „Tools laden" im Datei-Menu lassen sich weitere nützliche Blöcke importieren, die die Programmierung mit Snap4Arduino erleichtern.

Die Arduino-Blöcke

Es gibt fünf verschiedene Arduino-Blöcke. Diese werden genutzt, um an den einzelnen Pins anliegende Signale der angeschlossenen Sensoren auszulesen und um Signale über die Pins nach außen an die Aktoren zu geben.

Setze Servo



Dieser Block dient dazu, einem angeschlossenen Servomotor einen Wert zuzuweisen.

TinkerKit-Shield:
Der mitgelieferte Servomotor kann an allen digitalen Pins (3, 5, 6, 9, 10 und 11) genutzt werden. Servomotoren dürfen nur mit zusätzlicher Stromquelle (Netzteil oder Batterie) betrieben werden!

Es gibt zwei Arten von Servomotoren. Continuous Rotation (CR)-Servomotoren können sich kontinuierlich mit oder gegen den Uhrzeigersinn drehen, Standard-Servomotoren können auf einen bestimmten Winkel eingestellt werden. Die Auswahl im ersten, abgerundeten Feld muss dem Pin entsprechen, an dem der Servomotor angeschlossen ist. (Bsp.: Servomotor an 3 —> Auswahl „3")

Erklärung des Auswahlmenüs
Winkel (0-180): Der Winkel für einen angeschlossenen Standard-Servomotor wird hier eingestellt. Dazu wird ein Wert zwischen 0 und 180 eingegeben, was ungefähr einer Bewegung des Servomotors zwischen 0° bis 180°. entspricht. Der genaue Minimal- und Maximalwert kann von Motor zu Motor unterschiedlich sein, sollte sich aber in der Nähe von 0 und 180 befinden. Hier muss experimentiert werden. Einige Servomotoren lassen sich auch über ein kleines Stellrädchen mit einem Schraubendreher physisch justieren.

Folgende Menü-Punkte sind ausschließlich für CR-Servomotoren:
gestoppt: Ein CR-Servomotor wird mit diesem Block angehalten. Sollte der Motor nicht komplett anhalten, muss der tatsächliche Stoppwert experimentell ermittelt werden. Der Wert befindet sich in der Nähe von 1500 und muss in diesem Fall manuell als Zahl eingegeben werden, indem man einmal auf das bereits ausgewählte Wort „gestoppt" klickt. Auf diese Weise lässt sich auch die Drehgeschwindigkeit regulieren.

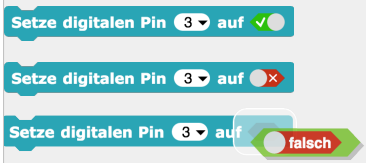
im Uhrzeigersinn: Ein CR-Servomotor wird mit diesem Block gestartet und dreht sich fortan kontinuierlich mit dem Uhrzeigersinn.

gegen den Uhrzeigersinn: Ein CR-Servomotor wird mit diesem Block gestartet und dreht sich fortan kontinuierlich gegen den Uhrzeigersinn.

Das TinkerKit enthält einen Standard- und einen CR-Servomotor.

TinkerKit: Die digitalen Pins 3, 5, 6, 9, 10 und 11 des Arduino Uno werden durchgereicht und können genutzt werden.

Setze digitalen Pin



Dem angegebenen digitalen Ausgang wird ein Wahrheitswert (wahr oder falsch) zugewiesen. Das bedeutet, dass an den Pin die entsprechende Spannung (5V) angelegt oder dieser geerdet wird (0V). In dem abgerundeten Feld hinter dem Wort „Pin" wird die Nummer des Pins eingetragen. Im rechten Sechseck wird der Wahrheitswert per Klick eingetragen. Alternativ kann er auch per Drag & Drop hineingezogen werden.

TinkerKit: Die digitalen Pins 3, 5, 6, 9, 10 und 11 des Arduino Uno werden durchgereicht und können genutzt werden.

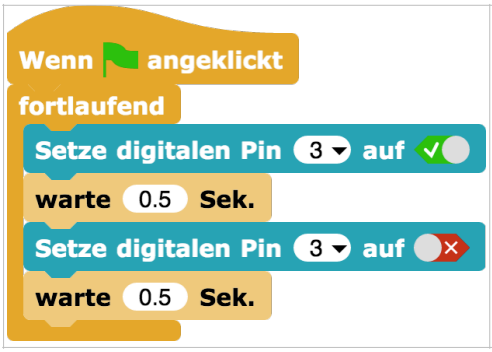
Setze Pin 	Die digitalen Pins 3, 5, 6, 9, 10 und 11 des Arduino lassen sich per PWM nutzen. PWM steht für Pulsweitenmodulation und bedeutet, dass die digitalen Signale in pseudoanaloge Signale umgewandelt werden. So entsteht die Möglichkeit, 256 verschiedene Werte (0 bis 255) ausgeben zu können, die im rechten abgerundeten Feld des Blocks eingegeben werden. Im linken Auswahlfeld wählt man den entsprechenden PWM-Pin. **TinkerKit-Shield:** Die digitalen Pins 3, 5, 6, 9, 10 und 11 des Arduino Uno werden durchgereicht und können genutzt werden.
Lies analogen Pin 	Mit diesem Block können analog angeschlossene Sensoren ausgelesen werden. Die am Pin anliegenden Spannungswerte (0 bis 5 Volt) werden hierbei in 1024 verschiedene Zahlenwerte (0 bis 1023) umgerechnet. Nur die im Bereich „ANALOG IN" befindlichen Pins A0 bis A5 am Arduino können als analoge Eingänge verwendet werden. Der entsprechende Eingang wird im Auswahlmenü gewählt. Es handelt sich bei diesem Block um einen Reporter-Block (abgerundete Gestalt). Das bedeutet, dass bei Klick auf den Block der aktuelle Wert des gewählten Pins angezeigt wird. **TinkerKit-Shield:** Alle analogen Pins des Arduino Uno werden durchgereicht und können genutzt werden.
Lies digitalen Pin 	Mit diesem Block können digital angeschlossene Sensoren ausgelesen werden. Die am Pin anliegenden Spannungswerte (0 oder 5 Volt) werden hierbei in die Wahrheitswerte ( oder ) umgerechnet. Der entsprechende Eingang wird im Auswahlmenü gewählt. Es handelt sich bei diesem Block um einen Reporter-Block (abgerundete Gestalt). Das bedeutet, dass bei Klick auf den Block der aktuelle Wert des gewählten Pins angezeigt wird. **TinkerKit-Shield:** Die digitalen Pins 3, 5, 6, 9, 10 und 11 des Arduino Uno werden durchgereicht und können genutzt werden.

Beispiele

Ein Programm wird gestartet, indem die grüne Flagge oben rechts im Fenster angeklickt wird (Abb. 7). Ein einzelner Programmblock kann auch gestartet werden, indem mit der linken Maustaste einmal auf den Programmblock geklickt wird. Laufende Programme sind grün umrandet. Enthält ein Programm Fehler und kann nicht ausgeführt werden, so ist es rot umrandet. Um das Programm wieder zu beenden, wird der rote Punkt oben rechts in der Ecke des Programmfensters angeklickt oder erneut einmal mit der linken Maustaste auf den Programmblock geklickt.

Abb. 7

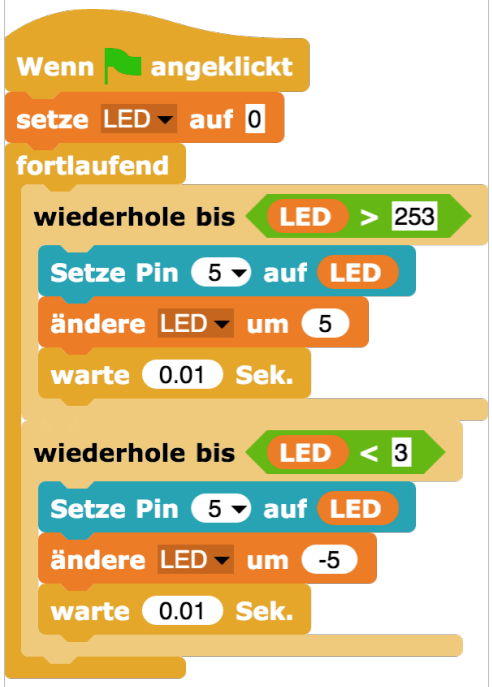
Blinkende LED



Zunächst wird der Start-Block mit der grünen Fahne hinzugefügt.

Im Sch[wahr]lock w[falsch]dem digitalen Pin 3 im Wechsel die Werte und zugewiesen. Damit sich ein nicht zu schnelles Blinken ergibt, wurde zusätzlich ein Warteblock eingefügt, der die angeschlossene LED jeweils 0,5 Sekunden in ihrem Zustand belässt. Alles im „fortlaufend"-Block Befindliche wird in einer Dauerschleife solange ausgeführt, bis das Programm beendet oder die Verbindung zum Arduino getrennt wird.

Selbstdimmende LED



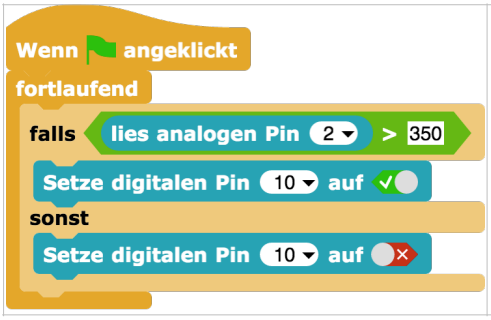
Für dieses kurze Programm wird die Variable „LED" benötigt, in welcher der aktuelle Helligkeitswert der LED gespeichert wird. Die Variable wird durch Klick auf den Button „Neue Variable" innerhalb der Kategorie „Variablen" erzeugt.

Zunächst wird der Anfangswert der Variablen auf 0 gesetzt. Die LED wird also zu Programmbeginn ausgeschaltet sein.

Im Schleifen-Block werden der LED nun immer neue Helligkeitswerte zugewiesen. Hierzu gibt es zwei Wiederholungs-Schleifen. Diese Schleifen werden ausgeführt, bis die Bedingung im Sechseck hinter den Worten „wiederhole bis" erfüllt ist. In diesem Fall ist die überprüfte Bedingung der aktuelle Helligkeitswert der LED. Innerhalb der Schleife wird der Helligkeitswert jedes Mal um den Wert 5 (bzw. -5) geändert, so dass die angegebene Bedingung irgendwann wahr wird.
Der Wert der Variablen LED wird in jedem Schleifendurchlauf an die angeschlossene LED an Pin 5 ausgegeben, so dass sich ihre tatsächliche Helligkeit entsprechend ändert.

Damit sich ein nicht zu schnelles Dimmen ergibt, wurde zusätzlich ein Warteblock eingefügt, der die angeschlossene LED jeweils 0,01 Sekunden in ihrem Zustand belässt. Alles im Schleifen-Block Befindliche wird in einer Dauerschleife solange ausgeführt, bis das Programm beendet oder die Verbindung zum Arduino getrennt wird.

LED in Abhängigkeit von Sensorwert an-/ausschalten



Im Schleifen-Block werden dem digitalen Ausgang 10 in Abhängigkeit von den am analogen Eingang A2 eingelesenen Werten die Werte wahr und falsch zugewiesen. Ist also der aktuelle Sensorwert an A2 höher als 350, wird die LED angeschaltet. Sonst (also wenn der Sensorwert kleiner oder gleich 350 ist) wird die LED ausgeschaltet.