SNAP! Workshop

Hopp Foundation

01. Februar 2020

Inhalt

Einführung in Snap!

Um mit Snap! arbeiten zu können, gibt es verschiedene Möglichkeiten:

1. Die Online-Version unter https://snap.berkeley.edu/
2. Die Offline-Version von Snap! unter https://github.com/jmoenig/Snap/releases/latest downloaden, entpacken und snap.html öffnen.

Hinweise für den Unterricht

- Bei der Online-Version wird der Internet Explorer nicht unterstützt, alle weiteren gängigen Browser sollten funktionieren.
- In der Offline-Version kann die Cloud-Speicherung nicht verwendet werden.
- Snap! ist nicht in Form einer App für Tablets zugänglich und wird es (vermutlich) auch nicht werden. Das Öffnen der Online-Version mit dem Tablet ist nicht zu empfehlen.

Laden und Speichern

Snap! bietet verschiedene Möglichkeiten, um Projekte zu verwalten. Hierzu zählen das Speichern im Browser, die Verwendung einer Cloud (Anmeldung mit E-Mail-Adresse notwendig) und das Exportieren als .xml-Datei. Es empfiehlt sich die letzteren beiden Varianten zu verwenden, damit die Schüler/innen ihre Projekte austauschen oder zu Hause daran weiterarbeiten können. Um Dateien auf den Computer oder einen USB-Stick zu speichern, verwenden Sie den Menüpunkt *Exportieren*, für die anderen beiden Varianten *Sichern als*.

Beim Laden können Sie entweder die Datei direkt in den Skriptbereich ziehen oder über den Menüpunkt *importieren* öffnen. Das Laden aus der Cloud oder dem Browser funktioniert über den Menüpunkt *öffnen.*

Teilen und Veröffentlichen

Nutzen Sie die Cloud-Funktion, so haben Sie die Möglichkeit Ihr Projekt zu teilen oder auf der Website von Snap! zu veröffentlichen. Hierzu speichern Sie Ihr Projekt in der Cloud ab und klicken anschließend auf teilen. Anschließend können Sie den aktuellen Link im Browser kopieren und Ihren Kollegen/Kolleginnen oder Schüler/innen zur Verfügung stellen. Diese können dann das Projekt als Grundlage für Ihre Arbeit verwenden oder Ihnen bei Problemen Rückmeldung geben.

Hinweis: Das Teilen wird besonders einfach, wenn Sie die Dienste von z.B. tinyurl.com nutzen. Diese Website kürzt Ihren Link. Nutzen Sie am besten die Option, in der Sie eingeben können, welcher Link entstehen soll. Beispielsweise finden Sie unter tinyurl.com/HoppWorkshops das Workshopangebot der Hopp-Foundation. Ihr gewählter Link (sofern noch verfügbar) ist dauerhaft verfügbar und wird niemals mit einer anderen Website überschrieben.

Mit der Teilen-Funktion Apps erstellen

Immer dann, wenn Sie einen geteilten Link öffnen, startet das Projekt so, dass die Bühne im Vollbildmodus angezeigt wird. Öffnen Sie es mit einem mobilen Endgerät, so sieht es aus, als hätten Sie eine App mit Benutzeroberfläche programmiert. Dabei sind Tastatureingaben nicht mehr möglich, aber das Antippen der Bühne entspricht der Aktion „Maustaste gedrückt“.

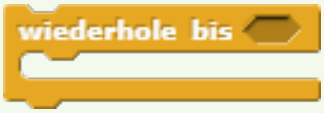
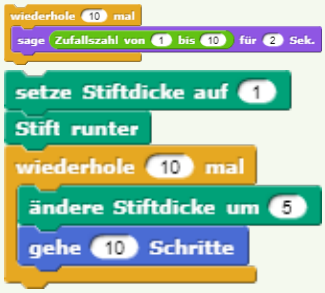
Weitere wichtige Funktionen im Menübereich

- Sprache wählen:
  Projekte können unabhängig von der gewählten Sprache durchgeführt werden. So haben Schüler/innen mit Sprachschwierigkeiten die Möglichkeit, in ihre Muttersprache zu wechseln.

  Hinweis: Verstellt man die Sprache auf z.B. Arabisch, ist es sehr schwer den Menüpunkt Sprache (nun in arabisch angezeigt) wiederzufinden, um die Sprache wieder auf Deutsch umzustellen. Merken Sie sich hierfür, dass die Spracheinstellung immer der erste Menüpunkt ist. Außerdem wird in manchen Fällen beim Umstellen der Sprache der Projekt- oder der Objektname (Name des Pfeils auf der Bühne) nicht geändert. Aktualisieren Sie hierfür ihren Browser.
- Blöcke vergrößern:
  Für Präsentationen am Beamer ist es sinnvoll, die Blockgröße zu verändern, damit das Programm für alle Schüler/innen gut sichtbar ist.
- Programmausführung verfolgen durch Verlangsamung: 
  Um das Programm schrittweise nachvollziehen zu können, kann die Ausführungsgeschwindigkeit verringert werden. Der Einsatz dieser Funktion ist auch dann sinnvoll, wenn das Programm nicht das Erwartete tut (Fehlerbehandlung).

Das Programmierkonstrukt *Schleife*

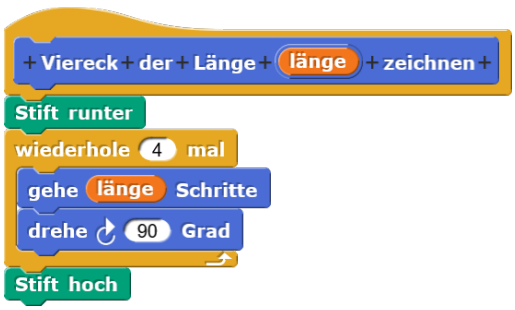
Wiederholen sich Anweisungen im Programm, ist es sinnvoll eine Schleife zu verwenden. Hierfür stehen folgende Blöcke zur Verfügung:

Block			
Bedeutung	Die Blöcke innerhalb des Blockes werden unendlich oft hintereinander ausgeführt.	Die Blöcke innerhalb des Blockes werden 10 mal hintereinander ausgeführt.	Die Blöcke innerhalb des Blockes werden solange ausgeführt, bis eine Bedingung eintritt.
Beispiele			

Blöcke erstellen

Um einen eigenen Block erstellen zu können, wählen Sie in der Kategorie, in der Ihr neuer Block später zu finden sein soll, *Neuer Block* aus. Geben Sie Ihrem Block einen Namen und bestätigen Sie mit *ok*. Soll Ihr Block eine Eingabe beinhalten (wie z.B. der Block  , der als Eingabegröße die Anzahl der Schritte hat), können Sie diese direkt im Namen des Blocks durch ein Prozentzeichen gefolgt von dem Variablennamen kennzeichnen. So erzeugt zum Beispiel die Eingabe *Viereck der Länge %Länge zeichnen* folgenden Block:

Es öffnet sich der Blockeditor, wo Sie nun alle notwendigen Blöcke hinzufügen können. Für das Beispiel des Vierecks ergibt sich



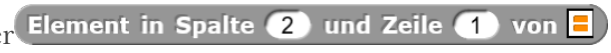
Den Block *länge* erhalten Sie durch Drag and Drop aus dem Blockanfang.

Aktuell kann als Parameter des Blocks alles eingefügt werden, so zum Beispiel auch Zeichenketten wie „Hallo“. Dies wollen wir natürlich verhindern, da der Befehl „gehe Hallo Schritte“ nicht möglich ist.

Damit nur gewünschte Eingaben bei der Verwendung des Blocks gemacht werden können, klicken Sie innerhalb der Blockdefinition doppelt auf den Eingabeparameter und anschließend auf den Pfeil, der nach rechts zeigt und wählen den korrekten Eingabetyp aus. In unserem Fall handelt es sich um ein Zahlfeld.

Das Eingabefeld des Blocks ändert nun seine Form von viereckig in oval und lässt Buchstabeneingaben per Tastatur nicht mehr zu. Beachten Sie, dass Variablen vom Typ String (Zeichenketten) trotzdem als Eingabe eingefügt werden können.

Beachten Sie unbedingt: Definierte Blöcke können nur innerhalb des aktuellen Projekts verwendet werden und sind bei einem neuen Projekt nicht mehr verfügbar. Möchten Sie einen Block in einem anderen Projekt verwenden, nutzen Sie den Menüpunkt „Blöcke exportieren“. Zum Laden der Blöcke wählen Sie „Importieren…“. Diese Funktion können Sie auch zur didaktischen Reduktion nutzen. So können Sie beispielsweise statt 

für das Auslesen der n-ten Spalte und m-ten Zeile einer Liste Ihren Schüler/innen den Reporter 

zur Verfügung stellen.

Sie wollen Ihren Block später noch einmal abändern? Machen Sie einen Rechtsklick auf den Block und wählen *bearbeiten* aus. Durch Klicken auf *Blockdefinition löschen* können Sie Ihren Block löschen.

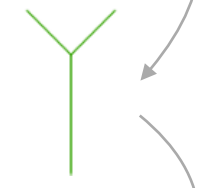
Bedingte Anweisungen

Möchten Sie gewisse Blöcke nur in einem bestimmten Fall ausführen, dann spricht man von einer bedingten Anweisung. Hierfür steht Ihnen in Snap! der Block *falls* zur Verfügung. Ein Anwendungsbeispiel können Sie in der obigen Tabelle bei *Das Programmierkonstrukt Schleife* finden.

Rekursion

Wird innerhalb eines Blocks der Block selbst wieder aufgerufen, dann spricht man von einer Rekursion. Beachten Sie bitte unbedingt, dass Sie bei einer Rekursion immer einen sogenannten Basisfall benötigen, bei dem das erneute Aufrufen durch den Block nicht mehr stattfindet. Als Beispiel haben Sie den rekursiven Baum kennengelernt, der auf folgende Grundform, den Ast, zurückzuführen ist:

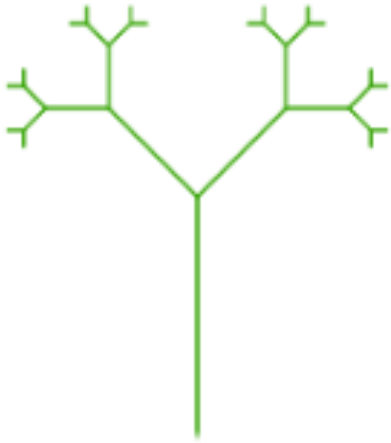
Nach einer Drehung um 45 Grad nach links kann rekursiv ein Ast mit halber Länge hinzugefügt werden. Anschließend dreht man sich um 90 Grad nach rechts und fügt einen weiteren Ast mit der halben Ausgangslänge hinzu. Es ergibt sich:



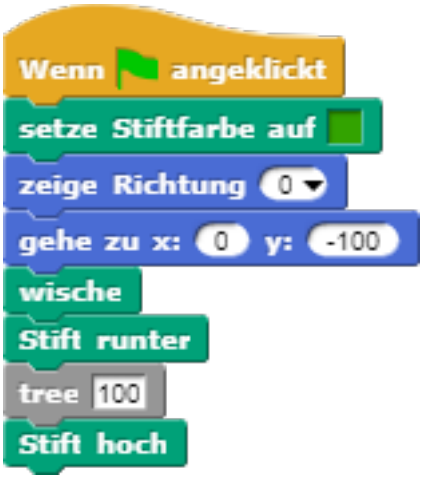
Verfahren Sie für den linken und rechten Ast dieser zwei neu entstandenen Äste genauso, ergeben sich vier weitere Äste:

Starten Sie also beispielsweise mit einer Basislänge von 100, könnten Sie nun bei 12.5 den Basisfall eintreten lassen und ihre Rekursion endet mit dem obigen Baum. Für eine weitere Iteration setzen Sie den Basisfall auf 6.25 usw.

Für den Basisfall 3.125 ergibt sich:

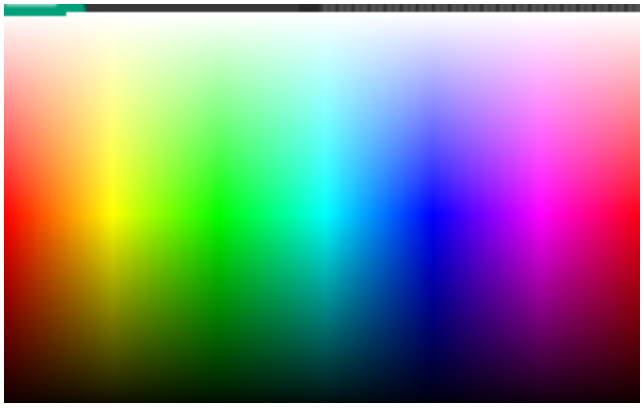


Codebeispiel zum Ausprobieren:



Weitere nützliche Tipps

- Falls Sie einen Block nicht auf Anhieb finden, können Sie mit *Strg+F* die Suchfunktion aktivieren. Alternativ befindet sich in jeder Kategorie eine Lupe. Durch einen Klick auf diese kann kategorieübergreifend nach Blöcken gesucht werden.
- Mit einem Rechtsklick auf einen Block im Skriptbereich (oder mehrere aneinanderhängende Blöcke) öffnet sich ein Menüfeld, bei dem Sie *Skriptbild* auswählen können. Es wird ein Bild der Blöcke erstellt, das Sie z.B. für Arbeitsblätter verwenden können. Möchten Sie ein Bild von allen im Skriptbereich befindlichen Blöcken, so können Sie mit einem Klick auf einen leeren Skriptbereich ebenfalls *Skriptbild* auswählen. Genau so können auch Bühnenbilder gespeichert werden, wenn Sie nach einem Rechtsklick auf die Bühne *Bild exportieren* auswählen.
- Der Block „ändere Stiftfarbe um“ könnte Sie verzweifeln lassen, da dessen Verwendung keine Änderung der Stiftfarbe zur Folge haben könnte. Dies liegt daran, dass die Farbe in der Farbpalette immer horizontal nach rechts verschoben wird.



Ist die gewählte Farbe schwarz oder weiß, wird zwar eine horizontale Änderung der Farbe vorgenommen, diese kann aber mit dem Auge nicht wahrgenommen werden. Setzen Sie die Startfarbe deshalb auf eine beliebige andere Farbe, dann kann auch eine Veränderung bei der Verwendung des „ändere Stiftfarbe um“-Blocks wahrgenommen werden.

- Der Pfeil ist auf der Bühne nicht mehr sichtbar? Kein Problem! Mit einem Klick auf den Block 

ist er wieder da. Alternativ können Sie auch unterhalb der Bühne einen Rechtsklick auf das Objekt machen und im Menüfeld *anzeigen* auswählen.

Turtlestitch

Turtlestitch stellt eine Erweiterung von Snap! dar. Hierdurch kann ein Zugang zur Programmierung für Kinder und Jugendliche über das Designen von Kleidung geschaffen werden. Dass diese Form des Designens eine Alltagsrelevanz besitzt, zeigen Modelabels, die Teile ihrer Kollektion mit Turtlestitch designed haben.

Durch die blockbasierte und intuitive Programmierung sieht man sofort, was später von einer Nähmaschine auf die Kleidung genäht wird. Dies erlaubt Ausprobieren und kreatives Gestalten eines eigenen Musters.

Leider funktioniert Turtlestitch nicht mit jeder beliebigen Nähmaschine. Eine kompatible Nähmaschine liegt in der Anschaffung bei circa 750€, wobei eine einzige Nähmaschine für den Einsatz im Unterricht vollkommen ausreicht, da das Nähen sehr schnell geht. Wichtig ist bei der Anschaffung darauf zu achten, dass die Nähmaschine .exp oder .dst Dateien lesen kann. Eine Konvertierung in eines der Dateiformate ist zwar generell möglich, allerdings mit einem möglichen Qualitätsverlust verbunden.

Die Programmierumgebung

Zur Programmierung steht eine eigene Website https://www.turtlestitch.org zur Verfügung. Diese Programmierumgebung basiert auf Snap!, wurde jedoch in ihrer Funktionalität auf die Anforderungen des Nähens angepasst.

Hinweis: Da es sich hierbei um eine „abgespeckte“ Version von Snap! handelt, können nicht alle Programme von Snap! in TurtleStitch importiert werden.

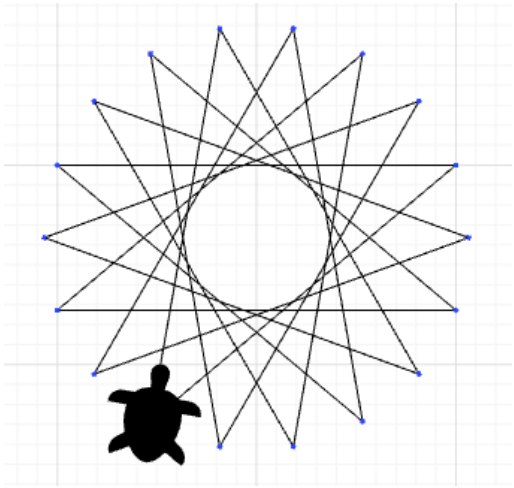
Die Turtle repräsentiert die Nadel unserer Nähmaschine. Die Linie, die die Turtle hinterlässt, ist die Linie, die später genäht wird. Bei genauerer Betrachtung sind auf der Linie die späteren Einstichstellen der Nadel zu erkennen. Bei der Programmierung gibt es folgendes zu beachten:

- Das Ergebnis wird am besten, wenn die Turtle in Abständen von 8 bis 12 Schritten bewegt wird. Die Anzahl der Schritte bestimmt die Größe einer Masche. Unter 8 Schritten werden die Maschen zu dicht, bei über 12 Schritten wiederum zu groß. Dies kann zu Problemen beim Nähen führen.
- Die Linien sollten sich nicht zu häufig überschneiden, damit sich die Fäden nicht gegenseitig stören.

Im Folgenden ein kurzes Beispiel für ein mögliches Muster:

Programm

Ergebnis



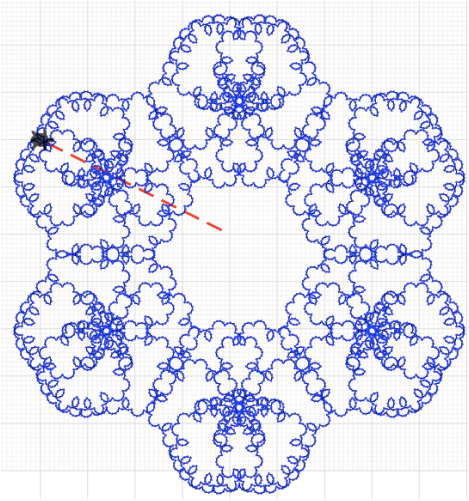
Vom Programm zur Nähmaschine

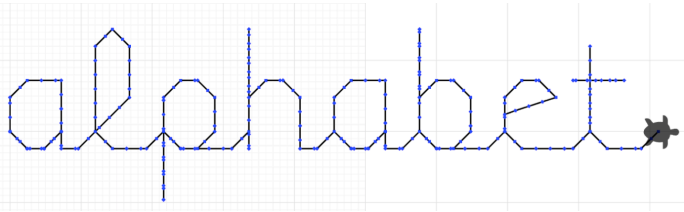
Um ein Programm nun auf die Nähmaschine zu übertragen, gehen Sie wie folgt vor:

1. Klicken Sie auf 
2. Exportieren Sie das Programm als EXP oder DST.
3. Speichern Sie die Datei auf einem USB-Stick und stecken Sie diesen in die Nähmaschine.
4. Spannen Sie ein Stück Stoff zusammen mit etwas Stickvlies in die Vorrichtung der Nähmaschine.
5. Starten Sie das Nähen.

Hinweis: Da sich jede Nähmaschine in ihrer Bedienung unterscheidet, wurde hier auf eine genauere Beschreibung verzichtet.

Projektideen





Listen

Anders als in Scratch sind Listen first class, das heißt sie können

- in Variablen gespeichert werden
- an Funktionen übergeben werden
- von Funktionen zurückgegeben werden
- ineinander verschachtelt werden (Listen in Listen)

Dadurch können komplexe und abstrakte Datentypen wie z.B. Stack (Stapel), Queue (Schlange) und Bäume implementiert werden.

Wie verwendet man Listen?

1) Vorbereitung
   Zunächst müssen die entsprechenden Blöcke zur Verwendung von Listen importiert werden:
   Klicken Sie auf  —> Module… —> Listen bearbeiten —> Import
   Nach dem Importieren sind die neuen Blöcke unter der Kategorie *Variablen* zu finden.

2) Liste anlegen
   manuell:  füllen, z.B. mit Zahlen, Buchstaben oder weiteren Listen
   automatisch: Erzeugen Sie eine Variable *Liste1* und weisen Sie dieser eine leere Liste zu. Anschließend kann die Liste mithilfe einer Schleife befüllt werden.

3) Liste anzeigen
   Um die Elemente einer Liste anzuzeigen, klicken Sie auf  .

4) Liste verändern (map, filter, kombinieren)
   1) map = wende eine Funktion auf **alle** Element einer Liste an
       : grau umringte Platzhalter sind für Funktionen vorgesehen

      Beispiel:
      Aufgabe: *Liste2* soll alle Listenelemente von *Liste1* um 10 erhöht abspeichern.
      - Zunächst der *Liste2* die *Liste1* zuweisen 
      - Auf alle Elemente von *Liste1* 10 addieren: 
      - Ergebnis: 

2) filter = behalte nur Element, für die der lambda-Ausdruck wahr wird

Beispiel:

Aufgabe: Nur ungeraden Zahlen sollen in der Liste enthalten bleiben.

- Zunächst einer neuen *Liste3* wie im vorherigen Beispiel die *Liste1* zuweisen.
- Ungerade Zahlen ergeben bei der Division durch 2 stets den Rest 1. Um zu überprüfen, ob eine Zahl ungerade ist, kann man die modulo-Operation benutzen. Folgende Blöcke werden benötigt:

- Ergebnis:

3) Kombiniere alle Element einer Liste

Beispiel:

Aufgabe: Berechne die Summe aller Listenelement.

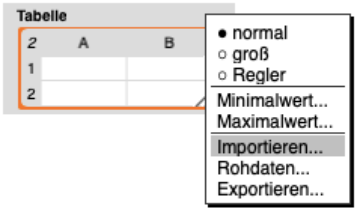
- Legen Sie eine Variable *summe* an, in der das Ergebnis gespeichert wird.
- Als Funktion wird die Addition  genutzt und auf die Elemente von *Liste1* angewendet.
- Ergebnis:

Tabellen

Mit Hilfe von ineinander verschachtelten Listen können Tabellen erzeugt werden.

Beispiel: Eine Tabelle mit zwei Spalten und zwei Zeilen könnte wie folgt manuell erzeugt werden.

Alternativ kann eine Tabelle auch bspw. mit Excel angelegt werden. Die Exceltabelle kann als .csv-Datei gespeichert werden und anschließend in Snap! importiert werden.



Umgekehrt können natürlich in Snap! angelegte Tabellen als .csv-Datei exportiert werden und in Excel weiterverarbeitet werden.

Nachdem eine Tabelle erstellt bzw. importiert wurde, soll nun auf bestimmte Inhalte zugegriffen werden. Da es sich bei einer Tabelle um eine Liste von Listen handelt, ist das erste Element der Tabelle 

gerade die erste Tabellenzeile.

Soll nun auf ein bestimmtes Tabellenfeld zugegriffen werden, muss zusätzlich noch die gewünscht Spalte ausgewählt werden. Durch 

wird bspw. auf das Tabellenfeld in der ersten Spalte und der ersten Zeile zugegriffen.

Da man bei vielen Zugriffen auf verschiedene Inhalte einer Tabelle schnell durcheinander kommen kann, ist es empfehlenswert einen eigenen Funktionsblock für den Zugriff auf bestimmte Zeilen und Spalten einer Tabelle zu definieren.

Somit steht nun der praktische Listen-Block 

zur Verfügung.

Anwendungen von Listen

1. **Bildbearbeitung**
   Das Thema Bildbearbeitung lässt sich in Informatik Klasse 7 - Daten und Codierung einbringen.

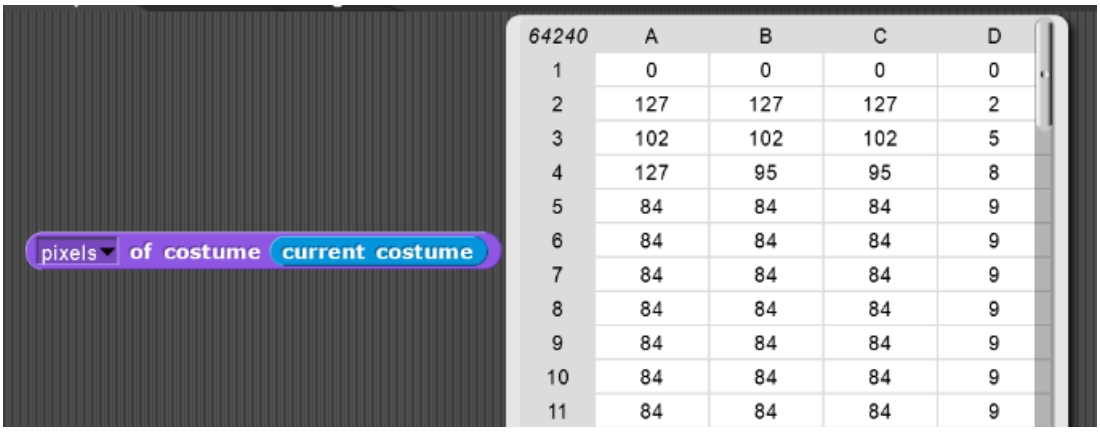
   1.1. Vorbereitung
   Zur Vorbereitung muss zunächst das Modul *pixels* importiert werden.

   1.2. Bild hinzufügen
   Anschließend wird ein Bild zur Bearbeitung benötigt. Hier bietet es sich an, (falls möglich) ein eigenes Foto aufzunehmen. Hierfür klickt man einfach auf  unterhalb der Bühne. Alternativ kann auch ein anderes Bild per Drag&Drop eingefügt werden.

1.3. Pixel auslesen

Klicken Sie auf 

.



64240	A	B	C	D
1	0	0	0	0
2	127	127	127	2
3	102	102	102	5
4	127	95	95	8
5	84	84	84	9
6	84	84	84	9
7	84	84	84	9
8	84	84	84	9
9	84	84	84	9
10	84	84	84	9
11	84	84	84	9

A = rot, B = grün, C = blau, D = Transparenz (alpha-Kanal)

1.4. Pixel verändern

Nach dem Auslesen der Pixel können diese auf verschiedene Art und Weise verändert werden. Nachfolgend sind einige Beispiele für verschiedene Bildeffekte angegeben.

- Monochrom: Nur einer von drei Kanälen.

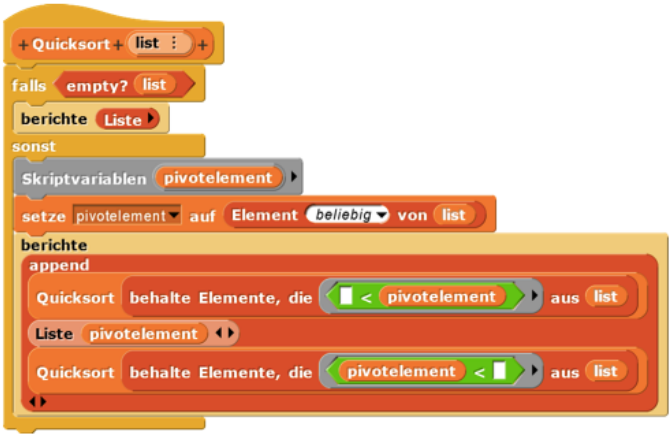
- Negativ

- Farbentausch

- Alpha-Kanal ändern mit Schieberegler

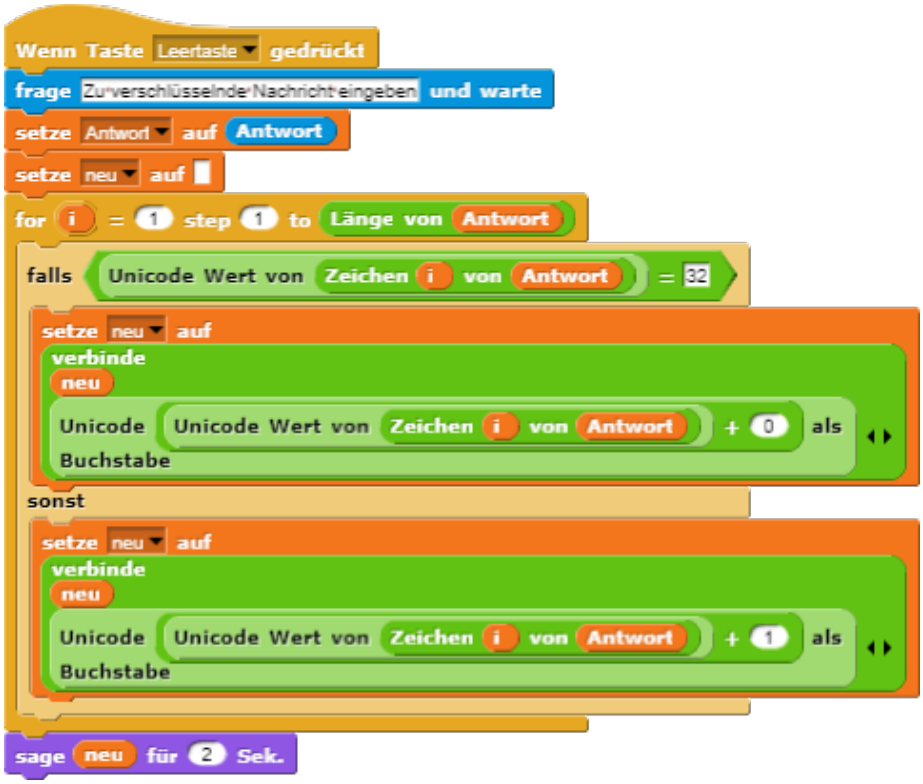
2. Sortieren

In Snap! lassen sich Sortierverfahren wie bspw. Quicksort einfach umsetzen. Hierfür ist das Importieren des Moduls *Listen bearbeiten* nötig.



3. Verschlüsselung

Einfache Verschlüsselungsverfahren wie die abgebildete Caesar-Verschlüsselung lassen sich ebenfalls gut in Snap! umsetzen.



4. Vokalbeltrainer (inkl. Sprachausgabe)

Vorbereitung:

Anlegen einer Liste von Wortpaaren (Liste von Listen)

Um auf einen bestimmten Eintrag zuzugreifen, genügt es nicht, auf das erste Element des Wörterbuchs zuzugreifen, da hinter jedem Eintrag wiederum eine Liste steht. Die folgende Verschachtelung von Blöcken erlaubt den Zugriff auf ein bestimmtes Wort:

Für den Zugriff auf ein beliebiges Element des Wörterbuchs wird eine Zufallszahl benötigt:

Hinweis: Wichtig ist, dass diese Zufallszahl für jede Vokabelabfrage in einer Variable gespeichert wird!

Erweiterung:

Die Wörter können nicht nur angezeigt, sondern auch vom Programm vorgelesen werden. Hierfür muss das Modul *Sprachausgabe* importiert werden.

Eine mögliche Umsetzung des Vokabeltrainers könnte wie folgt aussehen:

5. Musik

Verschiedene Musikstücke lassen sich einfach in Snap! selbst nachkomponieren. Als Anregung soll folgendes Beispiel dienen:

https://snap.berkeley.edu/snapsource/snap.html#present:Username=jens&ProjectName=Frere%20Jacques&editMode

Weitere Ideen für Projekte und Programme sind unter https://snap.berkeley.edu/ zu finden.

Snap4Arduino

Wäre es nicht interessant, wenn wir innerhalb unseres Snap!-Projektes auf die aktuelle Raumtemperatur, die Helligkeit oder sonstige Sensorwerte reagieren könnten? Oder wenn wir zusätzlich zu unserer Bühne eine weitere Form der Ausgabe mit LEDs, Motoren oder sonstigen Aktoren haben könnten?

Snap4Arduino ist eine Erweiterung der bisherigen Entwicklungsumgebung um genau diese Möglichkeiten. Ebenso wie die Standardversion von Snap! ist auch diese Erweiterung online (http://snap4arduino.rocks/run/), hier allerdings nur mit dem Browser Google Chrome , als auch offline (http://snap4arduino.rocks/) verfügbar. In der Offline-Variante funktioniert im Gegensatz zur Standardversion auch die Speicherung über die Cloud. Somit ist auch das Teilen von Projekten möglich!

Praxistipp: Projekte, die in Snap! entwickelt wurden, können auch in Snap4Arduino geöffnet werden. Es wird allerdings immer ein Warnhinweis angezeigt, da es sein kann, dass es in der einen Version einen Block gibt, den es in der anderen Version nicht gibt. Überprüfen Sie daher kurz, ob alle Blöcke vorhanden sind.

Den Arduino verbinden

Um einen Arduino mit Snap4Arduino verbinden zu können, muss einmalig die Standard-Firmata mithilfe der Arduino-Software (https://www.arduino.cc/en/Main/Software) auf den Arduino übertragen werden. Hierzu wählen Sie *Datei>Beispiele>Firmata>StandardFirmata* aus und laden das Projekt auf den Arduino herunter.

Wichtiger Praxistipp: Beim Herunterladen tritt häufig ein Fehler auf. Vergewissern Sie sich in diesem Fall, dass unter *Werkzeuge>Port* der richtige Port des verbundenen Arduinos ausgewählt ist. Anschließend wechseln Sie zur Version Snap4Arduino und wählen in der Kategorie Arduino *Mit Arduino verbinden* aus.

Nun können Sie verschiedene Sensoren (Bauteile, die etwas messen können) oder Aktoren (Bauteile, die etwas tun können) mit dem Arduino verbinden, die Sie in allen Preiskategorien erwerben können. Für den Workshop haben wir Sensoren und Aktoren der Firma Seeedstudio zum Probieren bereitgestellt, da diese durch ihre einheitliche Pinbelegung besonders einfach zu verwenden und damit für den Einstieg gut geeignet sind. Beachten Sie aber, dass es auch deutlich günstigere Varianten gibt! Folgende Sensoren/Aktoren konnten Sie ausprobieren:

Berührungssensor	Digital (Sensor)
Bewegungssensor	Digital – Sonderfall
Bodenfeuchtigkeit (Grove)	Analog (Sensor)
Button (Grove)	Digital (Sensor)
Button („Günstigere Alternative“)	Digital (Sensor) - Sonderfall
Drucksensor	Analog – Sonderfall
Joystick (Grove)	Analog - Sonderfall
Lautstärkesensor (Grove)	Analog (Sensor)
Rotary Angle Sensor (Grove)	Analog (Sensor)
Servomotor (Grove)	Digital (Aktor) - Sonderfall
Sliding Potentiometer (Grove)	Analog (Sensor)
Temperatursensor (Grove)	Analog (Sensor)
Vibrationsmotor (Grove)	Digital (Aktor)

Praxistipp:

In der Tabelle sind alle Grove Sensoren und Aktoren markiert. Alle anderen Sensoren und Aktoren – als Sonderfall markiert – haben eine aufwendigere Verkabelung und sollten für den ersten Kontakt mit der Soft- und Hardware erst einmal nicht verwendet werden.

Vor dem Verkabeln: Unterschied von analogen und binären digitalen Signalen

Der Arduino verfügt über 14 binäre digitale Pins, die mit den Nummern 0 bis 13 beschriftet sind. Diese Pins können sowohl als Eingang als auch als Ausgang verwendet werden. Dabei können immer nur genau zwei Zustände angenommen werden: LOW (= 0) oder HIGH (= 1).

Vereinfacht gesagt: Mithilfe von digitalen Pins kann man 5V Spannung anlegen oder trennen, oder auslesen, ob Spannung anliegt oder nicht. Folgende Blöcke können hierfür verwendet werden:

Block	Bedeutung
	Als Ausgang; Setzen des digitalen Pins 0 auf HIGH. Nun liegen 5V am Pin an.
	Als Ausgang; Setzen des digitalen Pins 0 auf LOW. Nun liegen 0V am Pin 0 an.
	Als Eingang; Auslesen der Spannung am digitalen Pin 0. Unterscheidung von 2 Fällen: 0V: LOW wird ausgelesen 5V: HIGH wird ausgelesen

Zum Auslesen von analogen Signalen stellt der Arduino die Pins A0 bis A5 zur Verfügung. Auch hier wird das Signal in Form einer anliegenden Spannung umgewandelt. Diese Werte werden durch einen Analog-Digital-Wandler (englisch: **a**nalog **d**igital **c**onverter) in ein digitales Signal umgewandelt. Anders als bei dem binären digitalen Signal werden hier Werte im Bereich 0 bis 1023 angenommen.

Block	Bedeutung
	Als Eingang; Auslesen der Spannung am analogen Pin 0. Je nach anliegender Spannung wird ein Wert von 0 bis 1023 angenommen.

Das Verkabeln von Sensoren und Aktoren

Hinweise zur korrekten Verkabelung finden Sie im nächsten Kapitel *Informationen Sensoren und Aktoren*.

Praxistipps:

- Der Sensorwert ändert sich nicht? Meistens fehlt das *fortlaufend* im Skript.
- Leider scheint die Software noch nicht vollkommen fehlerfrei zu sein. Nutzen Sie, falls möglich, die digitalen Pins 7 – 13 und nicht die digitalen Pins 0 – 6. Durch ihre Sonderstellung auf dem Arduino (z.B. für die serielle Schnittstelle UART) entstehen scheinbar Probleme innerhalb der Software und die Pins funktionieren nicht so wie gewollt. Bei den oberen Pins (7-13) kam es bislang zu keinen Fehlern.

Nützliche Tipps zum Umgang mit Sensoren

Bevor Sie einen Sensor in ein bestehendes Projekt einbinden, sollten Sie ihn immer auf seine Funktionsfähigkeit überprüfen. Lassen Sie sich hierzu in einem kleinen Projekt die Sensorwerte ausgeben. So kann die häufigste Fehlerquelle, die falsche Verkabelung, ausgeschlossen werden. Hierzu legen Sie sich eine Variable (z.B. *Sensorwert*) an, weisen ihr den Wert des Pins zu und lassen sich diesen anzeigen.

Aktion	Beispielcode	Steckplatz
Analogen Wert auslesen		z.B. A0
Digitalen Wert auslesen		z.B. 7
Digitalen Wert setzen		z.B. 8

Informationen Sensoren und Aktoren

Grove-Sensoren richtig verkabeln

Bei allen Grove-Sensoren finden Sie vier Anschlusspins (GND,VCC,NC,SIG), die auf der Rückseite des Sensors beschriftet sind. Verbinden Sie:

- den **schwarzen** GND Pin mit einem GND Pin auf dem Arduino
- den **roten** VCC Pin mit 5V auf dem Arduino
- den **weißen** NC Pin (in den meisten Fällen **n**ot **c**onnected Pin) verbinden Sie nicht mit dem Arduino
- den **gelben** SIG (signal) Pin verbinden Sie mit einem analogen oder digitalen Pin des Arduinos (je nach Sensortyp den Sie aus der folgenden Tabelle entnehmen können)

ODER:

- Verwenden Sie das Grove-Shield und die normierten Verbindungskabel. Stecken Sie dazu das Shield (Kosten von circa 5 Euro) auf den Arduino. Die digitalen Pins 2 bis 8 sind mit D2 bis D8 beschriftet, die analogen Pins A0 bis A3 sind gleichnamig zu verwenden.

Bei den als Sonderfall markierten Sensoren handelt es sich nicht um Grove-Sensoren oder Aktoren und es muss eine andere Verkabelung (auf den folgenden Seiten beschrieben) als oben beschrieben vorgenommen werden.

Rotary Angle Sensor	Analog (Sensor)
Button (Grove)	Digital (Sensor)
Button („Billig“)	**Digital - Sonderfall**
Bodenfeuchtigkeit	Analog (Sensor)
Vibrationsmotor	Digital (Aktor)
Joystick	**Analog - Sonderfall**
Berührungssensor	Digital (Sensor)
Lautstärkesensor	Analog (Sensor)
Servomotor	**Digital (Aktor) - Sonderfall**
Temperatursensor	Analog (Sensor)
Sliding Potentiometer	Analog (Sensor)
Bewegungssensor	**Digital – Sonderfall**

Auslesen von Sensorwerten im Allgemeinen

Sensorwerte können zum Beispiel folgendermaßen ausgelesen werden:

1. Erstellen Sie in der Kategorie *Variablen* eine Variable *Sensorwert* .
2. Die Variable erscheint in der Kategorie *Variablen*. Klicken Sie das Feld neben dem Namen an.

3. Weisen Sie dieser den Wert des Pins zu (je nach Typ und Steckplatz des Sensors). z.B

Analog (an Steckplatz A0 angeschlossen)	Digital (an Steckplatz D7 bzw. Pin 7 angeschlossen)
	

Aktoren steuern

Aktoren können mithilfe von digitalen Pins angesteuert werden. Hierfür verbinden Sie den Aktor gemäß der Anleitung und können diesen nun folgendermaßen steuern:

Aktor einschalten (z.B. an Pin 13 angeschlossen) :

Aktor ausschalten (z.B. an Pin 13 angeschlossen):

Beispiel LED

Eine LED kann mithilfe eines digitalen Pins angesteuert werden. Verbinden Sie hierzu das lange Beinchen der LED mit z.B. dem digitalen Pin 13 und das kurze Beinchen mit dem GND.

LED an

LED aus

Verwenden Sie den Block *warte* aus der Kategorie *steuern*, um die Lampe 1 Sekunde leuchten zu lassen, z.B. :

ACHTUNG: In dem Workshop verwenden wir spezielle LEDs, die für 5V Betriebsspannung ausgelegt sind. Im Allgemeinen darf man LEDs nur mit einer deutlich geringeren Spannung betreiben. In den meisten Fällen sollten Sie deshalb einen Vorwiderstand (von meistens 220 Ohm) verwenden, um ein Explodieren der LEDs zu vermeiden. Dies brauchen Sie bei diesen LEDs nicht zu beachten.

Praxistipp: Funktioniert die LED nicht? Überprüfen Sie, ob Sie die LED richtig herum eingesteckt haben oder ob Sie die digitalen Pins 0-6 verwendet haben. Diese funktionieren manchmal nicht wie gewünscht.

Vibrationsmotor

Die Steuerung funktioniert wie bei LEDs durch das Setzen eines digitalen Pins auf HIGH oder LOW.

Sonderfall Joystick

Der Joystick besitzt **zwei** analoge Pins. Dieses mal ist das weiße Kabel also nicht **n**ot **c**onnected, sondern ein analoges Signal, das ausgelesen werden kann. Stecken Sie diese an zwei verschiedene analoge Pins des Arduinos und lesen Sie folgendermaßen (Sensor an Pins 0 und 1 angeschlossen) die x- und y-Koordinaten aus:

Überprüfen Sie, welche Werte der Sensor in der Nullstellung liefert. Diese können schwanken. Ausgehend hiervon können Sie durch Anpassen der Werte beispielsweise eine Steuerung des Pfeils in Snap! realisieren. Ein Ansatz für die Umrechnung könnte **gehe** (x-500)/20 **Schritte** sein.

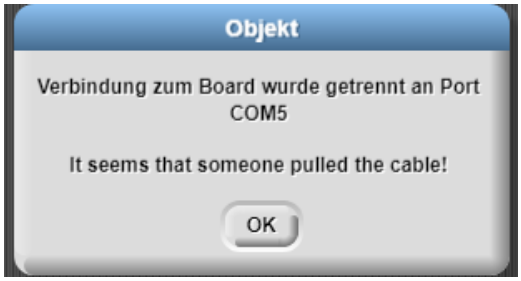
Hinweis für das Grove-Shield: Haben wir bisher einen Sensor an D7 angeschlossen, konnten wir ihn am Pin 7 auslesen. Schließen wir diesen Joystick nun ebenfalls an D7 an, stellt sich die Frage, welche zwei Pins nun ausgelesen werden sollen. Ein kurzer Blick auf die Beschriftung der Platine hilft dabei. Das gelbe Kabel ist mit Pin 7 verbunden und das weiße Kabel mit Pin 6.

Servomotor

Verkabeln Sie den Servomotor folgendermaßen mit dem Arduino:

- rotes Kabel an 5V
- braunes Kabel an GND
- gelbes Kabel an einen digitalen Anschluss (z.B. 8)

In den meisten Fällen erhalten Sie folgende Fehlermeldungen:



Klicken Sie in diesem Fall einfach auf ok und anschließend erneut auf **Arduino verbinden**. Sie können den Servomotor nun beispielsweise folgendermaßen steuern:

Mögliche Werte für die Stellung des Servomotors sind 0 bis 180. Ein durchgängiges Drehen, um beispielsweise ein Auto zum Fahren zu bringen, ist nicht möglich.

Temperatursensor

Die Verkabelung des analogen Grove-Temperatur-Sensors erfolgt wie oben beschrieben. Das Umrechnen des analogen Wertes in eine Temperatur ist leider sehr umständlich, kann aber in dem Wiki des Sensors unter http://wiki.seeedstudio.com/Grove-Temperature_Sensor_V1.2/ nachgelesen werden:

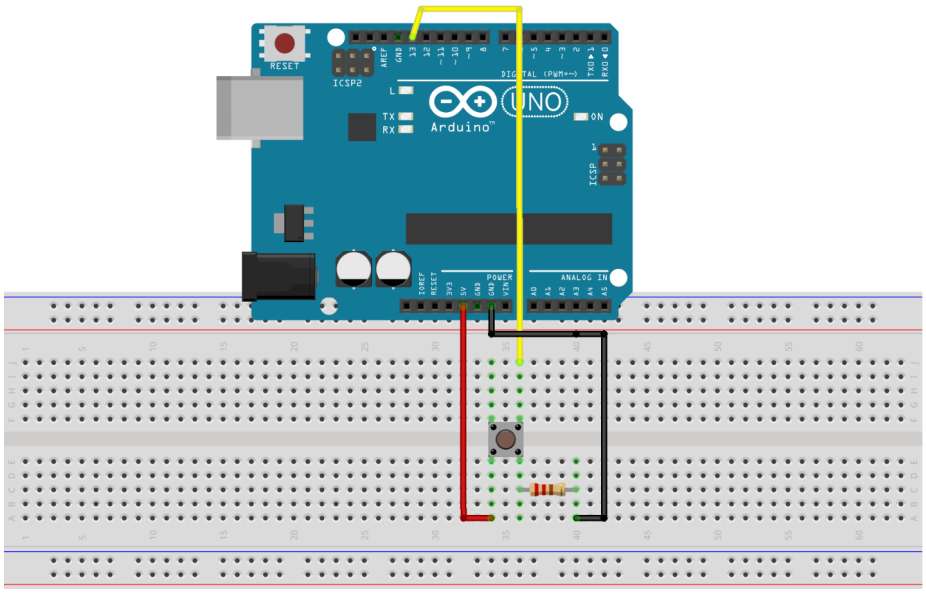
*Seien B=4275, R0=100000 und a der Sensorwert. Und setzen Sie R= (1023/a) -1) * R0. Dann gilt für die Temperatur:*

temperatur= 1.0 / (log(R/R0) / B + 1/298.15) - 273.15

Hinweis: Das Auslesen von Temperaturwerten muss nicht immer so kompliziert sein. Es gibt auch Sensoren, die die Temperatur direkt als Wert ausgeben. Diese funktionieren aber meist mit dem Signalprotokoll I2C, das für den Einstieg nicht geeignet ist.

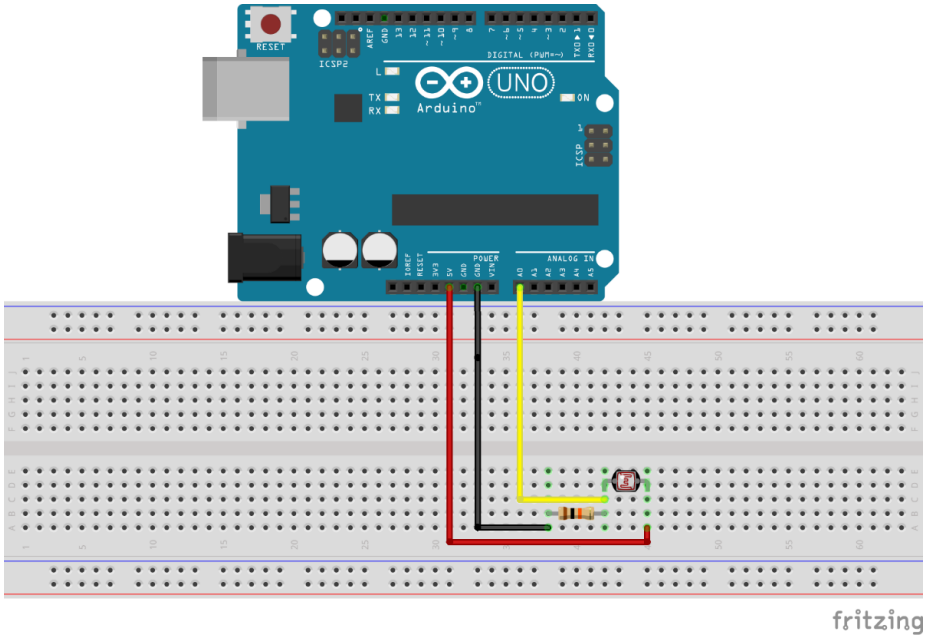
Button

Statt teure Grove-Buttons zu kaufen, kann man auch einen günstigen Button mit 4 Beinen verwenden. Verkabeln Sie den Arduino wie in folgender Abbildung und lesen Sie den digitalen Wert (1 für gedrückt, 0 für nicht gedrückt) des Sensors an Pin 12 aus. Den Widerstand finden Sie bei den Fotowiderständen.



Helligkeitssensor

Verwenden Sie folgende Verkabelung für den Fotowiderstand:

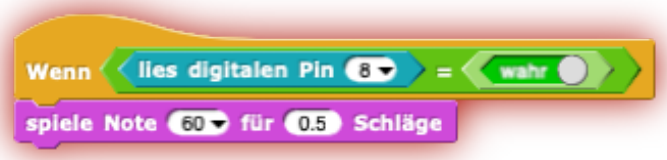


Der analoge Helligkeitswert kann über den Pin A0 ausgelesen werden.

Bewegungsmelder

Bei diesem Bewegungsmelder handelt es sich um einen **digitalen Sensor** bei dem eine 1 ausgelesen werden kann, sobald eine Bewegung registriert wird. Legen Sie den Sensor so, dass sich die beiden Drehpotentiometer oben befinden. An diesen Potentiometern kann die Sensitivität (links drehen bis 3 Meter , rechts drehen bis 7m Erkennung möglich) und die Dauer des HIGH-Signals bei einer Bewegungsfeststellung (links 5 s, rechts bis 300 s) eingestellt werden. Die Versorgungsspannung (VCC) beträgt **5V**, das digitale Signal kann am mittleren Pin (OUT) ausgelesen werden.

Beispielprogramm:



Der OUT-Pin wurde mit dem digitalen Pin 8 verbunden. Sobald eine Bewegung registriert wird, wird ein Ton abgespielt.