

Jay-D coding – Erste Programmierschritte

Einleitung

Installation

Herzlich Willkommen bei der Programmieranleitung zu Jay-D

Danke dass Du unsere Arbeit unterstützt! Wir hoffen, dass Du mit dieser Anleitung sehr viel über die Programmierung von Jay-D lernen wirst.

Zur Programmierung Deines neu gebauten Mischpultes werden wir **CircuitBlocks** verwenden.

Aber was ist CircuitBlocks?

CircuitBlocks ist eine maßgeschneiderte App, die wir zur Programmierung entwickelt haben.

Mit ihrer grafischen Oberfläche erleichtert sie Dir den Einstieg indem das Programm aus vorbereiteten Bausteinen zusammengestellt wird.

Installation

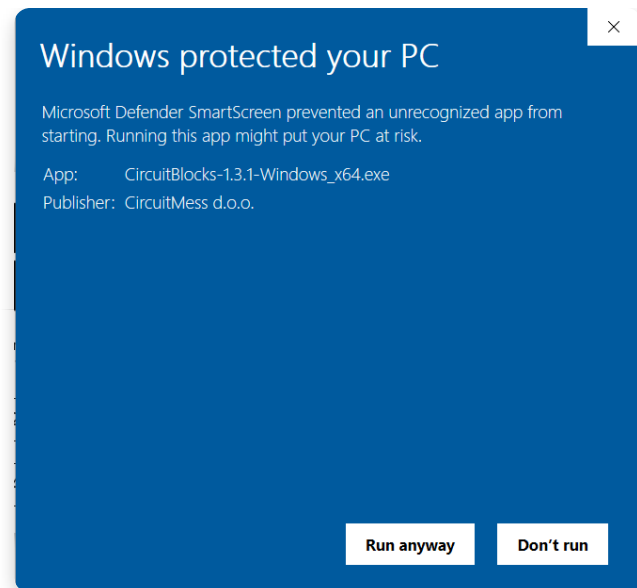
CircuitBlocks gibt es zur Zeit für Windows, Linux und Mac OS.

Du hast einen Computer mit Windows Betriebssystem

1. **Öffne die [CircuitBlocks Download Webseite](#)**
2. **Lade die aktuelle Version für Windows herunter** – Prüfe vor der Installation, ob Dein Computer die 32- oder 64-bit Version benötigt. Klicke dazu mit der rechten Maustaste auf den Windows-Button in der Start-Leiste. Es öffnet sich dann ein Menü in dem Du bitte auf „System“ klickst. Nun öffnet sich ein Fenster in dem Du unter „Systemtyp“ siehst Du, welche Version Du brauchst.

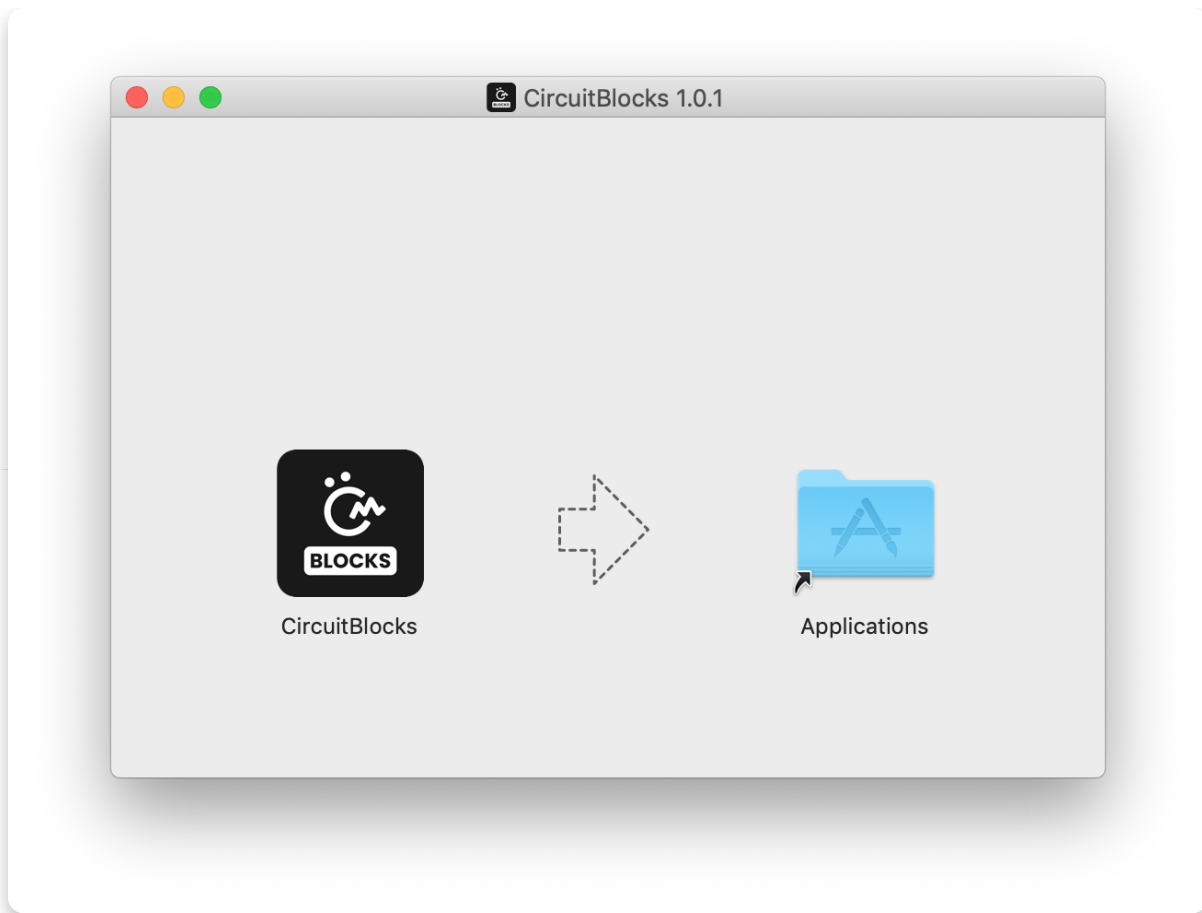
Klicke bitte auf „Weitere Informationen“ um mit der Installation fortzufahren. Danach erscheint ein Auswahlfeld.

Hier klickst Du bitte auf „Trotzdem ausführen“.



Du hast einen Mac Computer

1. Öffne die [CircuitBlocks Download Webseite](#)
2. **Lade die die aktuelle Version für Mac OS herunter** (die Datei "CircuitBlocks-1.0.5-Mac.dmg" oder ähnlich sollte heruntergeladen werden)
3. Verschiebe die Datei in den Ordner „Anwendungen (Applications)“
4. CircuitBlocks wird nun automatisch installiert.



Du hast einen Linux Computer

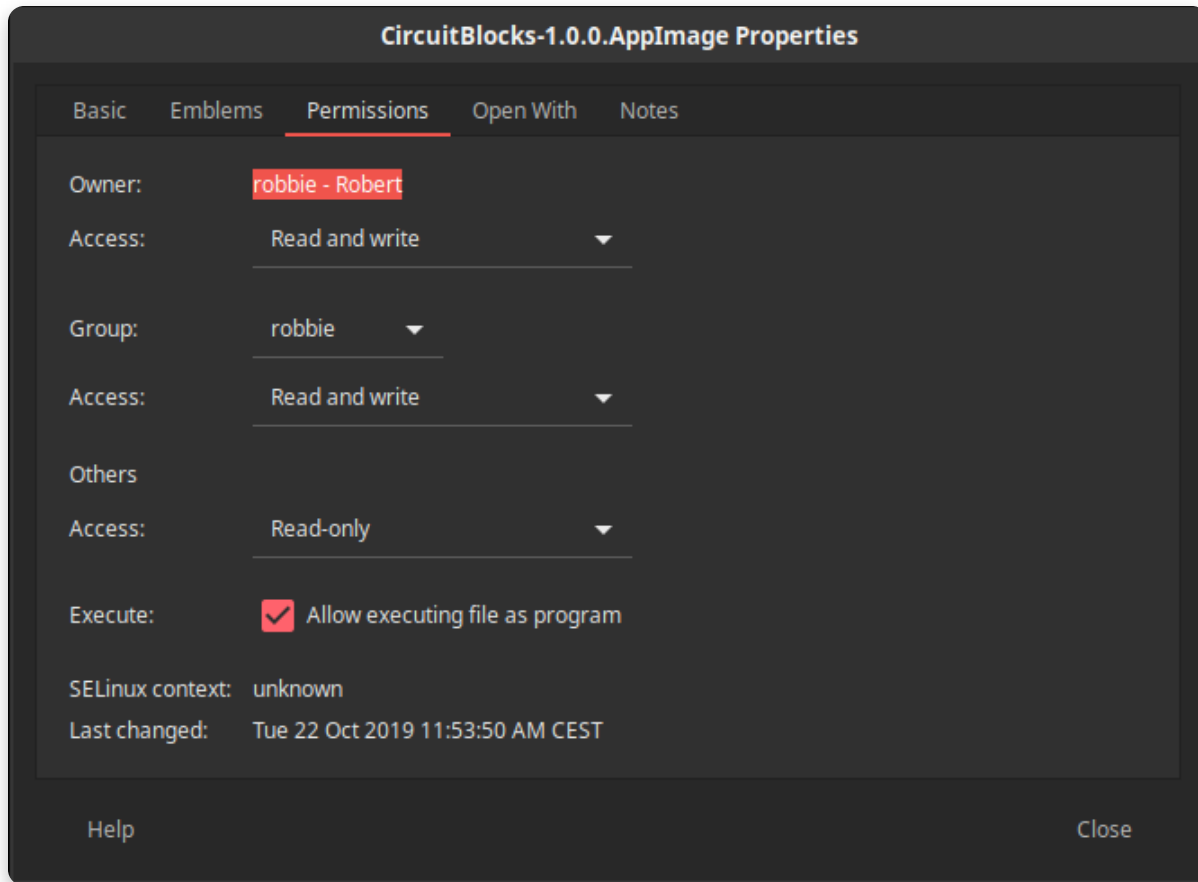
Es gibt zwei Wege um CircuitBlocks unter Linux zu installieren

1. Öffne die [CircuitBlocks Download Webseite](#)
2. **Drücke auf den Knopf: "Linux 64-bit" Download.**
3. Mache eine Doppelklick auf die heruntergeladene Datei um die Installation zu starten (Ubuntu)
der
Öffne ein Terminalfenster und gib folgendes ein: `sudo dpkg -i <path to the downloaded file .deb>` (andere Linux Distributionen)
4. CircuitBlocks wird nun automatisch installiert und legt danach eine Verknüpfung auf dem Desktop an

Fertige Image-Datei (ApplImage):

1. Öffne die [CircuitBlocks Download Webseite](#)
2. **Klicke auf die Schaltfläche "Linux ApplImage" Download**
3. Mache einen Rechtsklick auf die Datei und wähle **'Eigenschaften'**

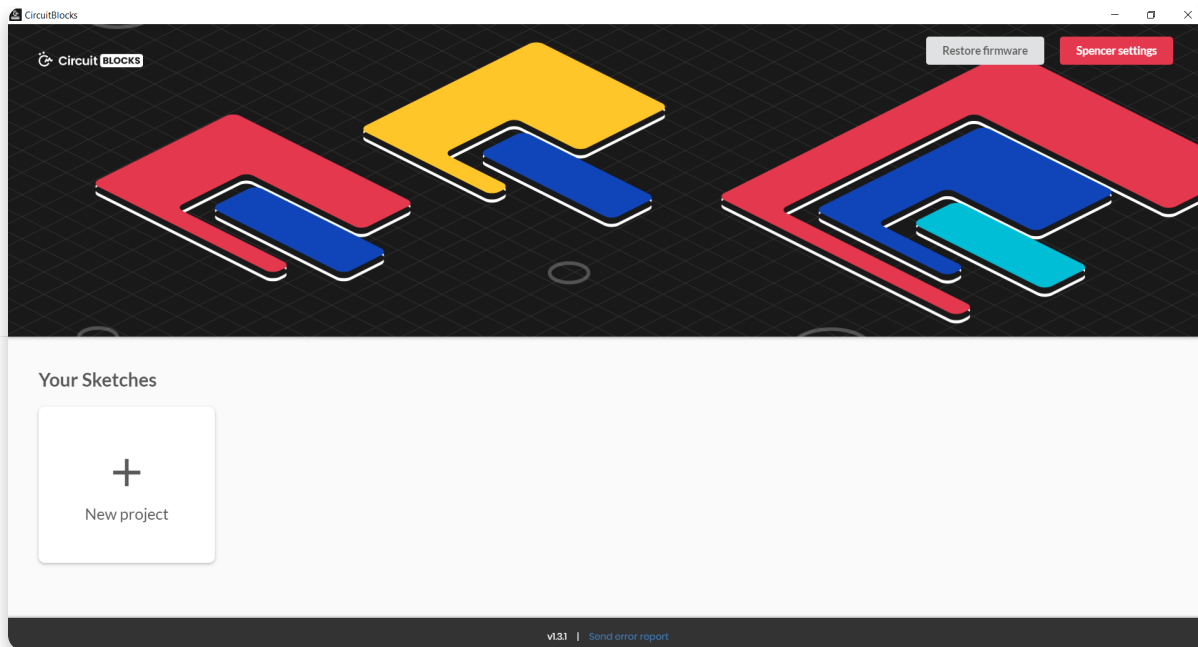
4. Gehe zu '**Berechtigungen**' und klicke auf '**Datei als Programm ausführen**'
5. Mache nochmals einen Doppelklick auf die Datei und die Installation startet nun automatisch.



Solltest Du bei der Installation Probleme haben und Hilfe benötigen, schreibe uns bitte eine Email an contact@circuitmess.com Beschreibe uns bitte Dein Problem und hänge wenn möglich ein Bildschirmfoto an.

Die Grundlagen

Benutzeroberfläche



Wenn Du CircuitBlocks startest, öffnet sich ein Fenster das wie dieses aussieht.

Alles ist einfach gehalten, **Ein neues Projekt (wir nennen sie auch „Sketches“)** legst Du mit einem Klick auf den Knopf **„New Project“** an.

Gespeicherte Sketches erscheinen rechts vom Knopf. Du kannst sie dort jederzeit öffnen.

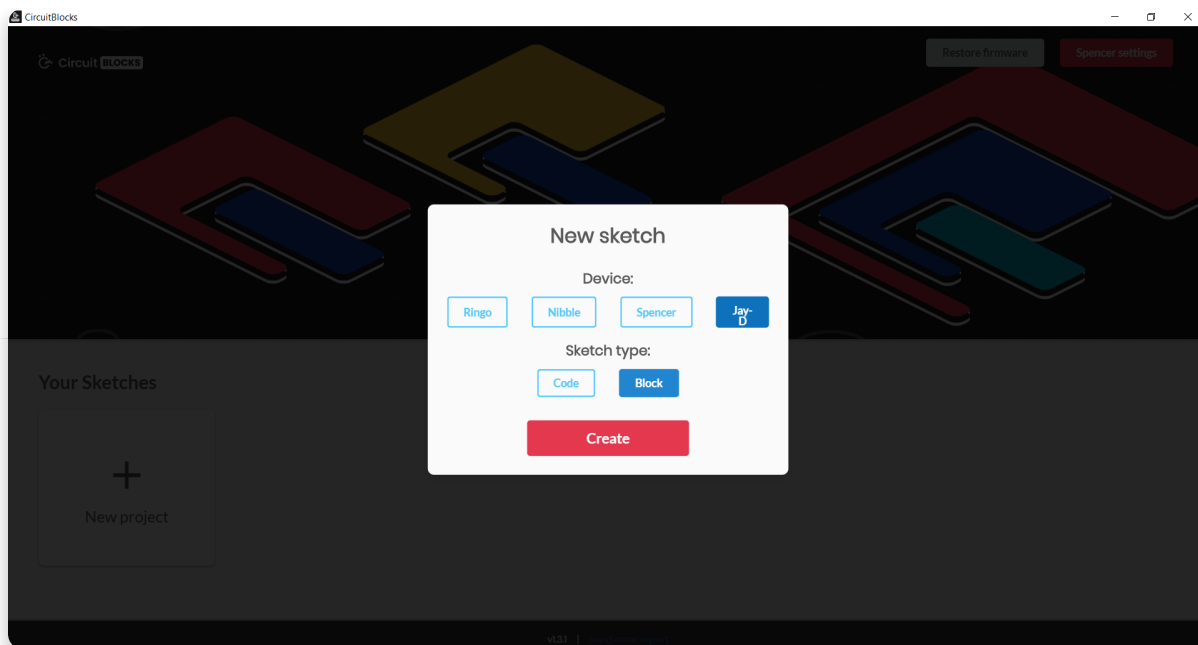
Wenn Du bei der Benutzung von CircuitBlocks auf irgendein Problem stößt, welches Du nicht alleine lösen kannst, drück bitte auf den Knopf **„Send error report“** am unteren Bildschirmrand. Dir wird dann eine Fehlermeldenummer angezeigt. Schreibe uns bitte eine Email und teile uns darin auch die Fehlermeldenummer (error report number) mit. Sende Deine Email bitte an contact@circuitmess.com und wir werden versuchen Dir so schnell wie möglich zu helfen.

Ein neues Projekt (sketch) anlegen

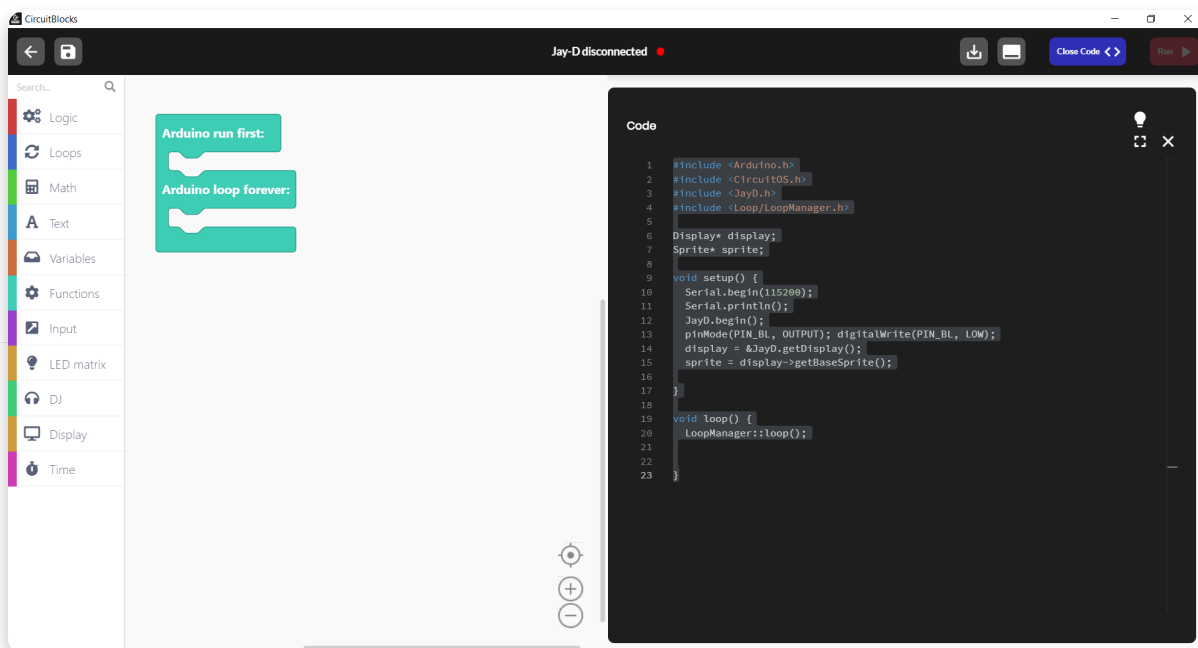
Drücke auf den großen Knopf „New Project“.

Nun kannst Du das Gerät (device) und den Sketch Typ auswählen.

Als Gerät (device) wähle bitte **Jay-D** und für den Sketch Typ **Block** aus.
Klicke nun auf die Schaltfläche „create“ (also: erstellen).



Es wird nun eine Oberfläche angezeigt, die so ähnlich wie diese aussieht:



Am oberen Bildschirmrand befindet sich eine **Werkzeugleiste (toolbar)** mit einigen Knöpfen.

Die Block-Auswahlleiste (block selection bar) befindet sich am linken Bildschirmrand. Von dort kannst Du Blöcke mit der Maus in den Zeichenbereich, in der Mitte des Bildschirms, schieben.

Im Zeichenbereich „zeichnest“ Du mit den bunten Blöcken Dein Programm.

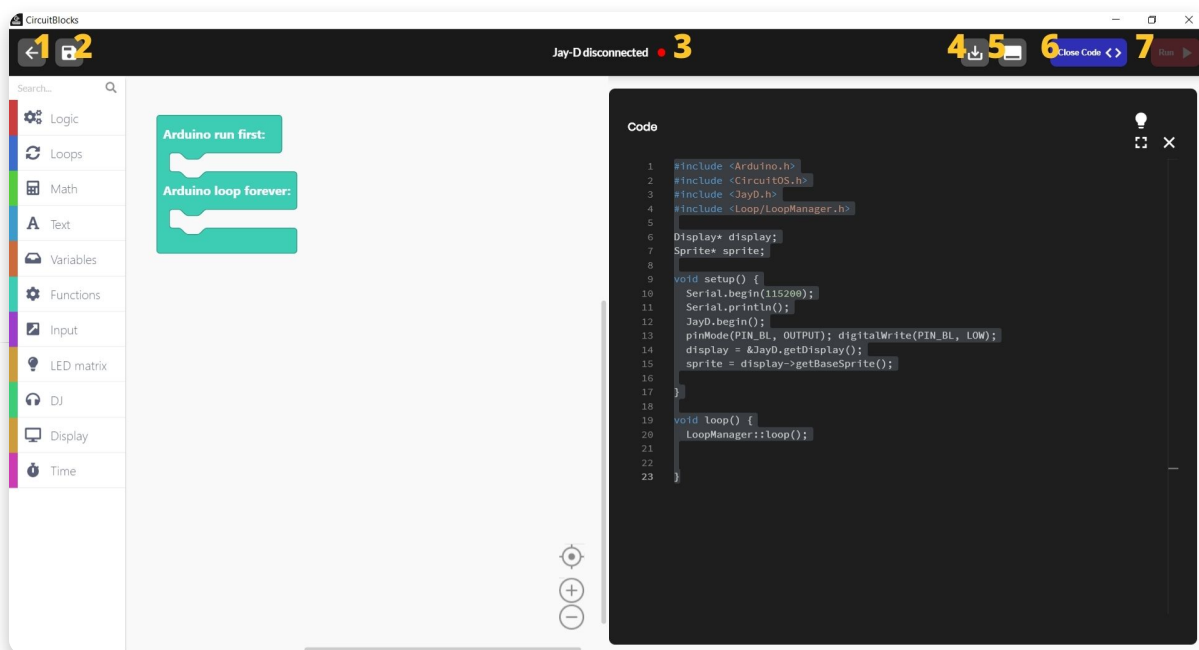
So bald Du einen Block im Zeichenbereich „fallen lässt“, erscheint am rechten Bildschirmrand dann, wie von Geisterhand, der zugehörige, in **C++ geschriebene**,

Code.

C++ ist eine der weitverbreiteten Programmiersprachen. Leider ist sie nicht so einfach zu verstehen, wenn Du noch nie vorher etwas programmiert hast.

Darum gibt es CircuitBlocks – hier steht jeder Block für ein Stück Code. Beim „malen“ Deines Programms kannst du aber auch gleichzeitig sehen, welcher C++ Code mit jedem Block hinzukommt. Wenn Du etwas Erfahrung mit den Blöcken gesammelt hast, kannst Du schon bekannte Befehle direkt in C++ eingeben, ohne die Blöcke zu nutzen.

Die Werkzeugleiste



Hier ist eine kurze Übersicht über die einzelnen Befehle der Werkzeugleiste:

1. **Zurück ins Hauptmenü** – Du gelangst ohne speichern zurück in das Hauptmenü.
2. **Speichern/Speichern als** – Speichere Deinen Sketch von Zeit zu Zeit und auch, bevor Du CircuitMess beendest.
3. **Jay-D Verbindungsstatus** – Der rote Punkt wird grün, wenn die USB-Verbindung zu Deinem Jay-D erfolgreich hergestellt wurde.
4. **Code als Binärdatei speichern** – Diese Schaltfläche speichert Deinen Code als Binärdatei. Du benötigst Diese Funktion jetzt noch nicht.
5. **Serieller Monitor** – Dieser Knopf öffnet ein Fenster, das wir den „Seriellen Monitor“ nennen. „Seriell“ ist ein Spitzname für den Datenaustausch

zwischen Jay-D und dem Computer. Später wirst Du in diesem Fenster die Nachrichten sehen können, die Jay-D an Deinen Computer über die USB-Schnittstelle sendet.

6. **Code-Fenster schließen** - Mit diesem Knopf kannst Du das Code-Fenster schließen und wieder öffnen. Das ist sehr hilfreich, wenn Du mehr Platz auf dem Bildschirm für die bunten Blöcke benötigst.
7. **Starten** - Dieser Knopf übersetzt Dein mit CircuitBlocks gemaltes Programm in Maschinencode. Den kann Jay-D lesen und verstehen (beep boop beep boop 1011100101). Ist der Code fertig übersetzt, wird er über die USB-Verbindung zu Jay-D gesendet.

Das Code-Fenster

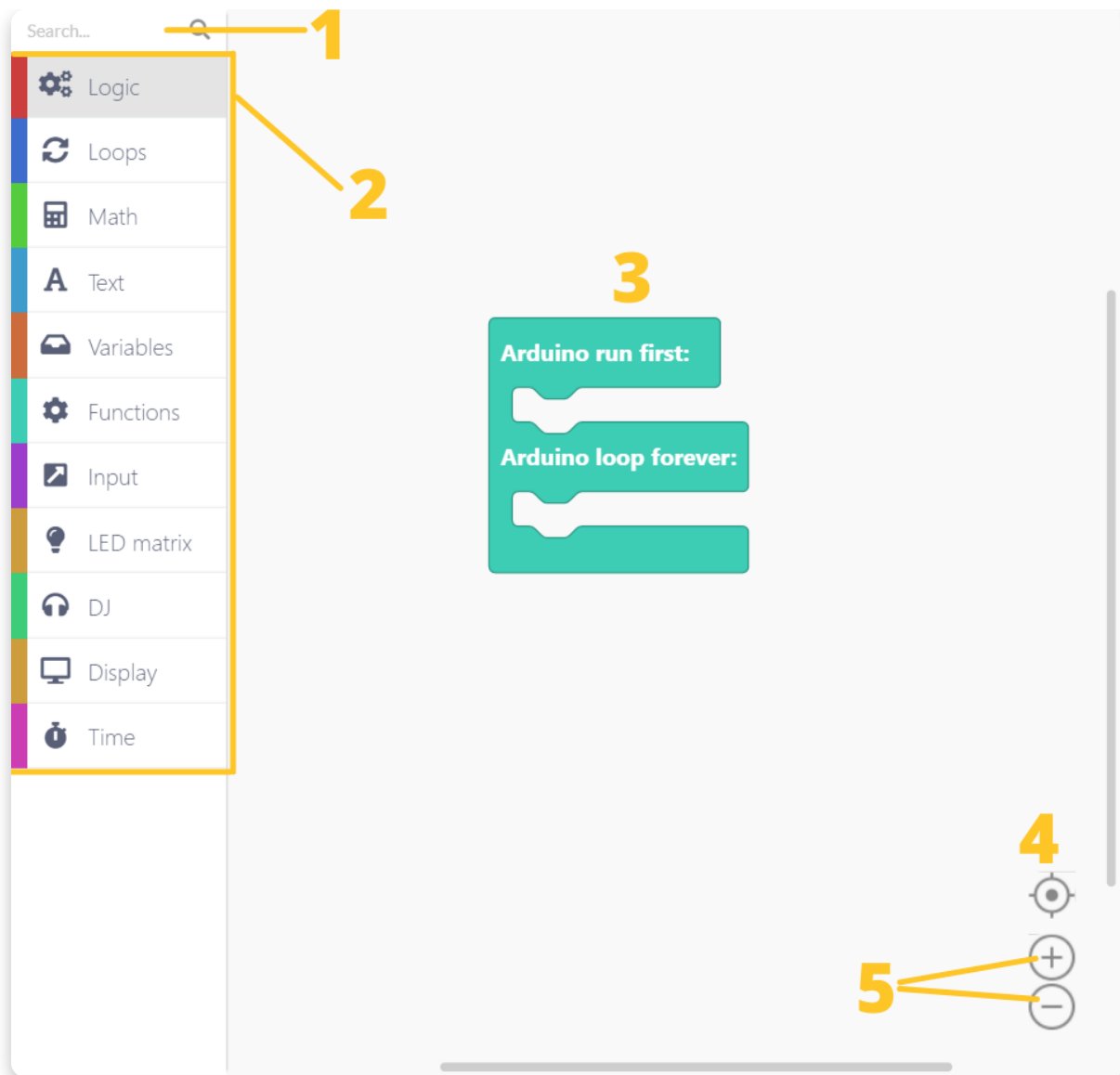
```
Code

1  #include <Arduino.h>
2  #include <CircuitOS.h>
3  #include <JayD.h>
4  #include <Loop/LoopManager.h>
5
6  Display* display;
7  Sprite* sprite;
8
9  void setup() {
10     Serial.begin(115200);
11     Serial.println();
12     JayD.begin();
13     pinMode(PIN_BL, OUTPUT); digitalWrite(PIN_BL, LOW);
14     display = &JayD.getDisplay();
15     sprite = display->getBaseSprite();
16
17 }
18
19 void loop() {
20     LoopManager::loop();
21
22
23 }
```

Das so genannte „Code-Fenster“ ist in folgende Abschnitte aufgeteilt:

1. **Code Bereich** – Hier erscheint der zugehörige Code für die bunten Blöcke, die Du in den Zeichenbereich gelegt hast. Du wirst feststellen, dass einige Textabschnitte in lustigen Farben dargestellt werden. Die Programmierer nennen das „Syntaxhervorhebung“. Einfach gesagt, es werden die gleichartige Bestandteile des Codes in der selben Farbe angezeigt. Das macht es den Programmierern einfacher den Code zu verstehen.
2. **Helle / Dunkle Darstellung** – Mit diesem Knopf kannst Du die Hintergrundfarbe und die Textfarbe umschalten.
3. **Erweitern** – Erweitert den Code Bereich auf die ganze Bildschirmfläche. Klickst Du erneut so wird der Code Bereich wieder verkleinert.
4. **Schließen** – Schließt das Fenster für den Code Bereich. Tipp: Das kannst Du ebenfalls mit dem Knopf „Close Code“ in der Werkzeugleiste machen. Zum erneuten Öffnen kannst Du dann in der Werkzeugleiste auf „Open Code“ klicken.

Der Zeichenbrett



Hier entsteht die Magie.

Das Zeichenbrett unterteilt sich in folgende Abschnitte

1. **Suchleiste** – Gib hier den Namen des Blockes ein, den Du suchst.
2. **Component selector** – Die einzelnen Blöcke sind in Kategorien zusammengefasst. Jede hat eine eigene Farbkennzeichnung am linken Bildschirmrand.
3. **Zeichenbereich** – Hierher ziehst Du die Blöcke, die du benötigst und lässt sie fallen. Daraus wird dann der Code erzeugt. Ganz simpel.
4. **Zentrierhilfe** – Wenn Du Dich beim Scrollen verirrt hast, gelangst Du mit diesem Knopf in die Mitte Deiner ausgewählten Blöcke.
5. **Zoomfunktion** – Hiermit vergrößerst bzw. verkleinerst Du den Ausschnitt des dargestellten Zeichenbereiches.

Schritt für Schritt

Lass uns mit den LEDs beginnen

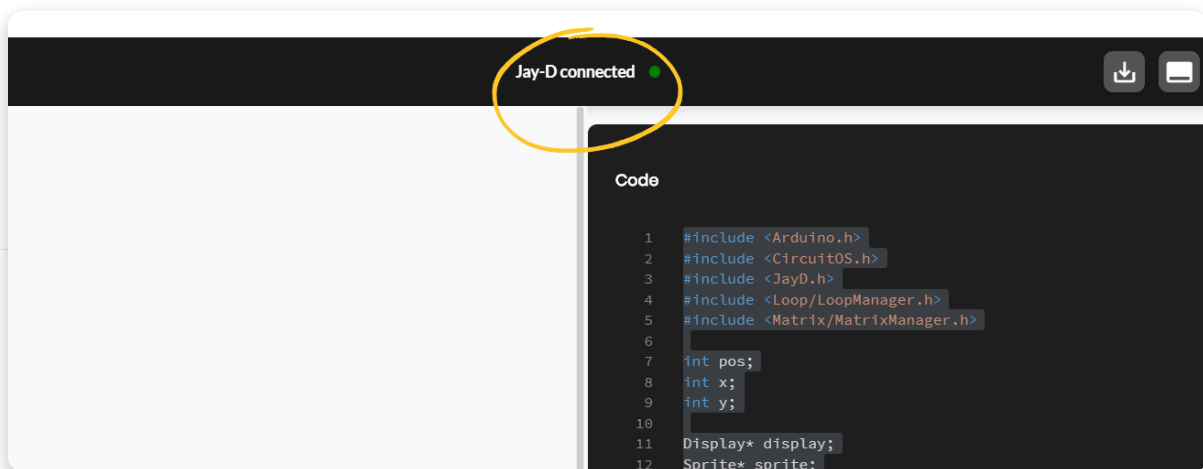
Bist Du bereit zu programmieren?

Als erstes startest Du CircuitBlocks und verbindest Jay-D mit einem USB-Anschluss Deines Computers.





CircuitBlocks sollte jetzt oben auf dem Bildschirm die Meldung "Jay-D connected" anzeigen.



Wenn CircuitBlocks Deinen Jay-D nicht erkennt, prüfe bitte ob das USB Kabel ordnungsgemäß eingesteckt ist und Dein USB Anschluss auch funktioniert. Klappt es dann immer noch nicht, ist vermutlich bei der Treiber-Installation auf Deinem Computer etwas schiefgelaufen. Treiber sind diese kleine Programme, die Deinem Computer helfen, mit angeschlossenen Geräten zu sprechen. Leider verhalten sich Treiber manchmal etwas seltsam.

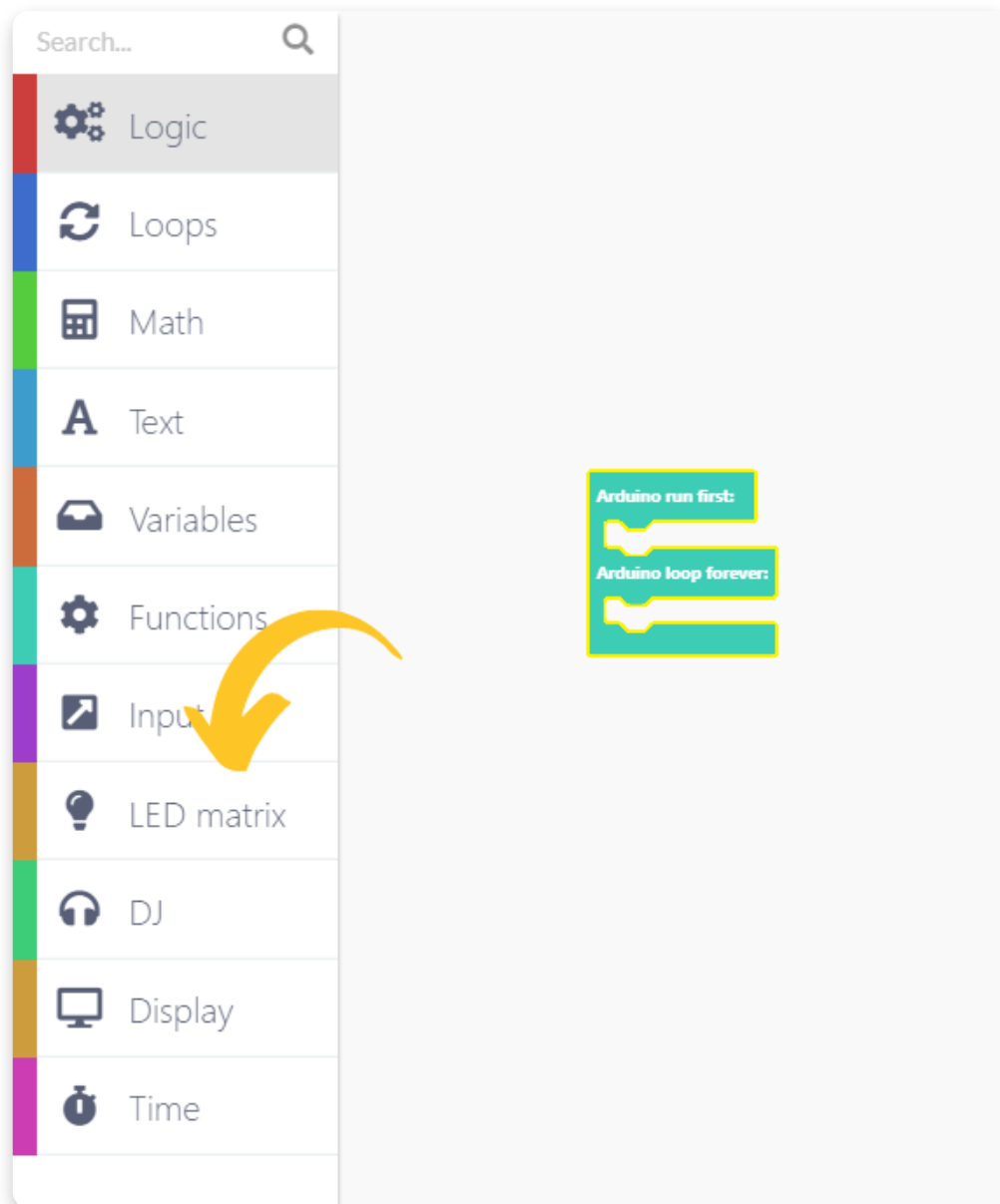
Setze Dich in diesem Fall bitte mit uns per Email in Verbindung. Unsere Email

Adresse lautet: contact@circuitmess.com.

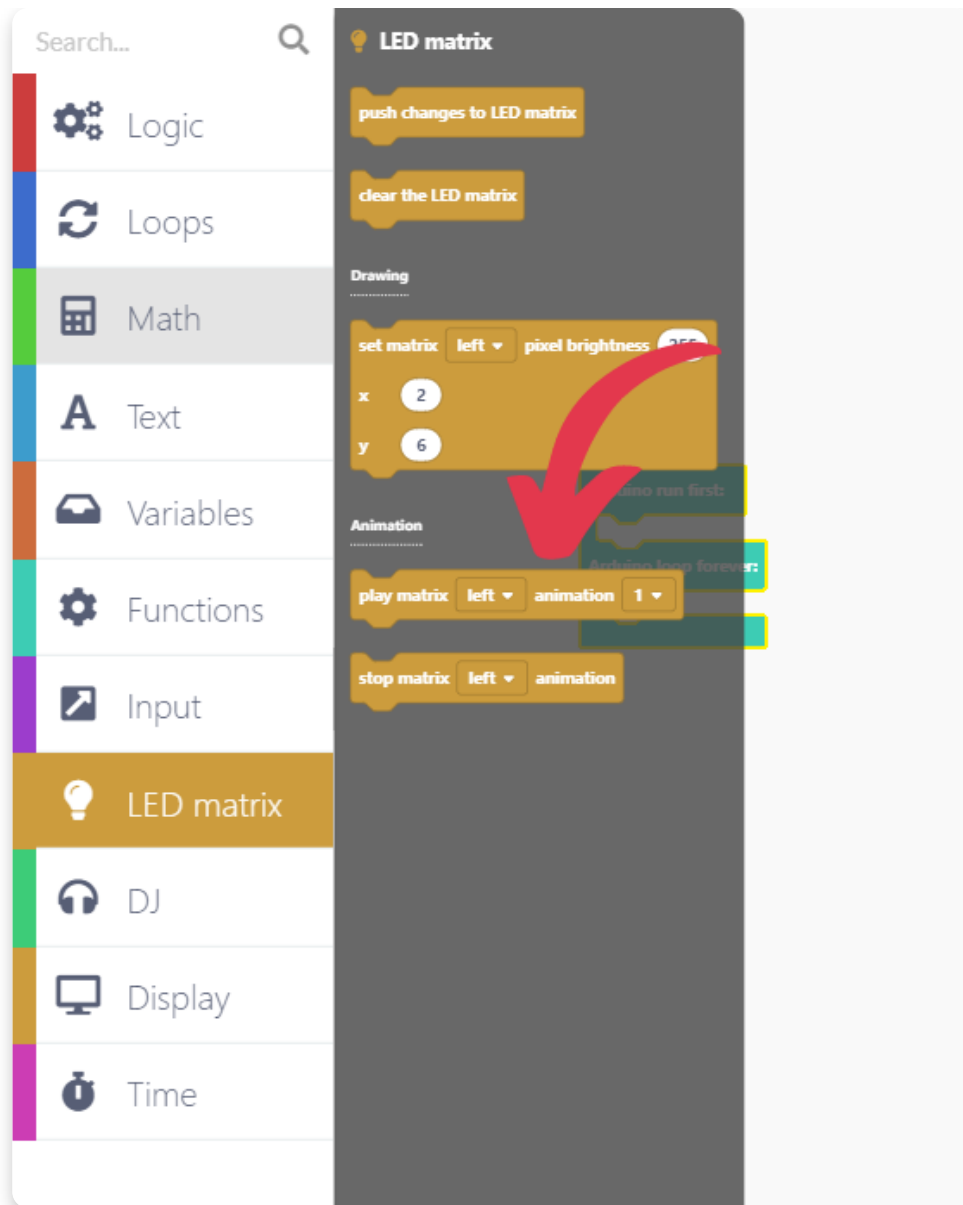
Lass uns eine Animation auf dem LED-Feld programmieren

Öffne die Kategorie „LED matrix“ auf der linken Bildschirmseite.

Diese Kategorie enthält Blöcke, die für die Animation des LED-Matrix-Displays (wir nennen es auch LED-Matrix) benötigt werden.



Suche den Block "*Play matrix left animation 1*".

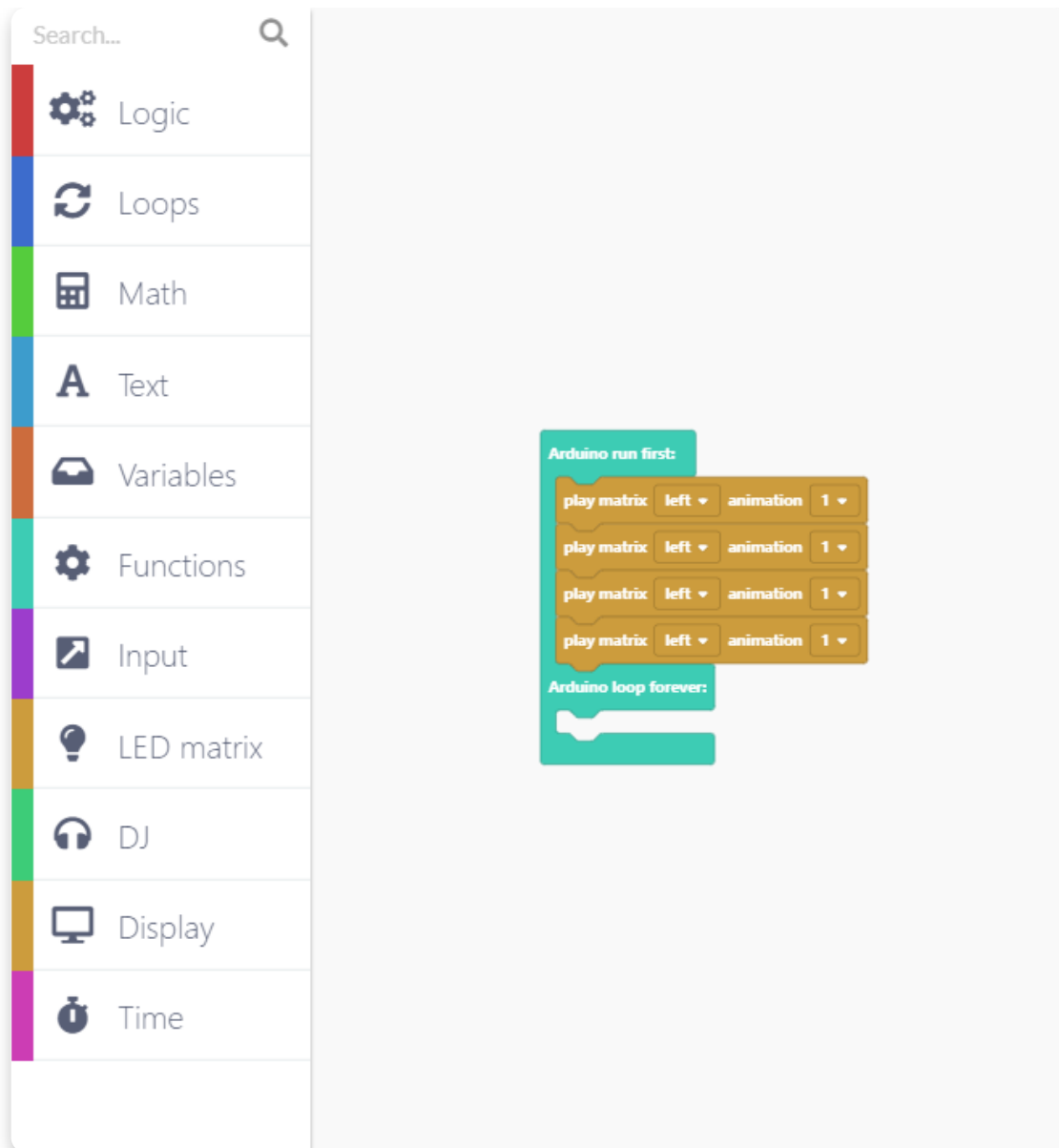


Zieh ihn über den türkisgrünen Block der sich bereits im Zeichenbereich befindet.

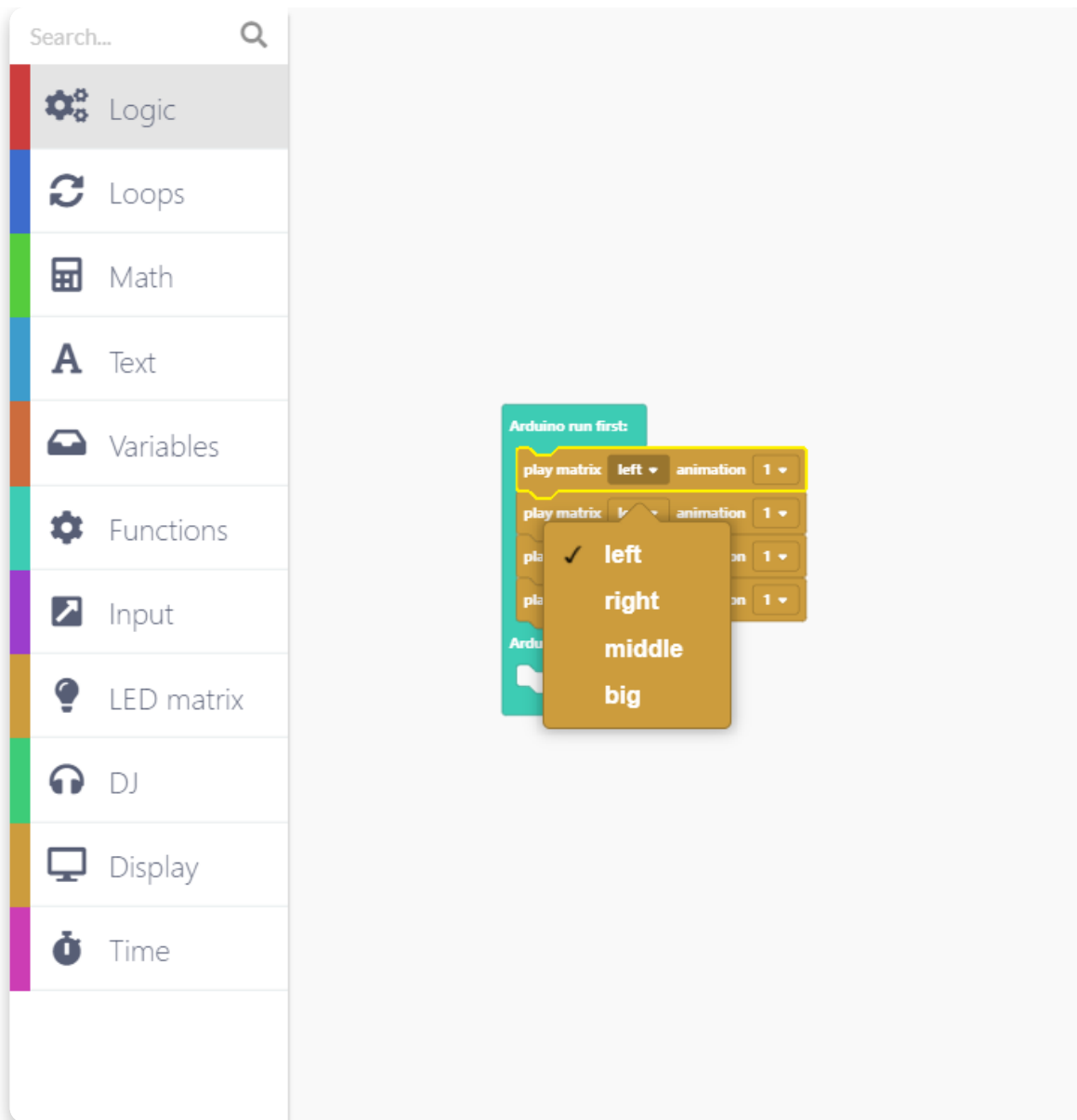
Lass ihn dann in der Lücke fallen über der "Arduino run first" geschrieben steht.

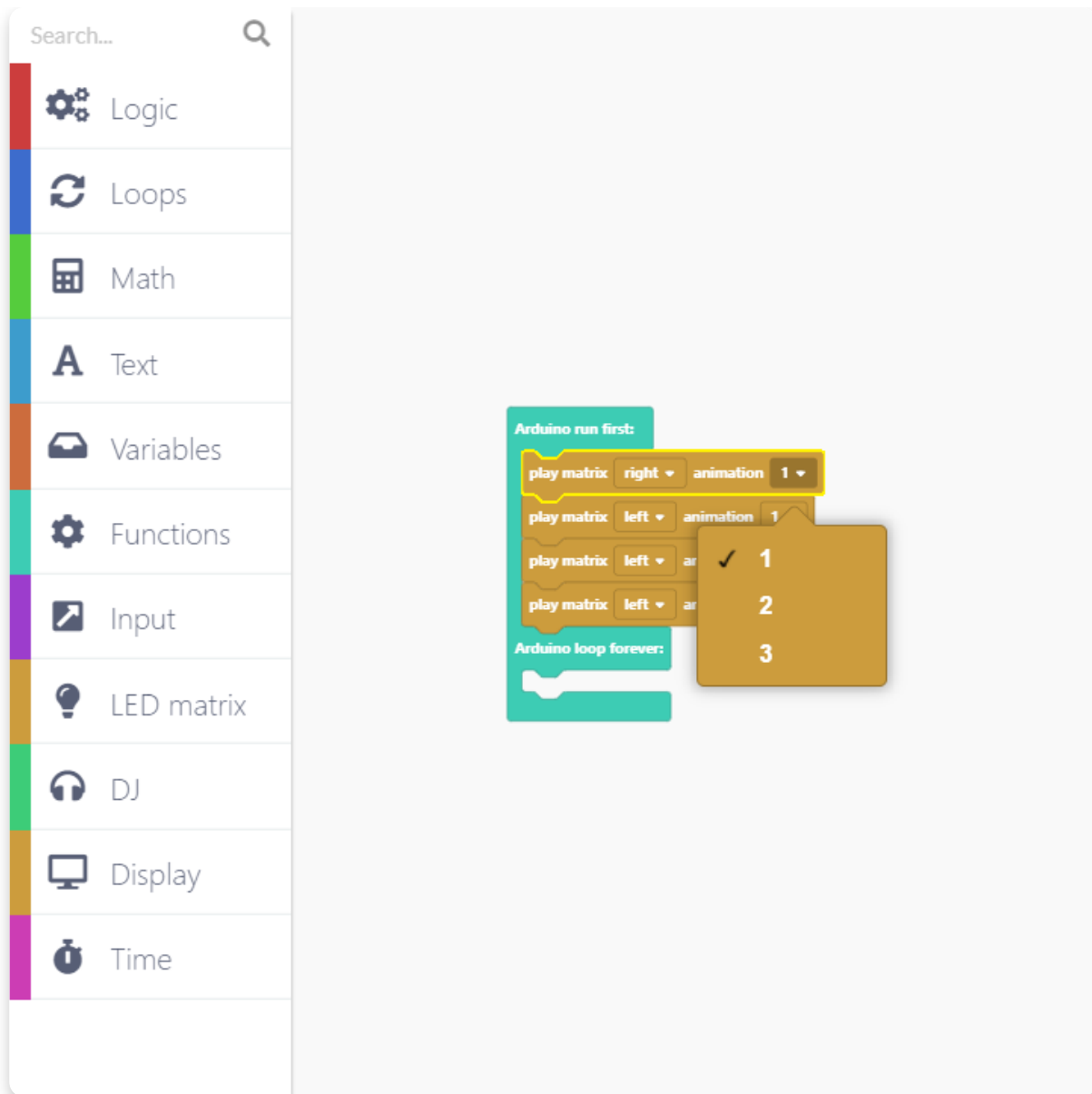
Das machen wir deshalb, weil die Animation dann nur einmal beim Start von Jay-D abgespielt wird.

Da Jay-D insgesamt vier LED-Matrix-Displays besitzt, wiederholen wir diesen Schritt noch dreimal. Anschließend sollte Dein Zeichenbereich in etwa so aussehen, wie auf dem untenstehenden Foto.



Jetzt legen wir fest, welcher Block für welche LED-Matrix zuständig sein soll. Klicke dazu bitte auf den kleinen Pfeil eines „Play Animation Blocks“ und es erscheint ein kleines Aufklappmenü.

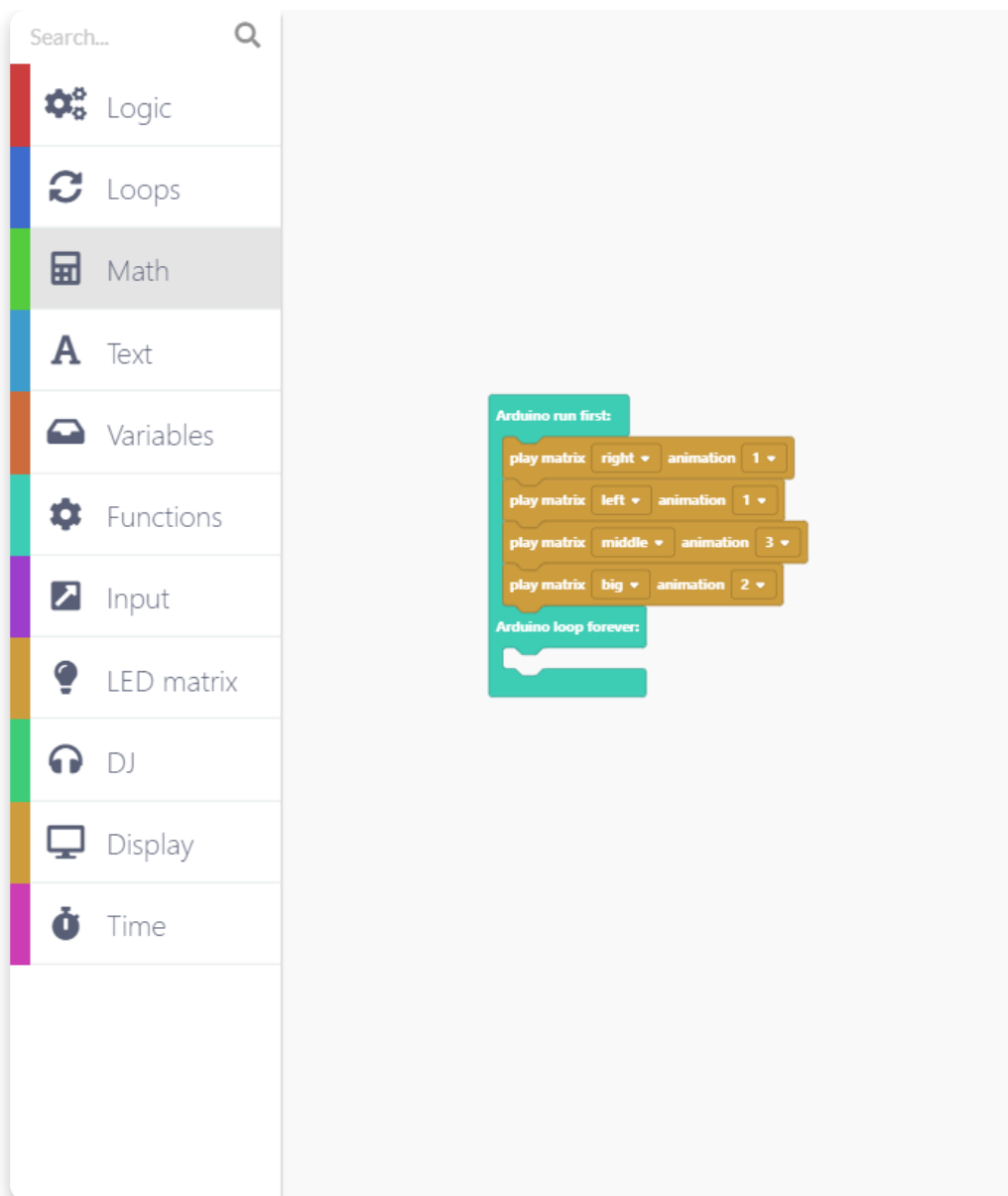




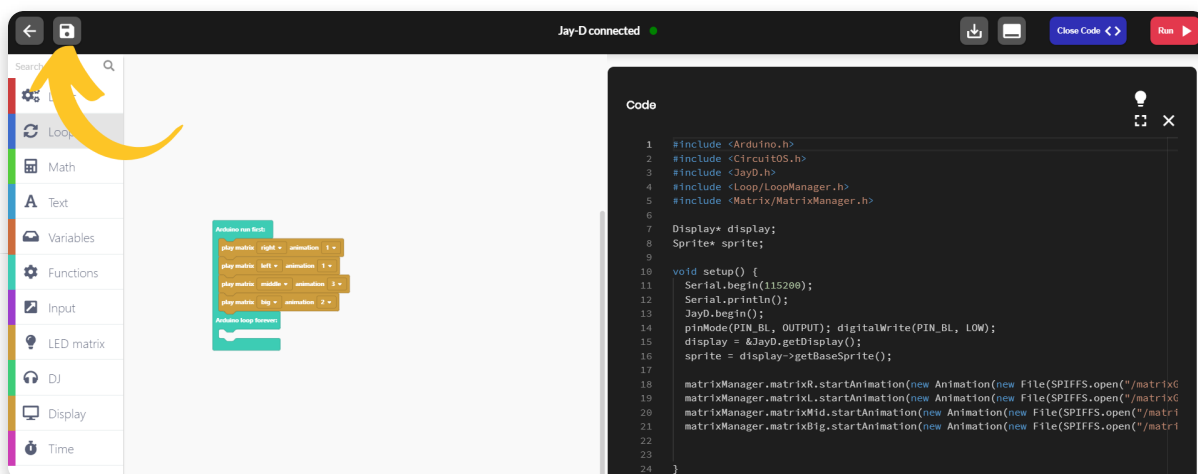
Hier kannst Du auswählen, welche Matrix gesteuert werden soll (**left, right, middle, big**). Außerdem kannst Du eine von drei verfügbaren Animationen wählen „animations (**1,2,3**)“.

Diese Animationen werden aus animierten GIF Bildern erzeugt, die sich im Flash-Speicher Deines Jay-D befinden.

Wir haben diese Auswahl getroffen:

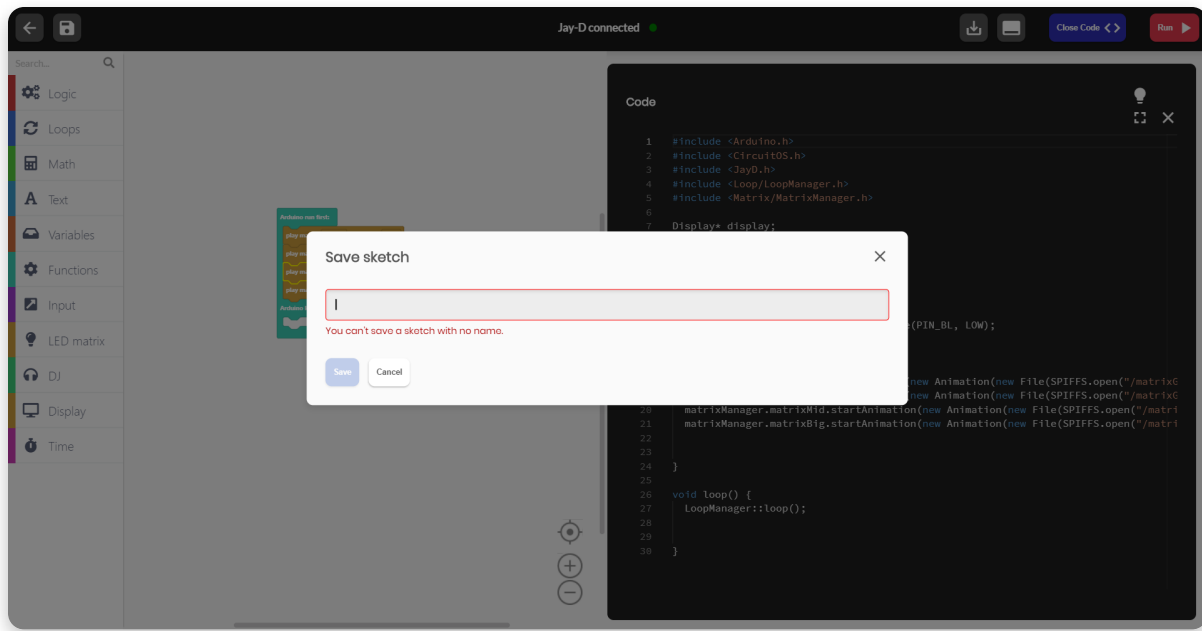


Jetzt ist ein guter Zeitpunkt um Dein Programm zu speichern. Klicke bitte auf die „Speichern“ Schaltfläche in der Werkzeugleiste.

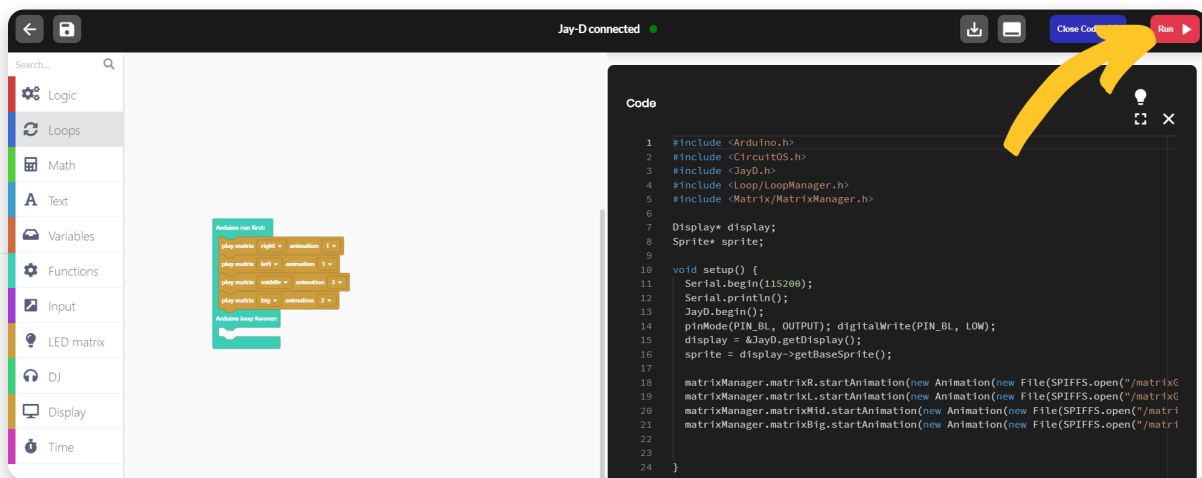


Überlege Dir einen Namen für Dein Programm und klicke dann zum Speichern

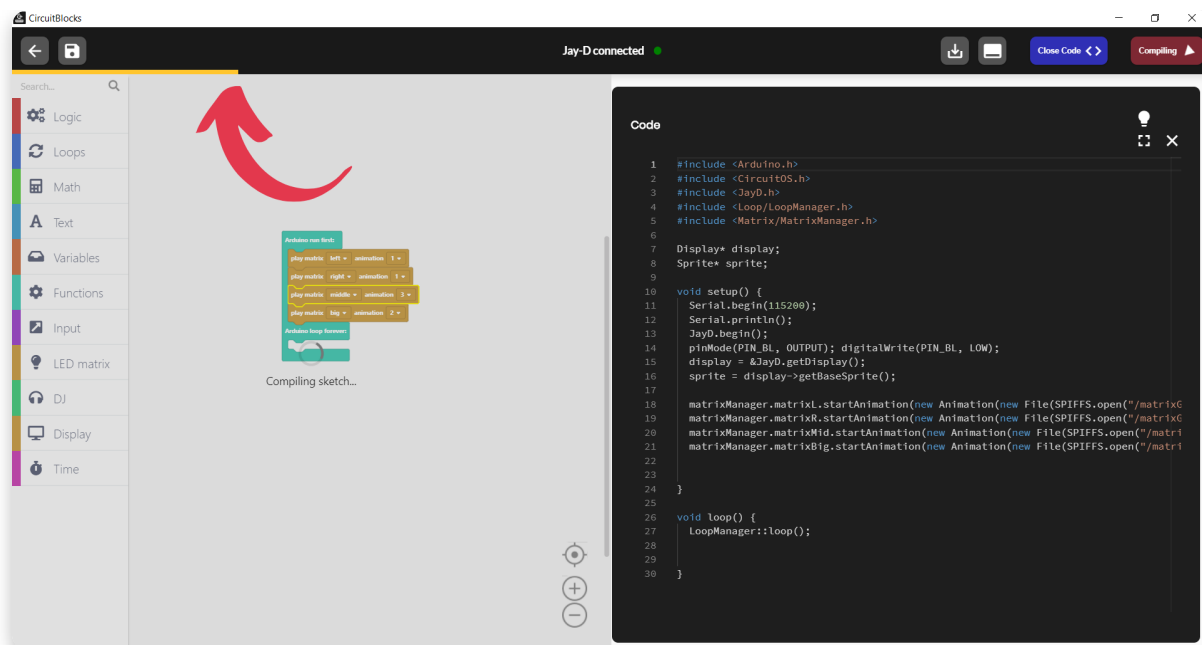
auf „Safe“.



Lass uns doch jetzt mal probieren, ob das Programm läuft. Such den großen roten Knopf oben rechts auf dem Bildschirm und drücke drauf.



Nun erscheint ein gelber Fortschrittsbalken oben auf dem Bildschirm.



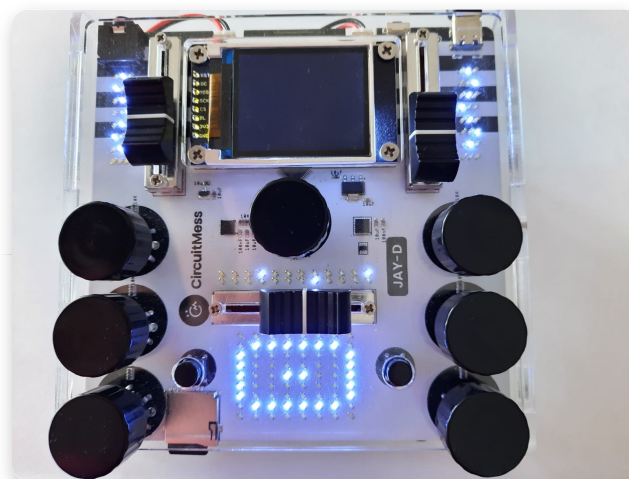
Dein Programm durchläuft nun eine Umwandlung **die Kompilation** genannt wird.

Dabei wird es aus der für Menschen verständlichen Form (wie zum Beispiel die bunten Blöcke) in eine Sprache übersetzt, die Maschinencode heißt.

Sie ist für Computer verständlich und besteht nur aus vielen Nullen und Einsen. Das kann nun einige Minuten dauern, da CircuitBlocks diese Umwandlung zum ersten Mal macht und auch die Grundbestandteile des Programm übersetzt werden müssen. Danach werden Sie auf Deinem Computer gespeichert und beim nächsten Mal geht alles viel schneller.

Wenn Dein Programm übersetzt und in Jay-D geladen wurde solltest Du die neuen Animationen auf den LED-Matrix-Displays sehen.

Funktioniert alles? Prima, dann lass uns weitermachen.



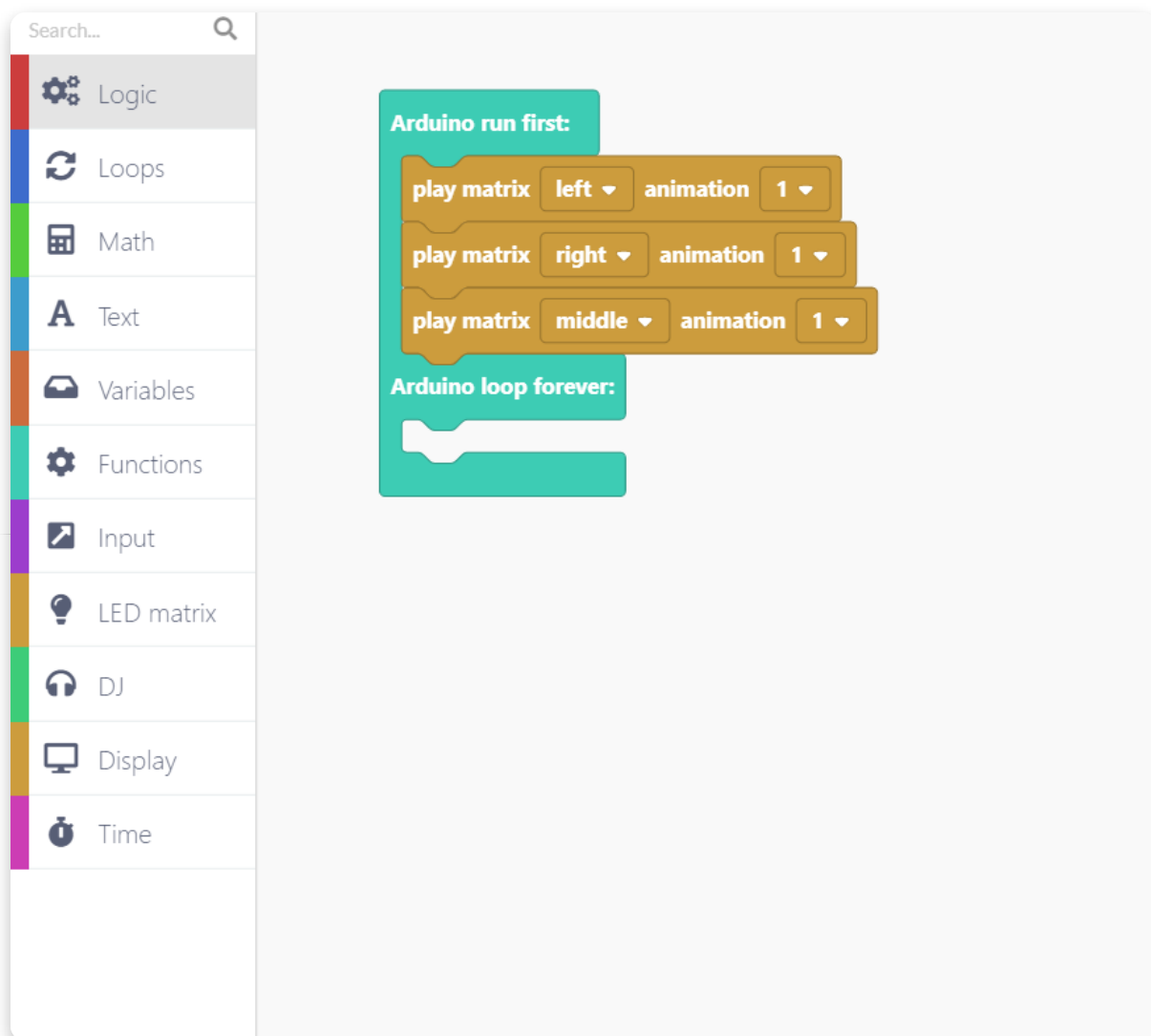
Klappt irgendetwas nicht? Dann drück bitte auf den Knopf **'Send error report'** am unteren Bildschirmrand. Dir wird dann eine Fehlermeldenummer angezeigt. Schreibe uns bitte eine Email und teile uns darin auch die Fehlermeldenummer (error report number) mit. Sende Deine Email bitte an contact@circuitmess.com und wir werden versuchen Dir so schnell wie möglich zu helfen.

LED Matrix

Lass uns nun noch etwas tiefer in die Programmierung der LED-Matrix-Displays einsteigen.

In diesem Abschnitt vertiefen wir das Gelernte aus der vorigen Übung und ergründen, was wir mit unterschiedlichen Variablen alles so machen können.

Wir erstellen zunächst ein neues Jay-D Block Projekt und lassen drei Blöcke Block "Play matrix left animation 1" wieder im Zeichenbereich fallen. So wie beim ersten Beispiel. Wir weisen den Blöcken die Displays „left, right, middle“ zu, ebenfalls wie im ersten Beispiel.

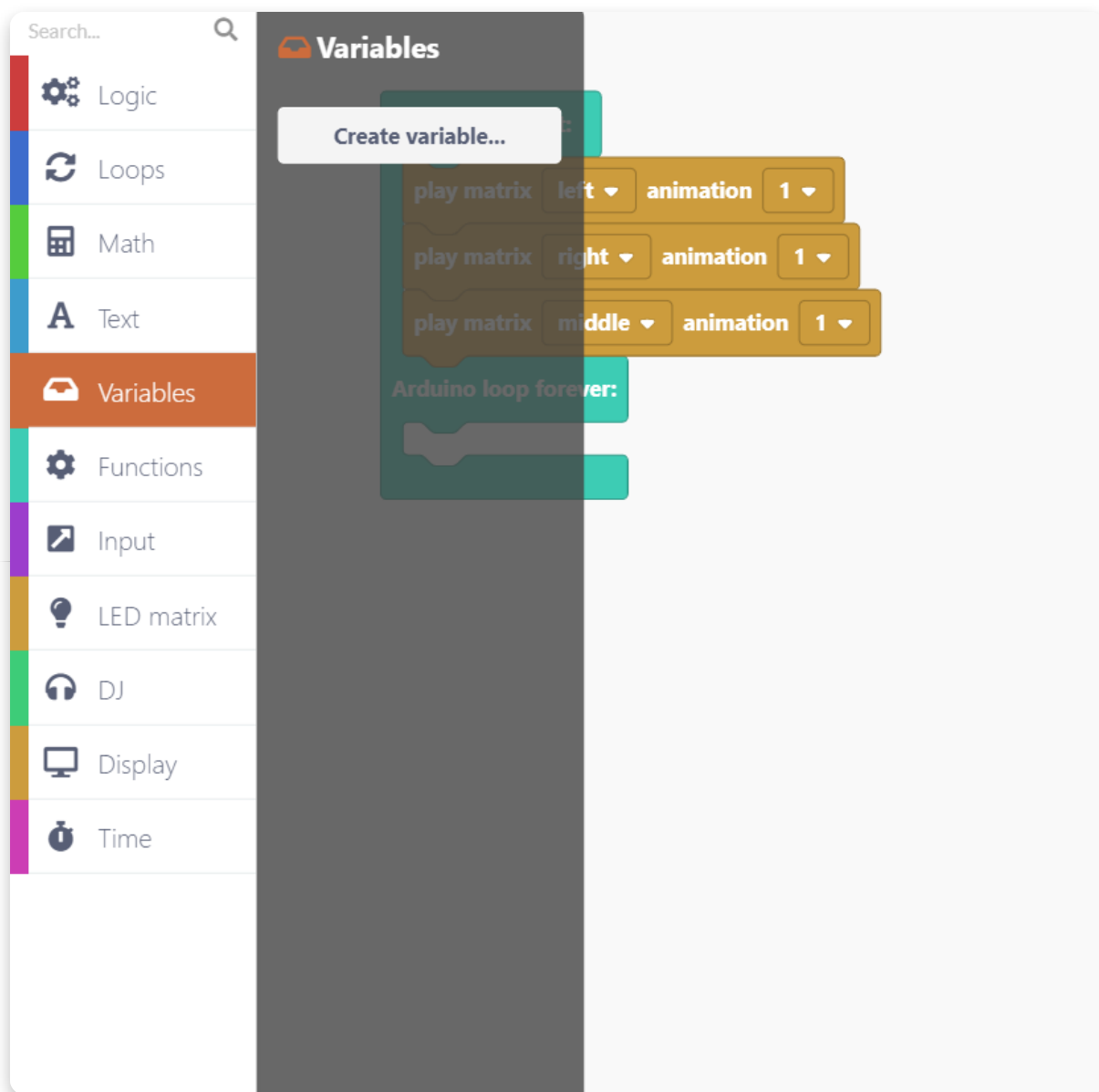


In der Computerprogrammierung ist eine **Variable** ein Ort, an dem ein Wert (z.B. eine Zahl) gespeichert werden kann.

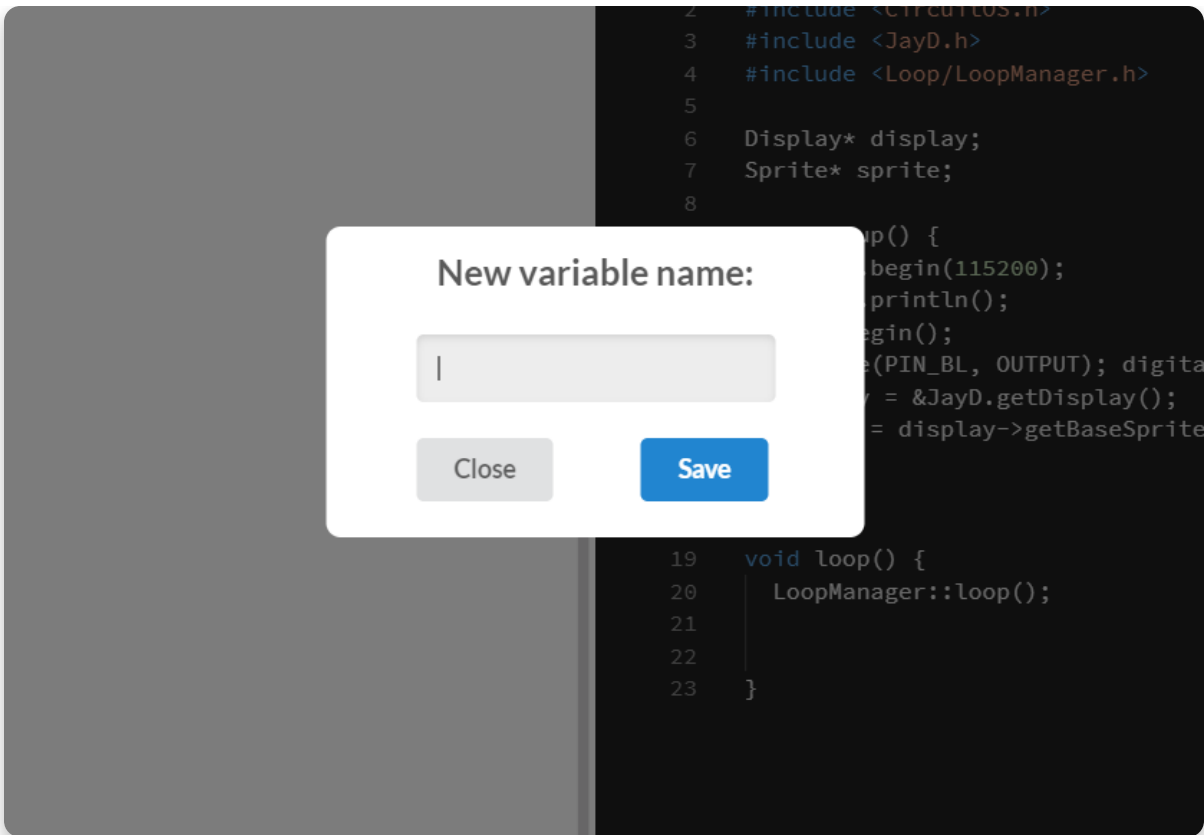
Du kannst den Wert jeder Variable ändern und speichern. Außerdem besitzt jede

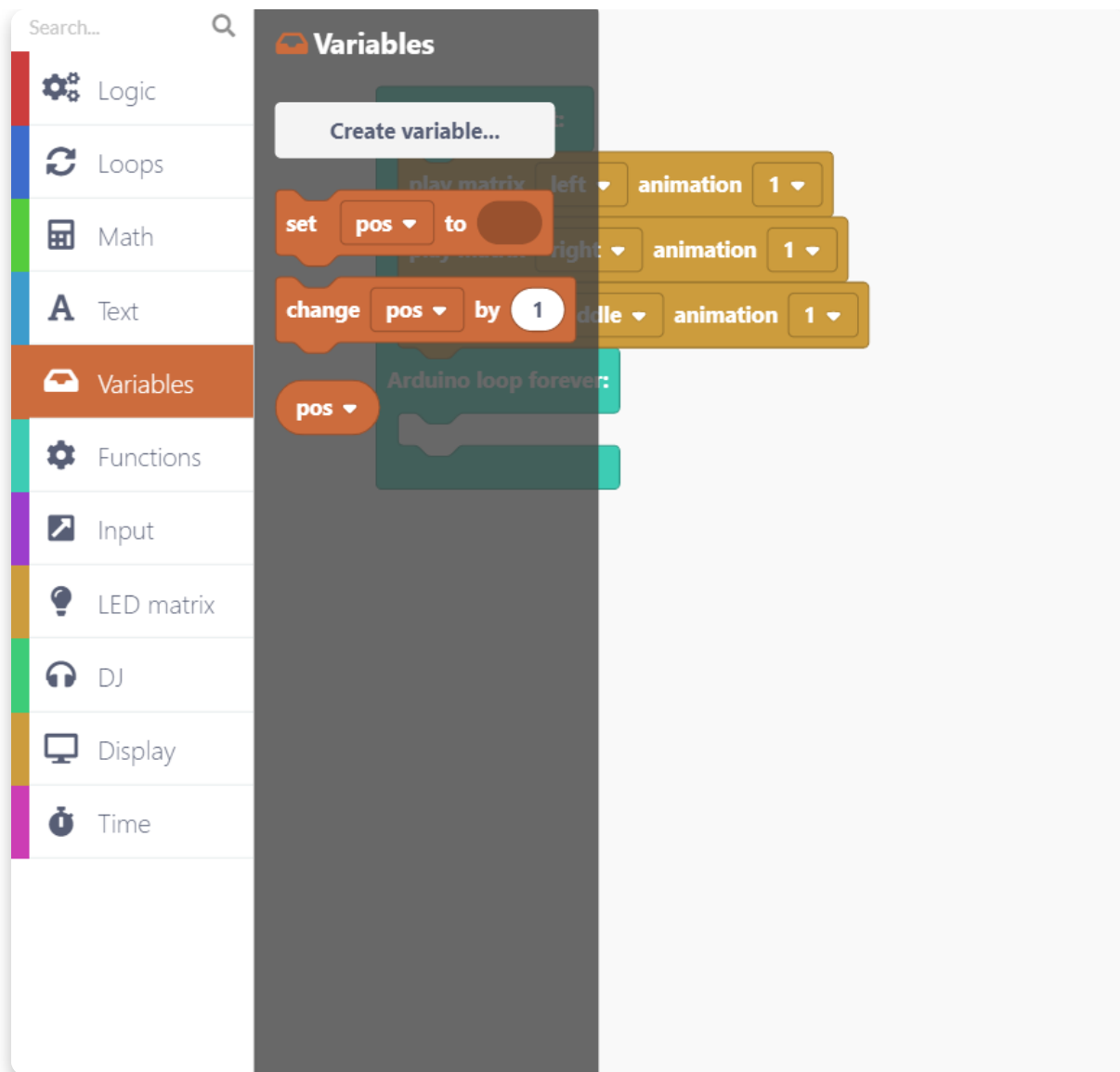
Variable einen spezifischen Namen.

Lass uns zunächst einmal eine Variable erstellen.



Suche die Kategorie „Variables“ und klicke auf "Create variable...". Nun musst Du der Variable einen Namen geben. Nehmen wir mal pos – als Abkürzung für Position auf dem Display.



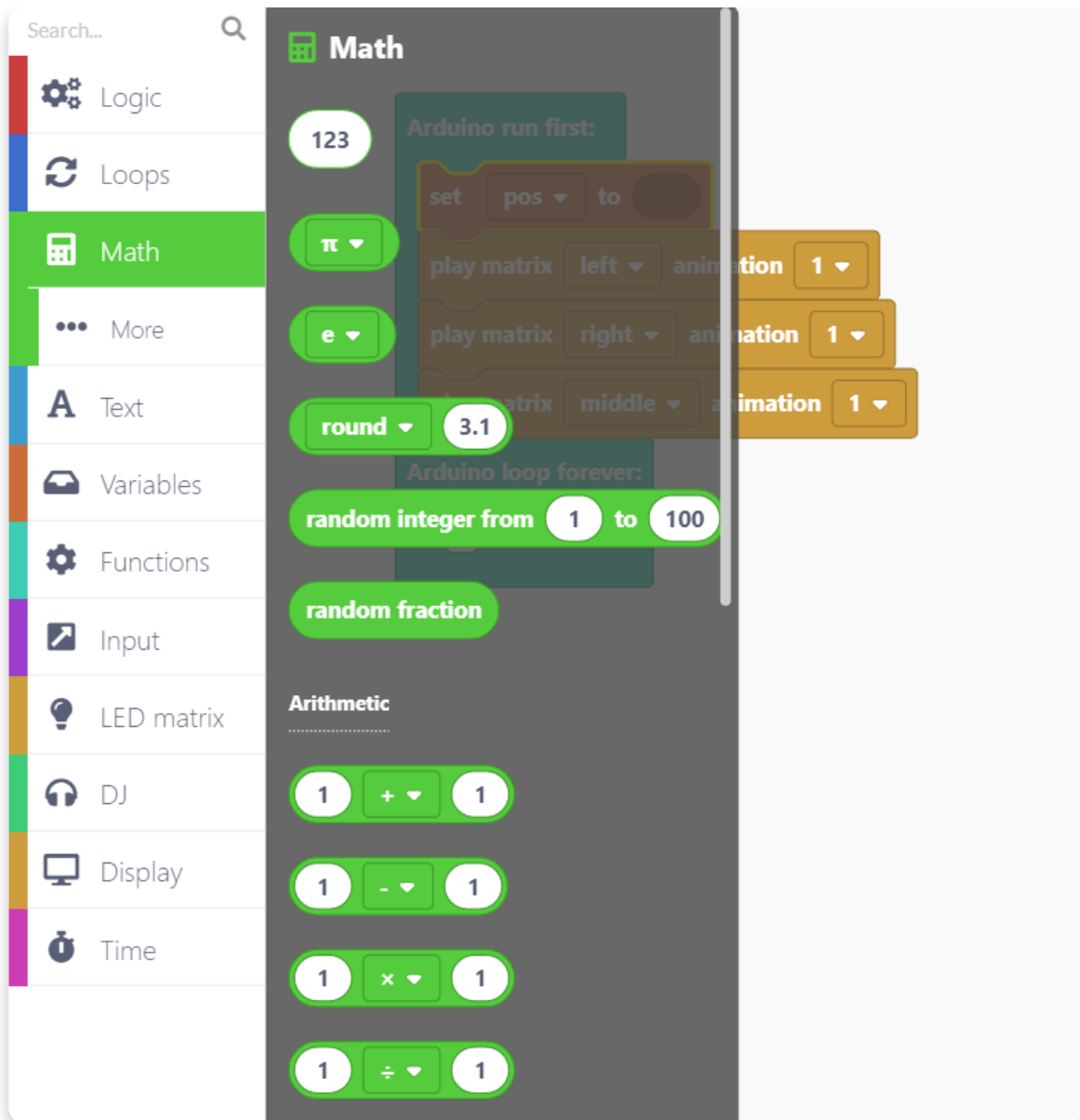


Jede Variable die wir neu erstellen besitzt noch keinen Wert, sie ist unbestimmt. Bei jedem Programmstart müssen wir jeder Variable einen Wert zuweisen. Dafür gibt es den Block „set variable“. Schiebe den Block „set pos to“ direkt unter "Arduino run first:".

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace displays an Arduino code snippet. The 'Arduino run first:' section contains three 'play matrix' blocks. The first block has 'pos' set to 'left' and 'animation' set to '1'. The second block has 'pos' set to 'right' and 'animation' set to '1'. The third block has 'pos' set to 'middle' and 'animation' set to '1'. The 'Arduino loop forever:' section is currently empty.

Jetzt müssen wir uns überlegen, welchen Wert wir der Variable geben. Suche den Block „123“ in der Kategorie „Math“.

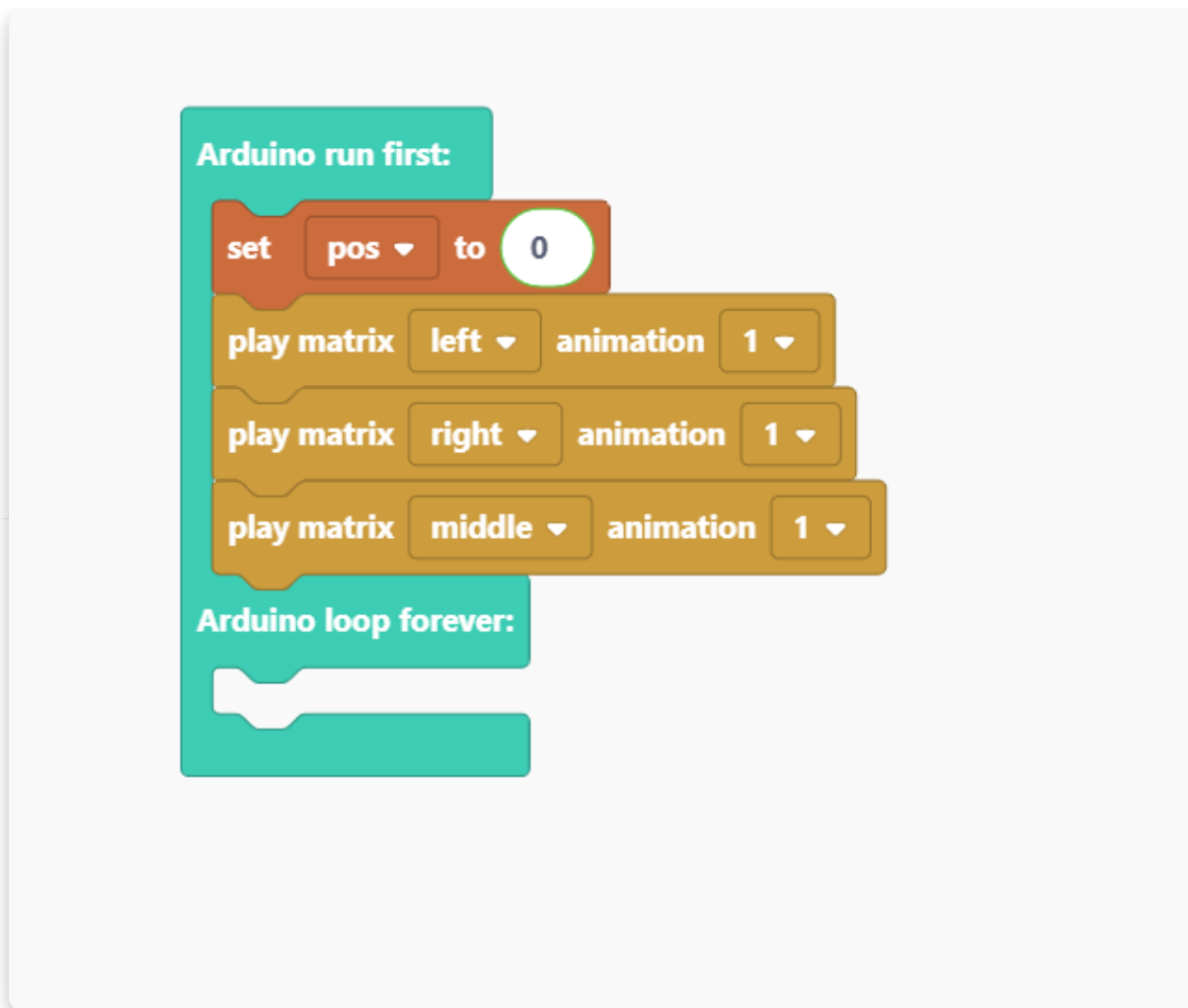
Dieser Block ist für Zahlenwerte. Nachdem Du ihn in den Zeichenbereich geworfen hast, kannst Du darin eine eine Zahl festlegen.



Schiebe den Block nach hier:



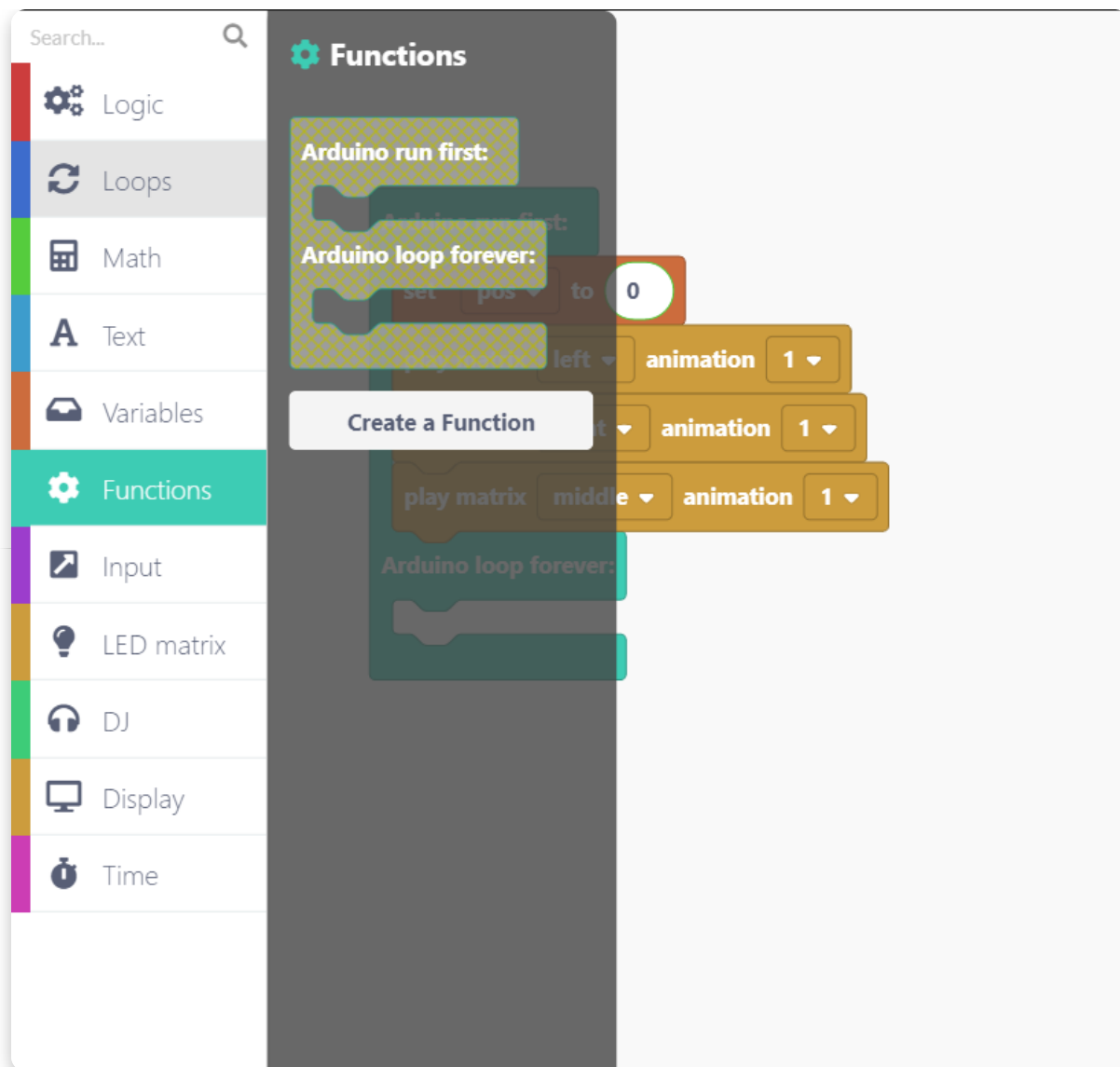
Jetzt klicke auf den Block und gib den Wert „0“ ein. Das machst Du, indem Du einfach die Taste 0 (Null) auf Deiner Tastatur drückst.



Nun haben wir eine Variable mit dem Wert 0. Jetzt lass uns den Wert ändern, wenn etwas bestimmtes passiert.

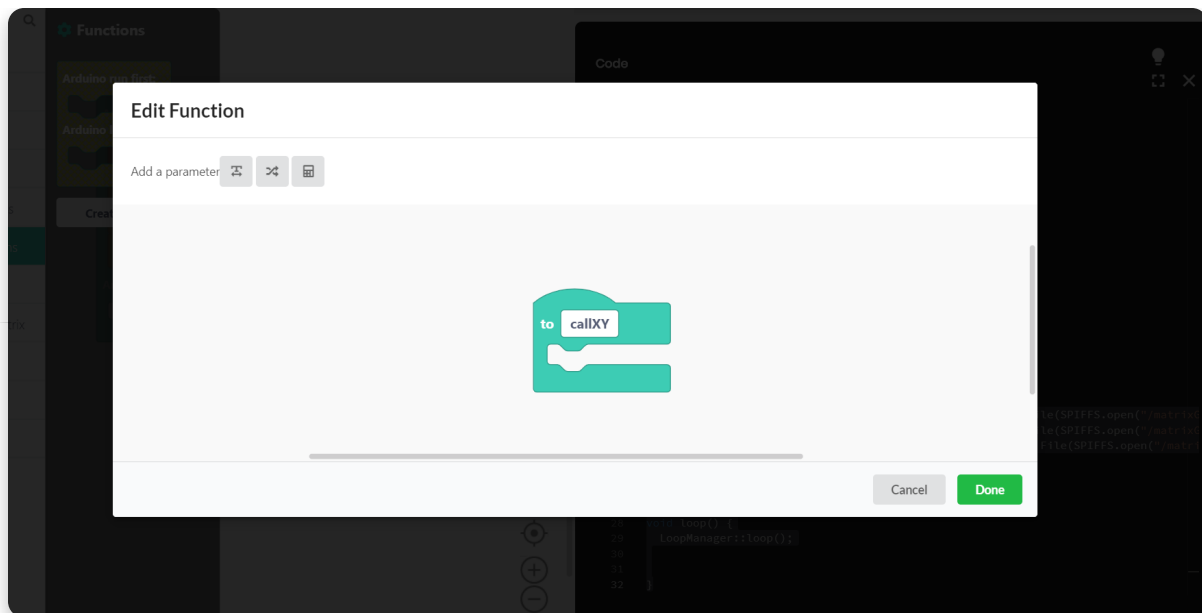
Im nächsten Schritt zeigen wir Dir, wie Du auf einem Display die X- und Y-Koordinaten für eine bestimmte LED bestimmen kannst.

Öffne die Kategorie Funktionen und wähle "Create a Function" aus.



Jetzt öffnet sich ein Fenster, wie das auf dem Foto. Hier kannst Du der neuen Funktion einen Namen geben.

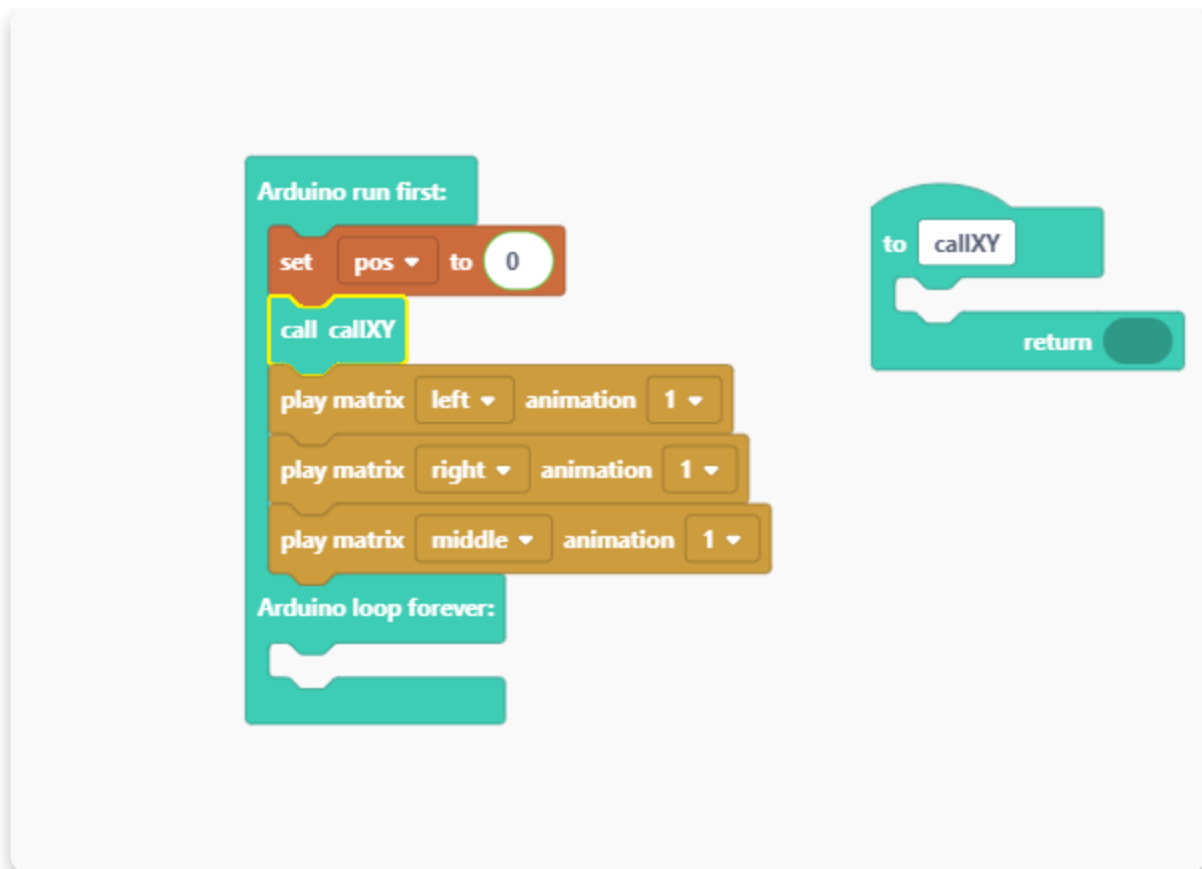
Wir werden damit die X- und Y-Koordinaten berechnen, darum nennen wir sie am besten „callXY“:



Nachdem die Funktion einen Namen bekommen hat, sollte sie als Block (mit ein paar weiteren Einstellmöglichkeiten) in der Kategorie „Functions“ erscheinen.

The screenshot displays the CircuitBlocks IDE interface. On the left, a sidebar contains a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions (highlighted), Input, LED matrix, DJ, Display, and Time. The main panel is titled "Functions" and features two "Arduino run first:" and "Arduino loop forever:" blocks. Below these is a "Create a Function" button and two "call callXY" blocks. The code editor on the right shows a "to callXY" block with a "return" button, followed by an "Arduino run first:" block containing a "set pos" block (set to 0) and three "play matrix" blocks (left, right, and middle, all with animation 1). Below this is an "Arduino loop forever:" block.

Nimm den obersten „callXY“ Block aus dem Abschnitt und lass ihn direkt unter dem „set pos“ Block im Zeichenbereich fallen.



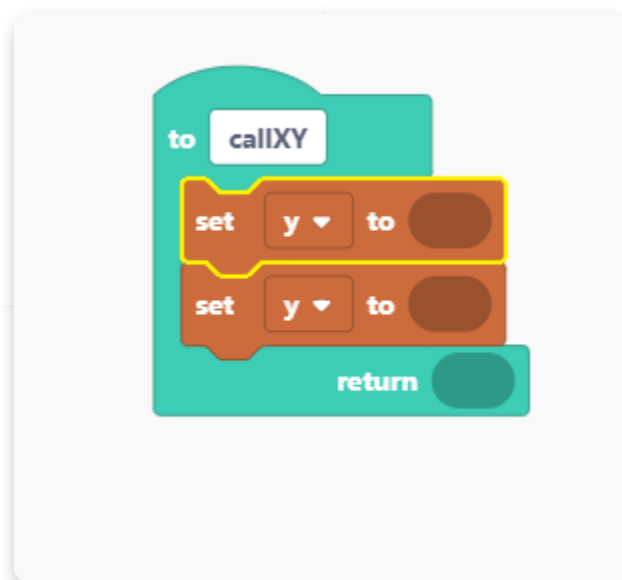
Jetzt legen wir fest, was die Funktion macht.

Dazu erstellen wir noch zwei weitere Variablen – genau so, wie wir es mit der „pos“ Variable gemacht haben.

Öffne die Kategorie „Variables“ und erstelle die Variablen x und y. Danach sollten beide nun als Blöcke in der Kategorie erscheinen.

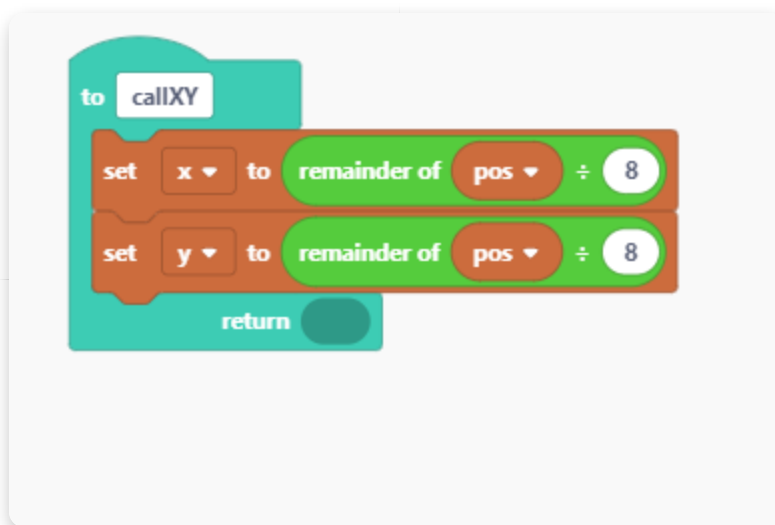


Nimm jetzt zweimal den Block "set y to" und lasse beide hier in die Funktion fallen:



Die LED-Matrix-Displays bestehen jeweils aus 8 Reihen. Wir benutzen diesen Befehl um die LEDs in der richtigen Reihenfolge einzuschalten:

- Wähle im Aufklappmenü der beiden neuen Variablen die Namen x und y aus.
- Um den Positionswert durch 8 zu teilen wähle die Mathematische Funktion "remainder of" aus.
- Wähle für beide „pos“ aus dem Aufklappmenü aus.



Gute Arbeit! Der Abschnitt "Arduino run first" ist jetzt fertig.

Search...

- Logic
- Loops
- Math
- Text
- Variables
- Functions
- Input
- LED matrix
- DJ
- Display
- Time

Arduino run first:

- set pos to 0
- call callXY
- play matrix left animation 1
- play matrix right animation 1
- play matrix middle animation 1

Arduino loop forever:

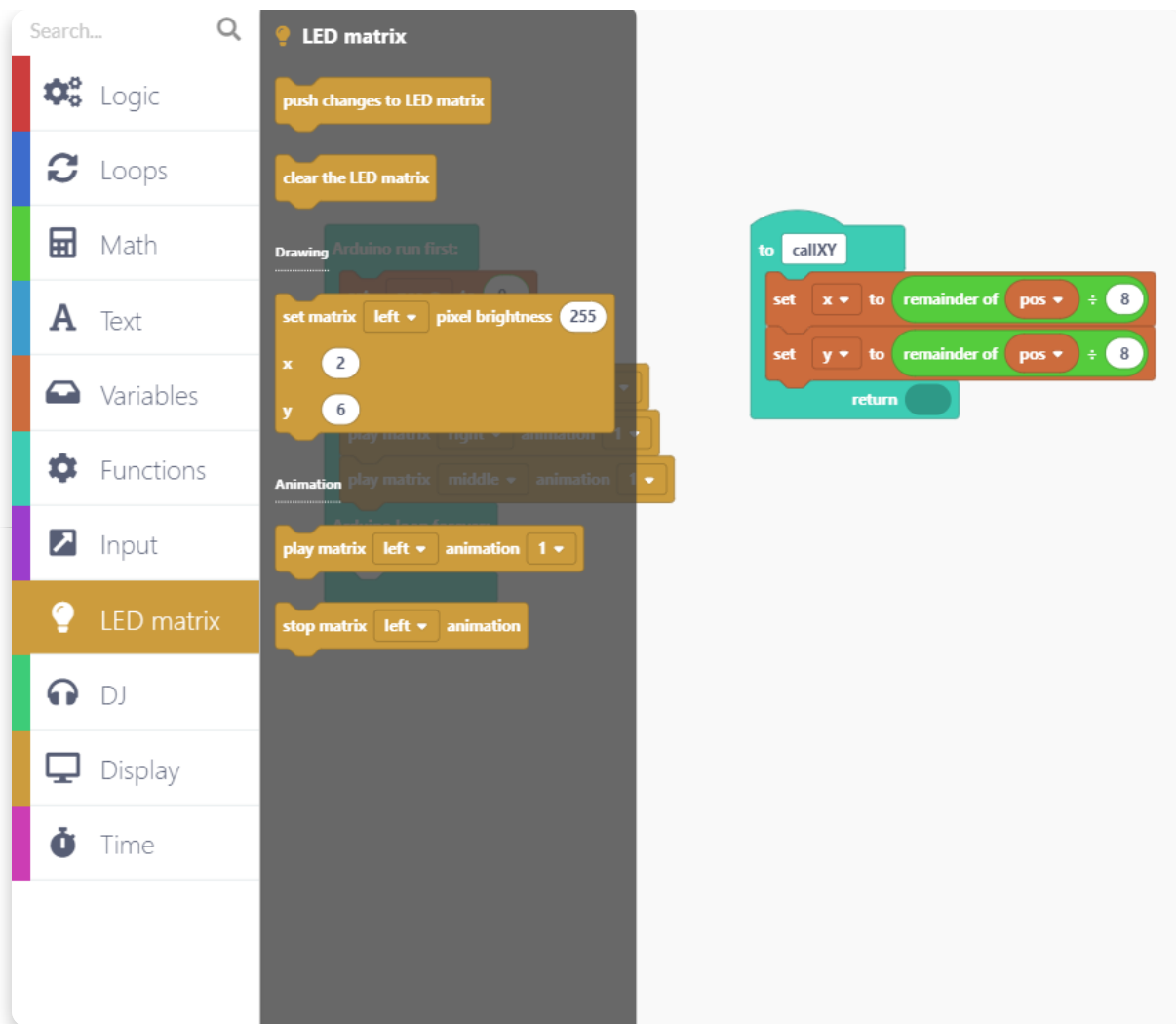
- key icon

to callXY

- set x to remainder of pos ÷ 8
- set y to remainder of pos ÷ 8
- return

Lass uns nun etwas mit den LEDs des großen Displays spielen.

Wähle aus das Kategorie „LED matrix“ den Block „set matrix“ aus.



Lasse den Block in die Lücke unter "Arduino loop forever" fallen.

Wähle im Aufklappmenü die große LED-Matrix (big) aus und setze die Helligkeit (pixel brightness) auf 0.

Setze in der Zeilen für x die Variable x ein. In der Zeile für y entsprechend die Variable y (wie Du auch auf dem folgende Foto erkennen kannst):

The screenshot displays the CircuitBlocks IDE interface. On the left, a sidebar with a search bar lists various coding categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace shows two code blocks. The first block, titled 'Arduino run first:', includes a 'set pos to 0' block, a 'call callXY' block, and three 'play matrix' blocks for 'left', 'right', and 'middle' columns, each with an animation of 1. The second block, titled 'Arduino loop forever:', includes a 'set matrix big pixel brightness 0' block and two variable input blocks for 'x' and 'y'. To the right, a separate 'to callXY' function block is shown, containing two 'set' blocks: 'set x to remainder of pos ÷ 8' and 'set y to remainder of pos ÷ 8', followed by a 'return' block.

Jetzt füge die Blöcke, wie im folgenden Foto gezeigt, hinzu. Durch diese Blöcke wird die Position der aktiven (leuchtenden) LED verändert. Dies wird mit den Logik- und Zeitfunktionen sowie den Variablen beeinflusst.

Jetzt wird es Zeit für einen ersten Probelauf Deines Programms.

Klicke auf den „Run“ Knopf, der sich oben rechts auf dem Bildschirm befindet. Solltest Du Deinem Projekt noch keinen Namen gegeben haben, so mach dies bitte nun. Warte, bis CircuitBlocks mit dem Kompilieren fertig ist.

TRANSLATE to [de]: Let's code some more

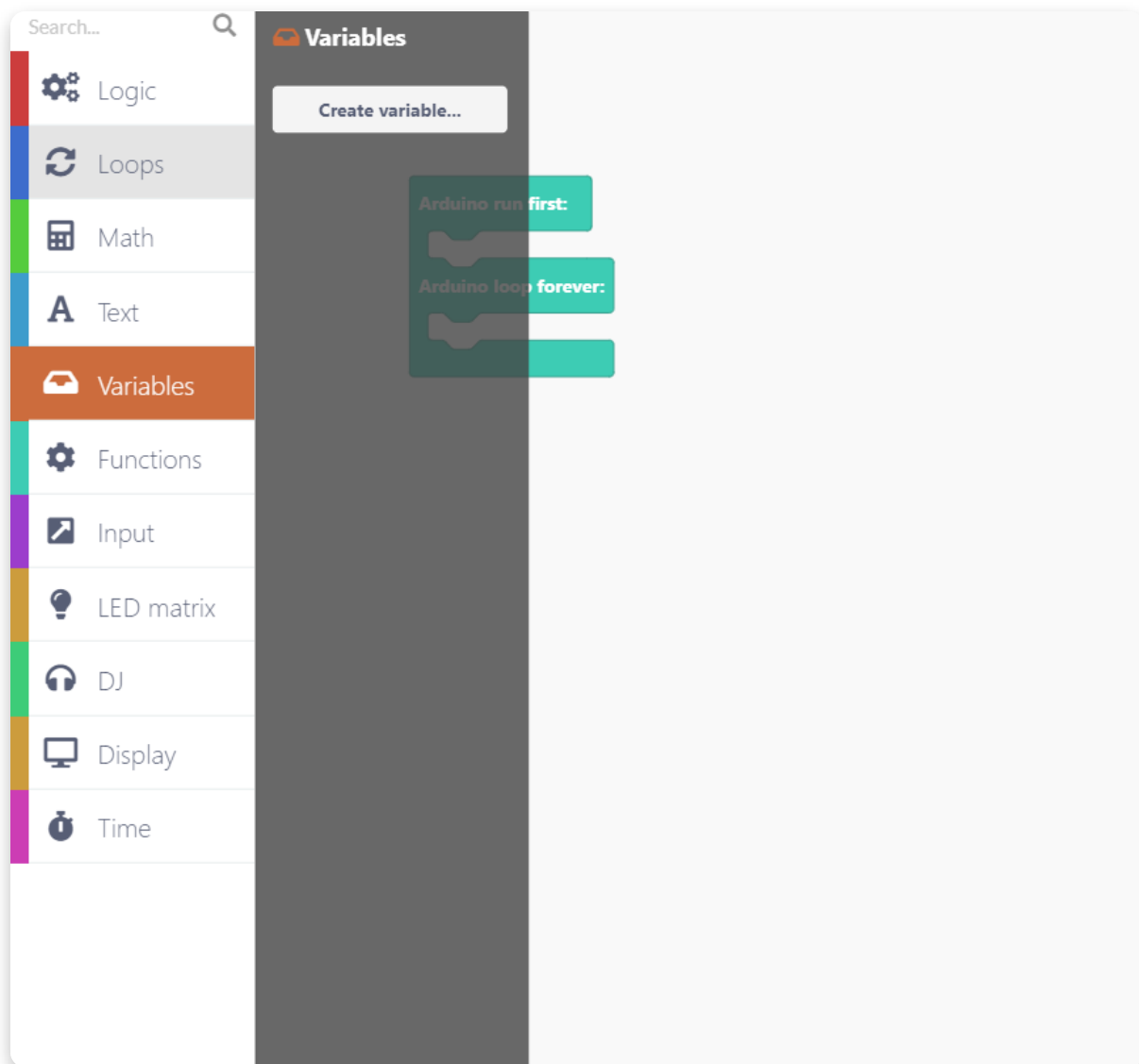
Zeichnen

Hast Du schon gewusst, das Du auch auf dem Display zeichnen kannst?

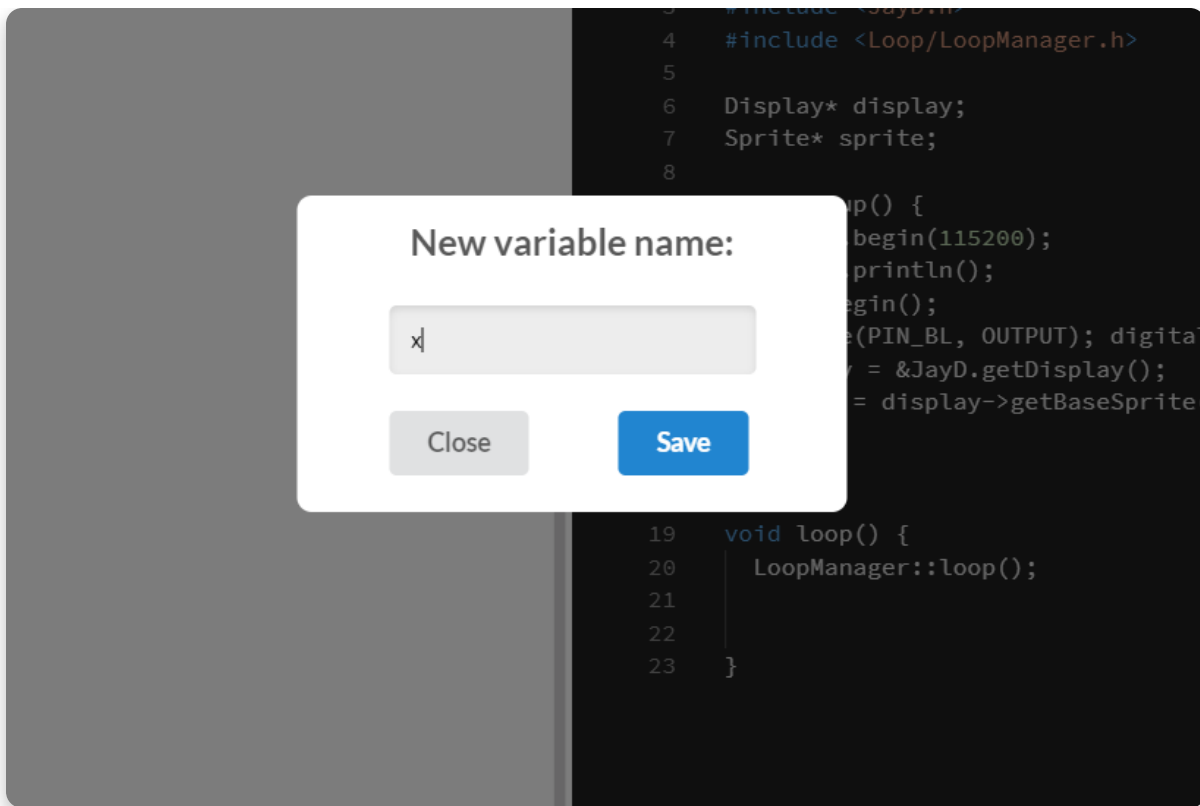
In diesem Abschnitt der Anleitung zeigen wir Dir, wie Du mit Jay-D's virtuellem Zeichenstift arbeiten kannst.

Öffne bitte zunächst ein neues Projekt.

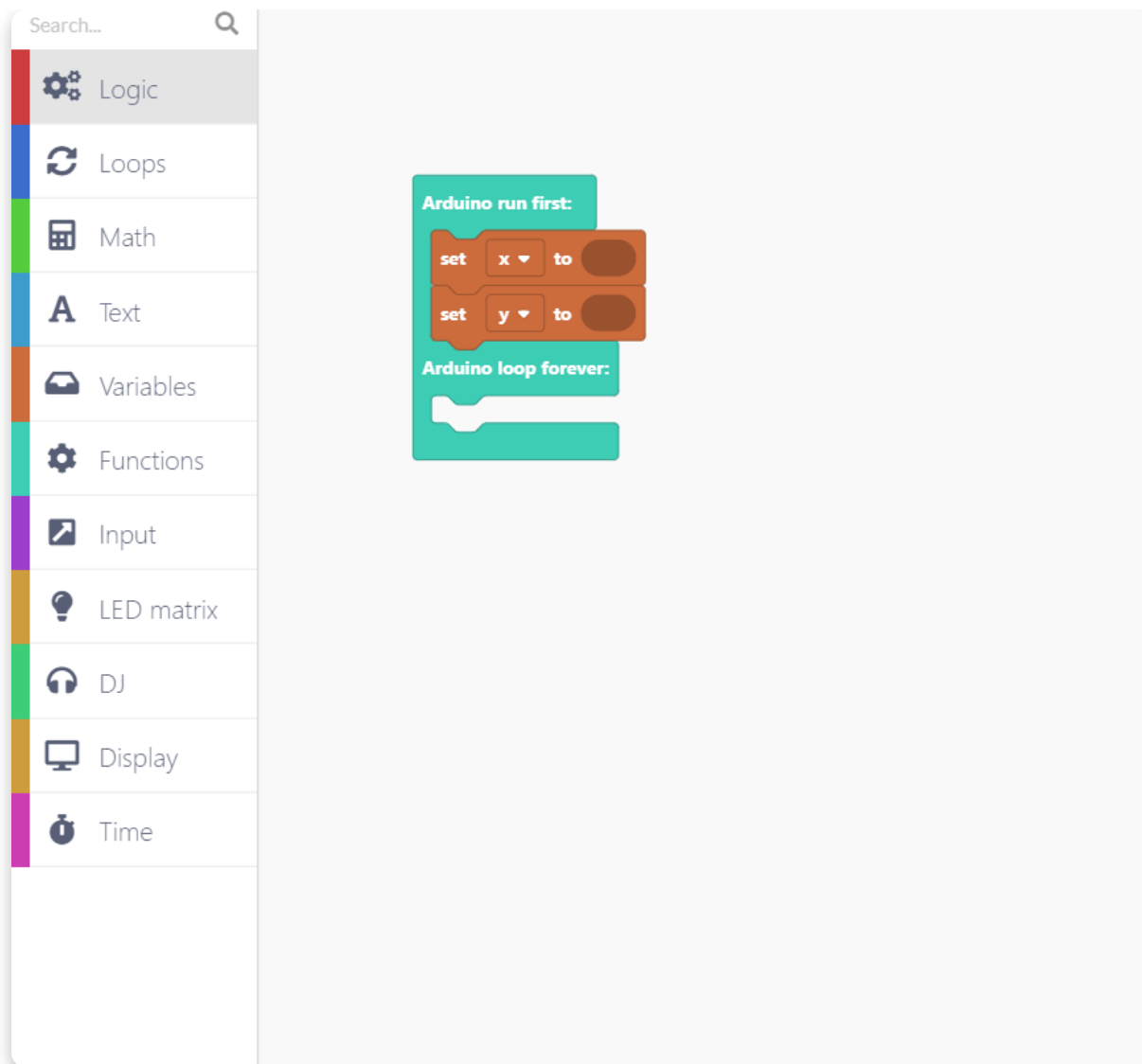
Erstelle als erstes zwei neue Variablen und nenne sie x und y. **X steht für die Koordinate auf der x-Achse, Y entsprechend für die Koordinate auf der y-Achse.**



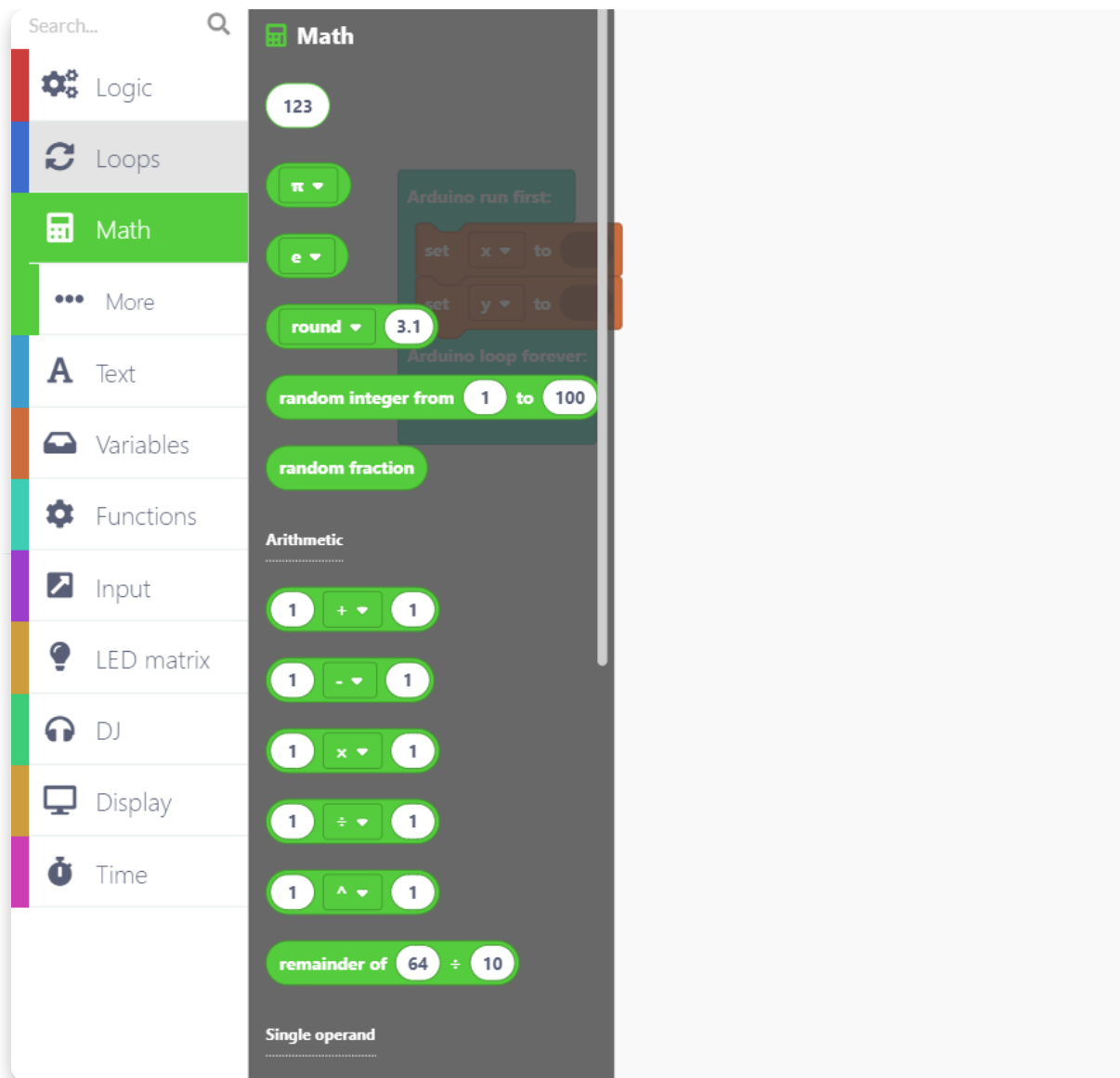
Erstelle zwei neue Variablen: x und y:



Ziehe bitte jetzt zweimal den Block "set y/x to" hier hin und lasse ihn fallen:

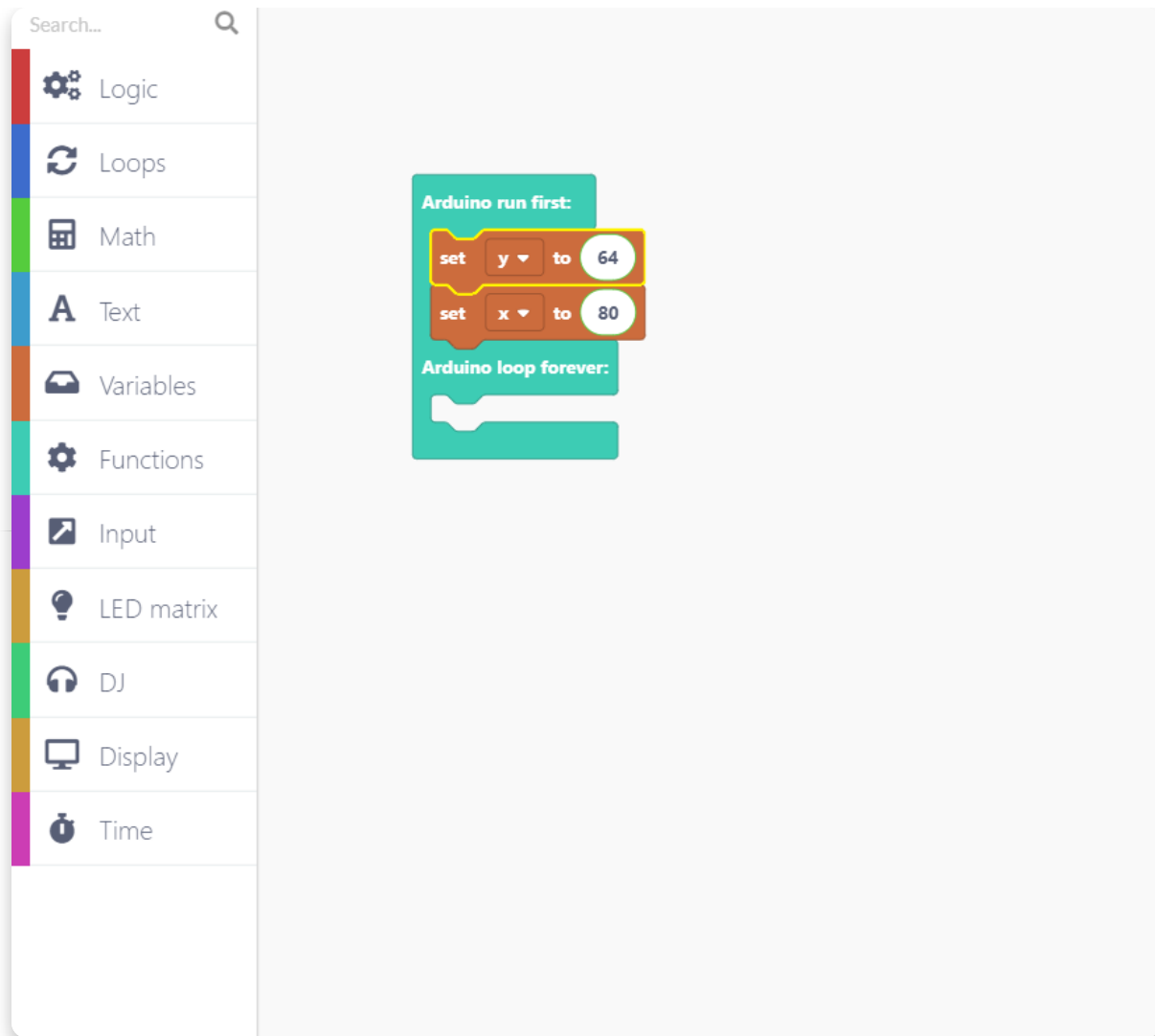


Wähle aus der Kategorie „Math“ den „123“-Block aus und lasse ihn im vorherigen Block (der die Koordinaten der Variablen bestimmt) fallen.

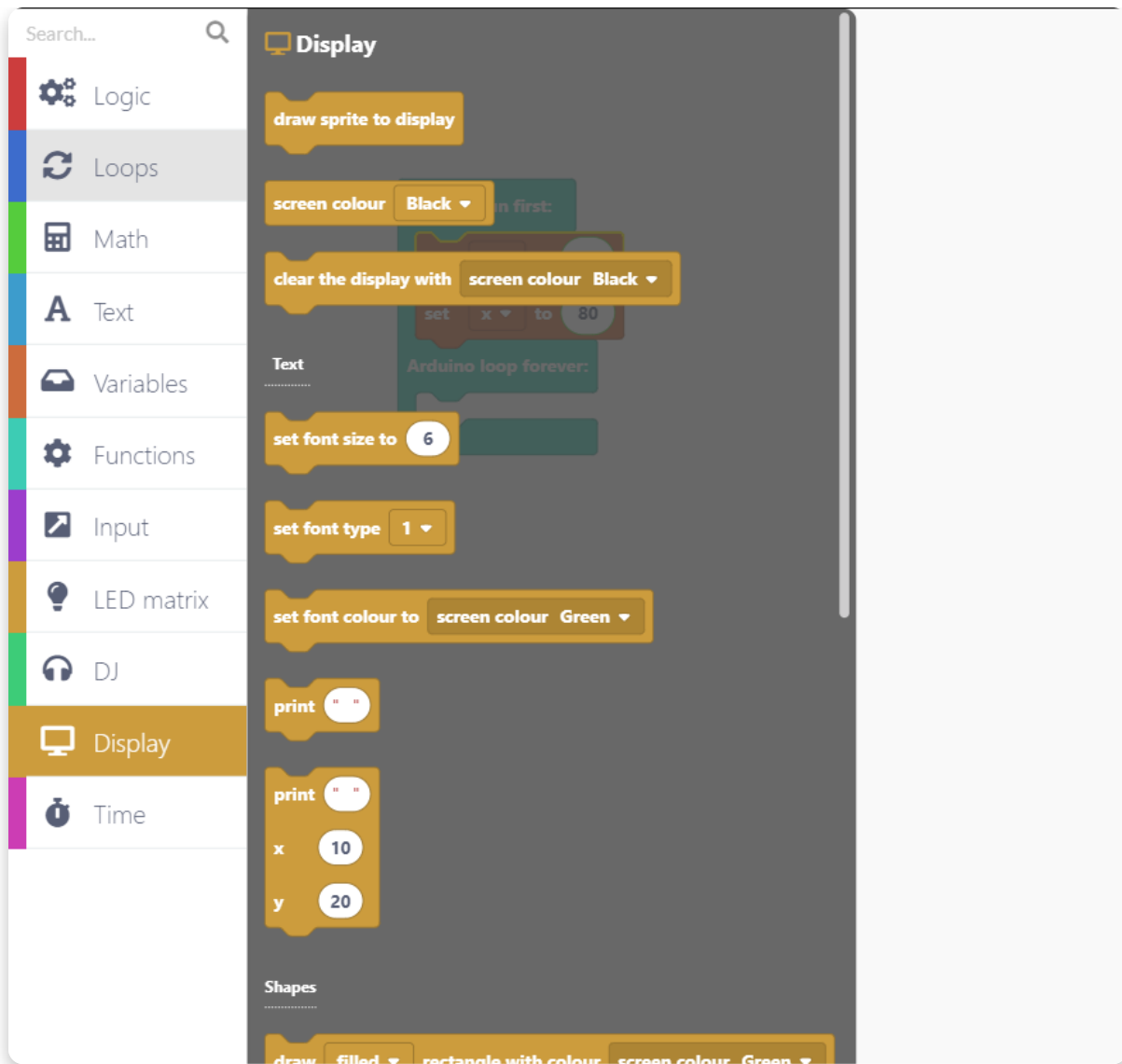


Lasse den „123“-Block in beiden "set x/y to" Blocks fallen.

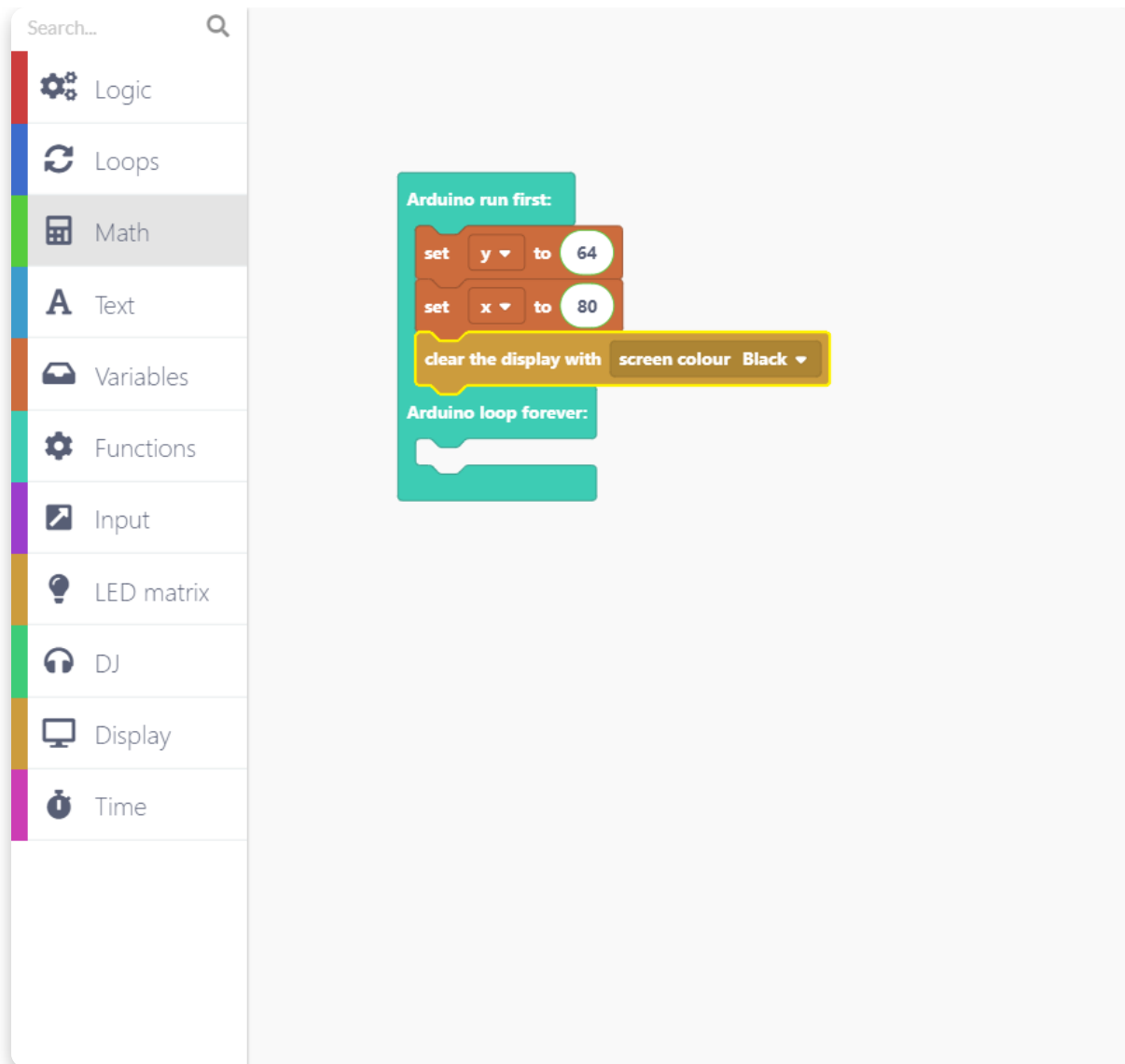
Das Display hat eine Auflösung von 128x160 Pixeln. Um in der Mitte zu beginnen müssen wir also der Variablen x den Wert 64 und der Variablen y den wert 80 geben.



Um die Hintergrundfarbe auf schwarz zu setzen, gehe in die Kategorie „Display“ und suche den Block "clear the display with screen color".

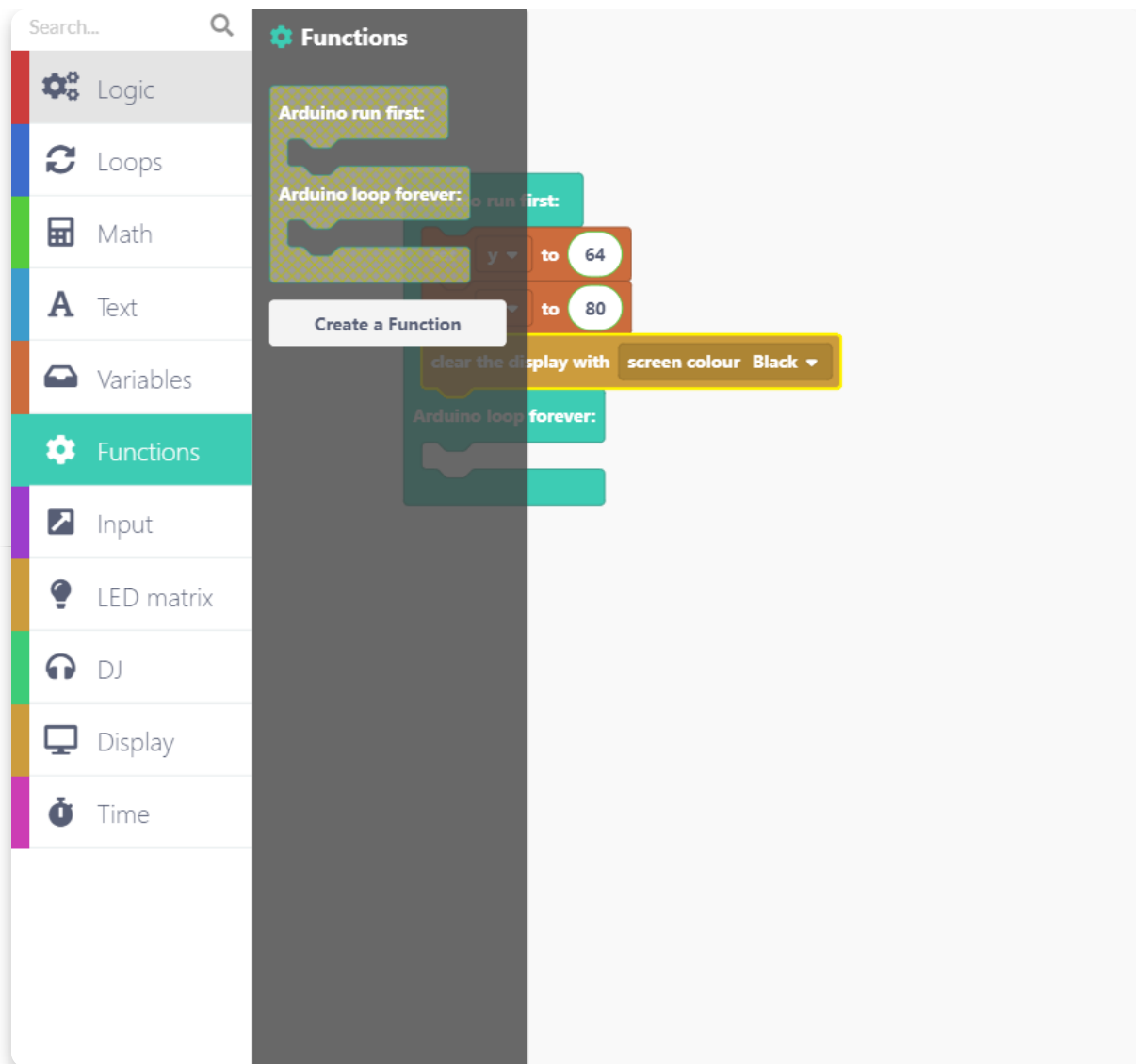


Zieh den Block hier her und lass ihn fallen:

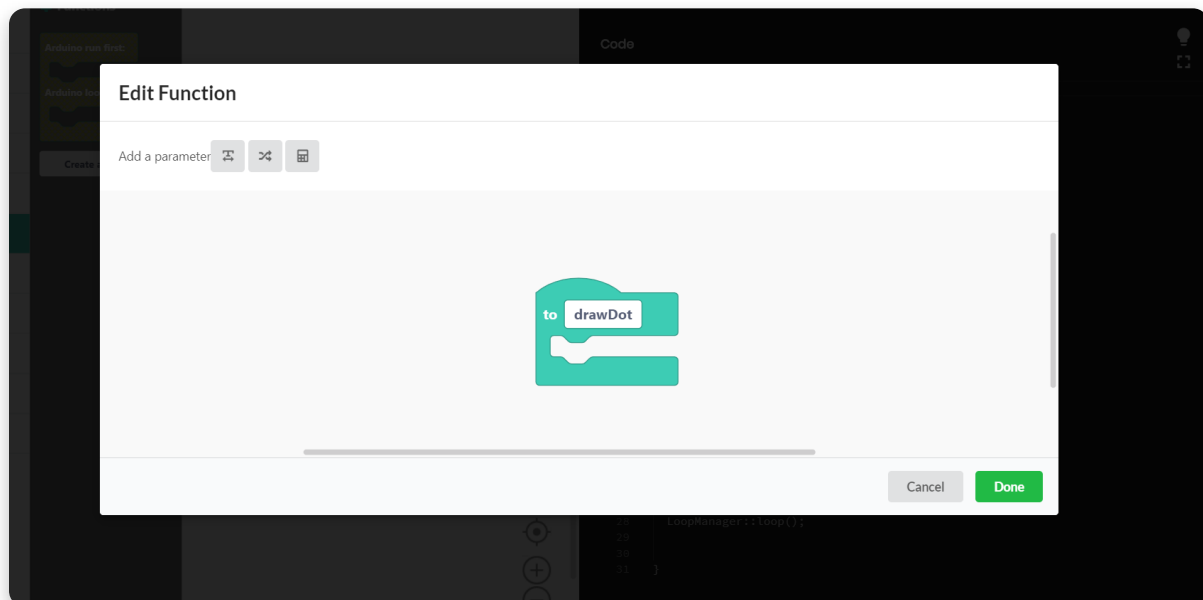


Jetzt haben wir bereits den Startpunkt unseres Stiftes festgelegt und die Hintergrundfarbe „schwarz“ eingestellt.

Lass uns nun eine neue Funktion erstellen, die Pixel in einer Farbe Deiner Wahl zeichnet. Wähle dazu in der Kategorie „Functions“ den Knopf "Create a Function" und klicke drauf.

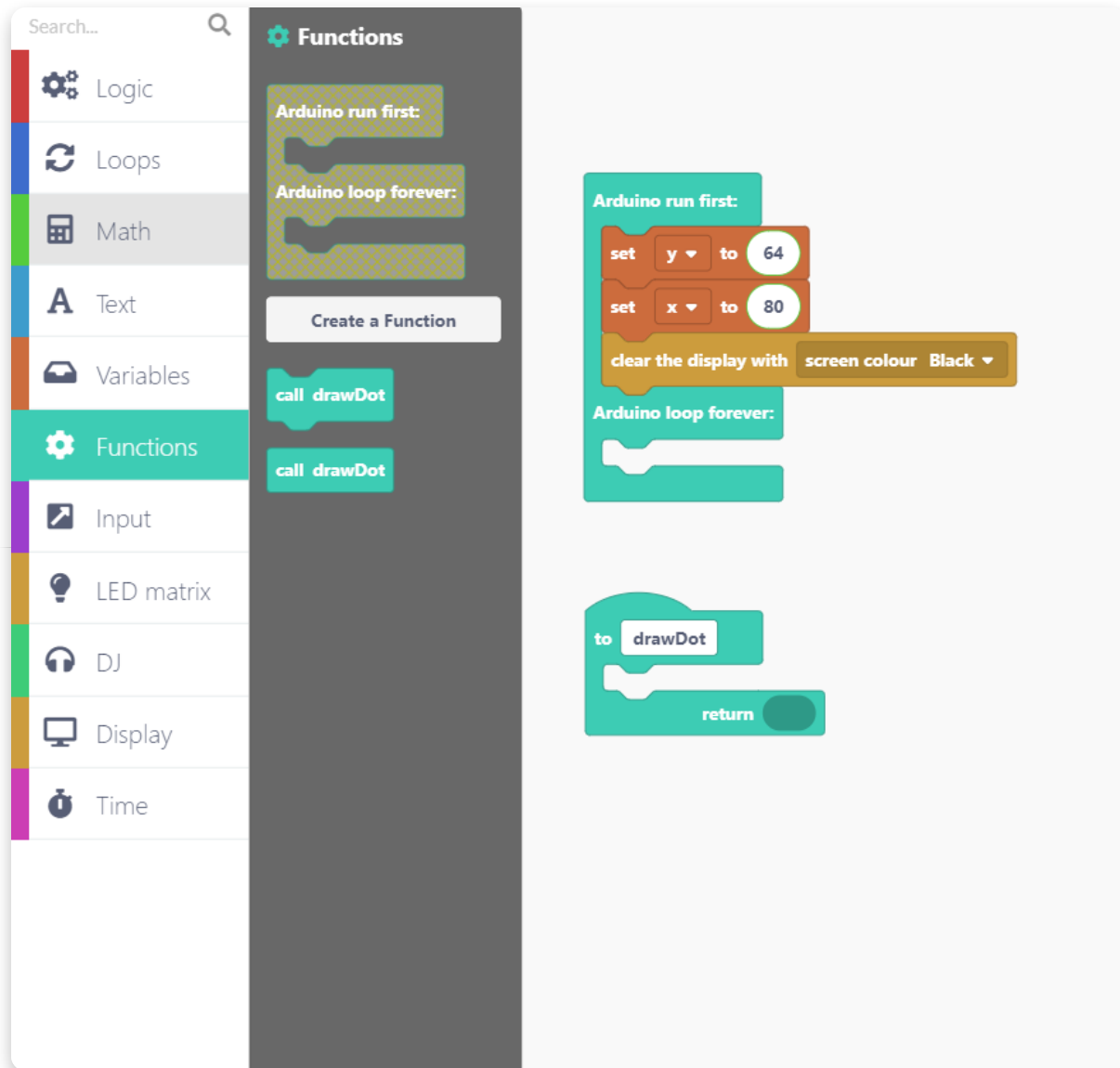


Wir nennen die neue Funktion "drawDot":

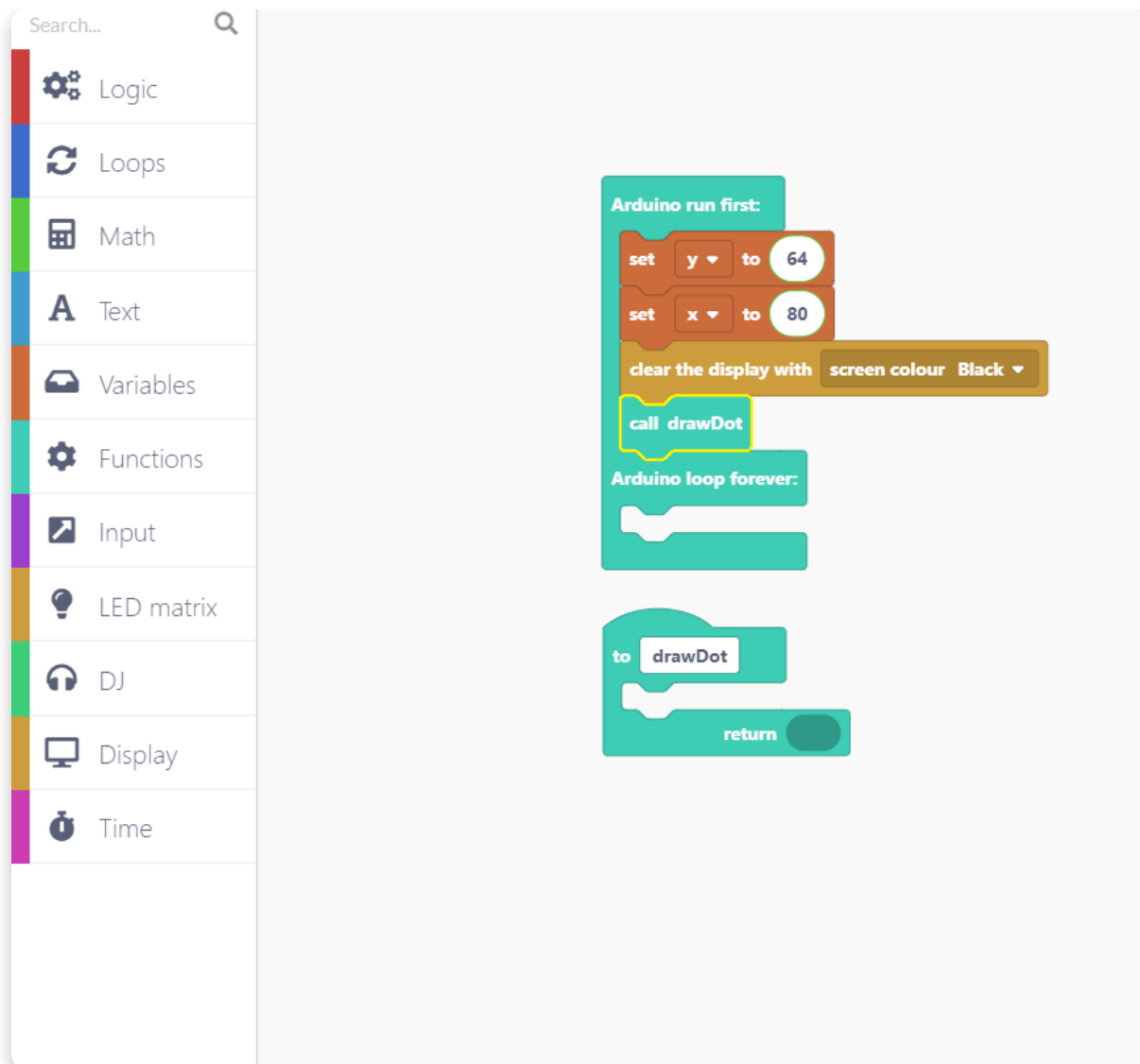


Wenn Du die Funktion benannt hast, erscheint sie als Block auf dem Bildschirm.

Darüber hinaus erscheinen in der Kategorie „Functions“ nun auch noch ein paar weitere Einstellmöglichkeiten.



Suche die „call drawDot“ Funktion und lasse sie unter dem „clear the display“ Block fallen:



Im nächsten Schritt legen wir fest, was die drawDot Funktion machen soll.

Gehe in die „Display“ Kategorie und suche unter „Shapes“ den Block „draw... rectangle with colour“.

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display (highlighted), and Time. The main workspace contains a code editor with the following blocks:

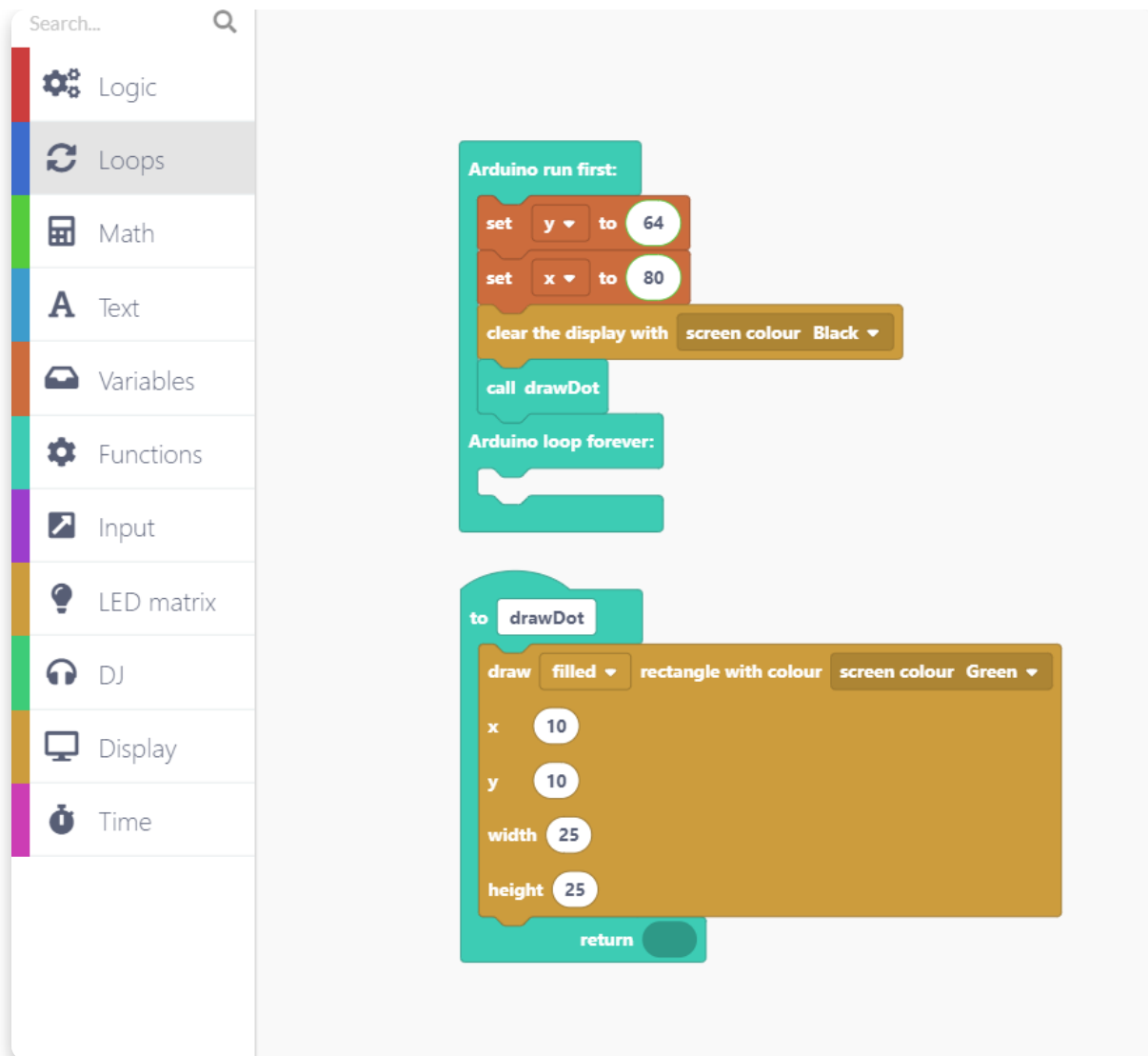
- set font colour to** screen colour Green
- print** " "
- print** " "
- x** 10
- y** 20
- draw filled rectangle with colour** screen colour Green
 - x** 10
 - y** 10
 - width** 25
 - height** 25
- draw filled circle with colour** screen colour Green
 - x** 30
 - y** 30
 - radius** 25
- draw filled ellipse with colour** screen colour Green

In the background, a semi-transparent window titled "Arduino run first:" is visible, containing:

- set y** to 64
- set x** to 80
- clear the display with** screen colour Black
- call drawDot**

Below it, another semi-transparent window titled "Arduino loop forever:" is visible.

Nimm ihn und lasse ihn in die drawDot Funktion fallen.



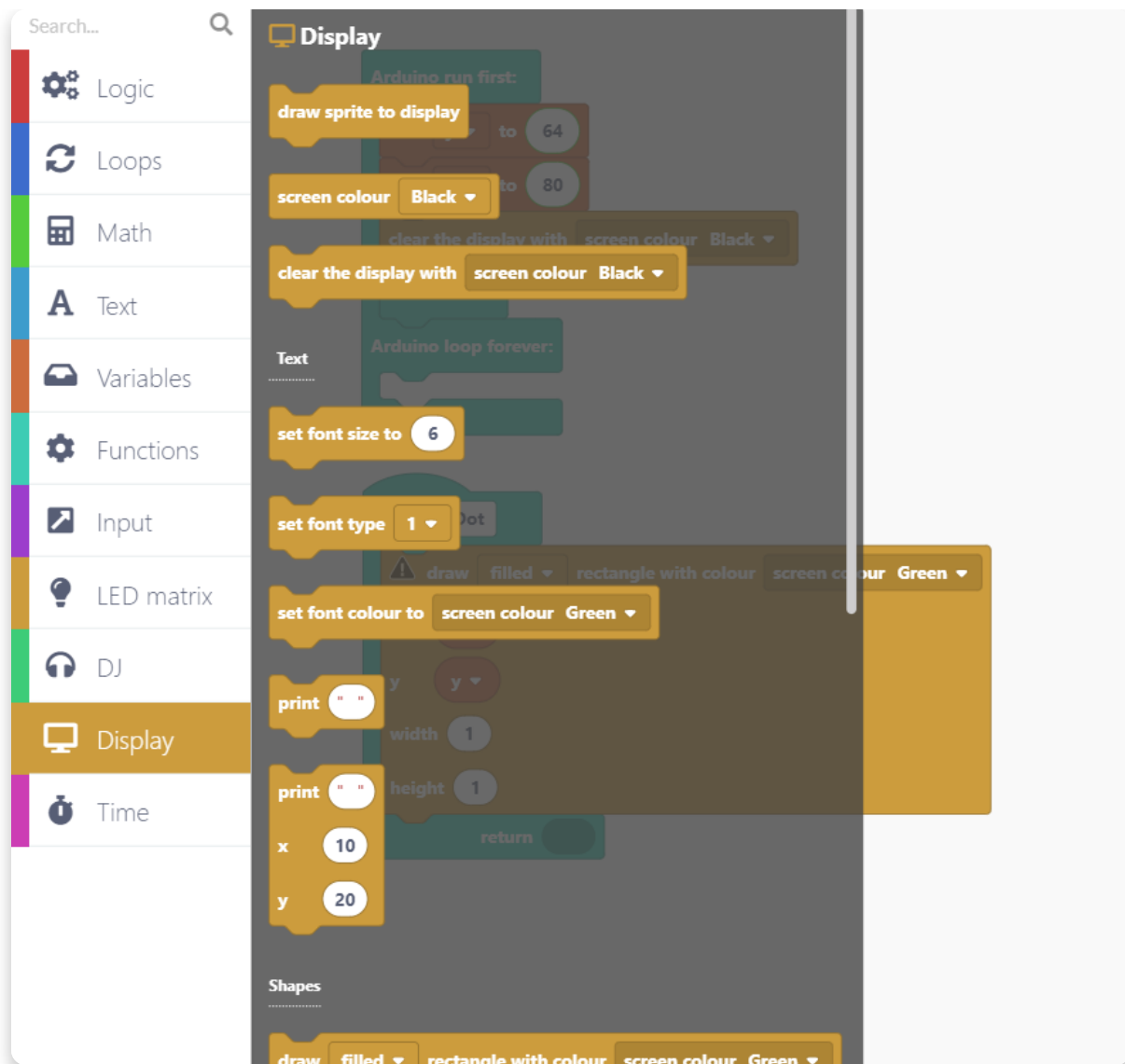
Nachdem Du das gemacht hast, nimm dort bitte folgende Einstellungen vor:

- Wähle im ersten Aufklappmenü "filled" aus. Wähle nun im Aufklappmenü weiter rechts eine beliebige Farbe aus. Wir lassen bei unserem Beispiel die Farbe grün stehen.
- Gib in den Feldern für die Variablen bei x ein x in den weißen Kasten ein. Entsprechend kommt ein y in den weißen Kasten neben dem x.
- Gib bei „width“ (Breite) und „height“ (Höhe) jeweils die Zahl 1 ein. So zeichnen wir dann einen kleinen Punkt.

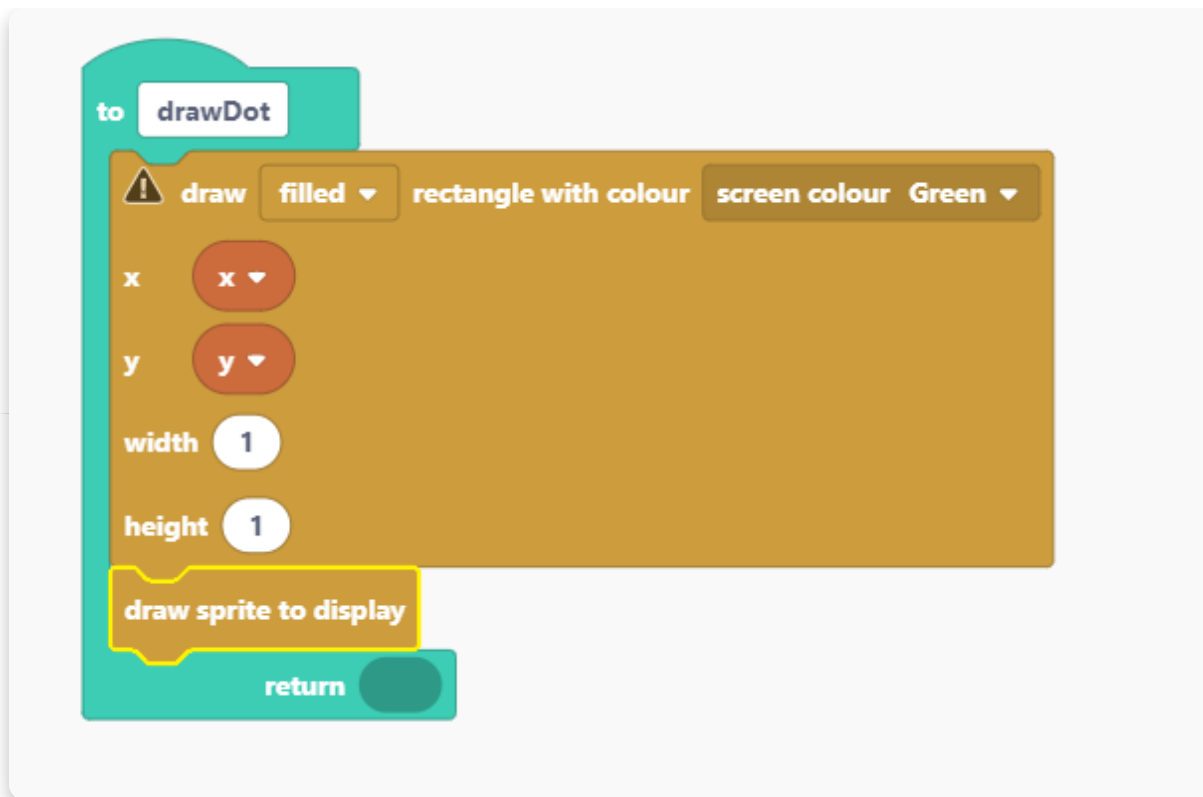
So sollte Dein Programm nun aussehen:

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace contains an Arduino sketch. The sketch starts with an 'Arduino run first:' section containing three blocks: 'set y to 64', 'set x to 80', and 'clear the display with screen colour Black'. This is followed by a 'call drawDot' block. Below this is an 'Arduino loop forever:' section containing a 'to drawDot' function block. The 'drawDot' function block is a 'draw filled rectangle with colour' block with the following settings: 'screen colour' set to 'Green', 'x' and 'y' variables for position, 'width' set to 1, and 'height' set to 1. A 'return' block is at the bottom of the function.

Damit die „drawDot“ Funktion laufen kann, benötigen wir noch einen Block aus der Kategorie „Display“. Öffne die Kategorie „Display“ und suche den Block "draw sprite to display".

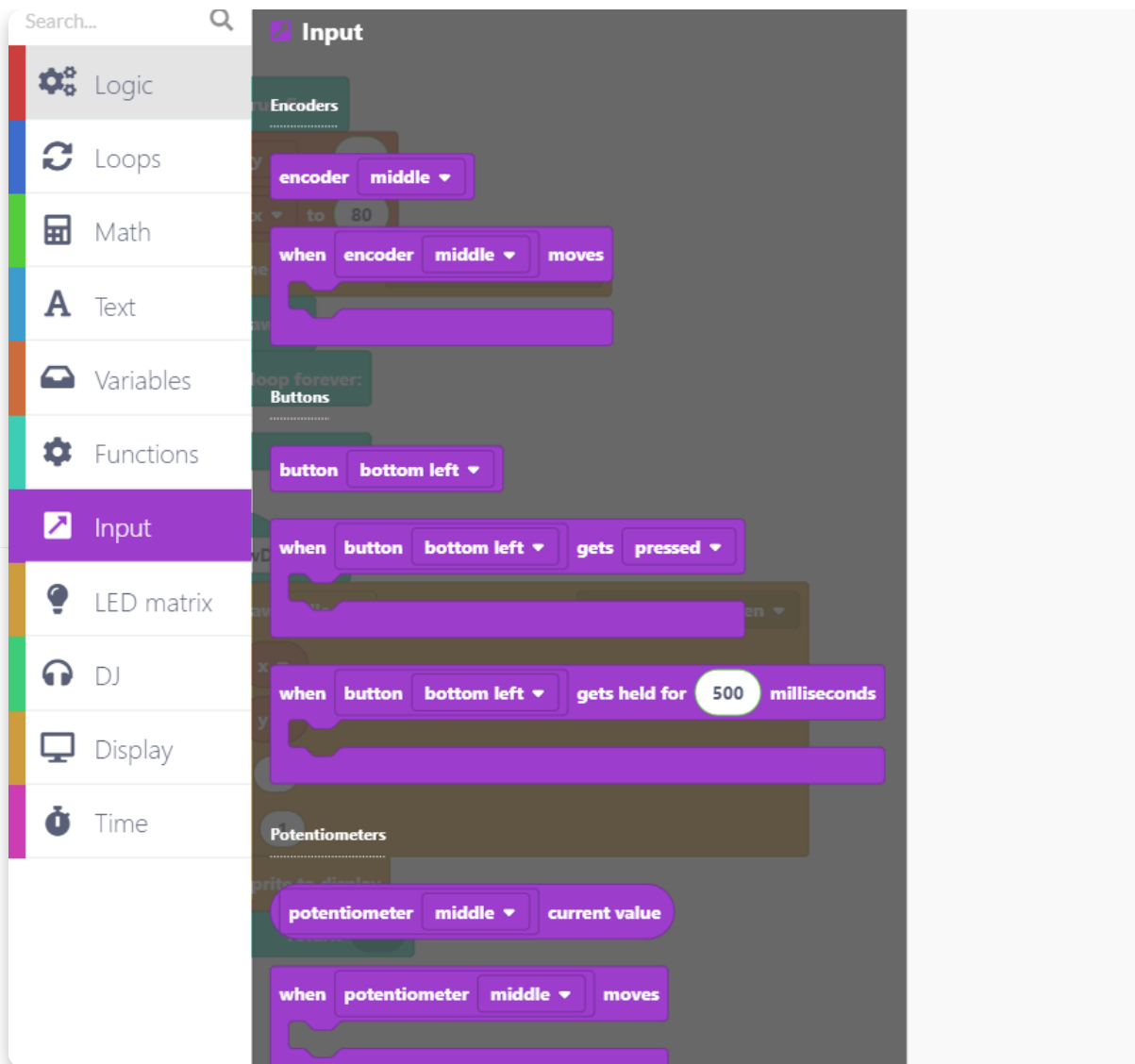


Ziehe sie in den Zeichenbereich und lasse sie am Ende der „drawDot“ Funktion fallen:

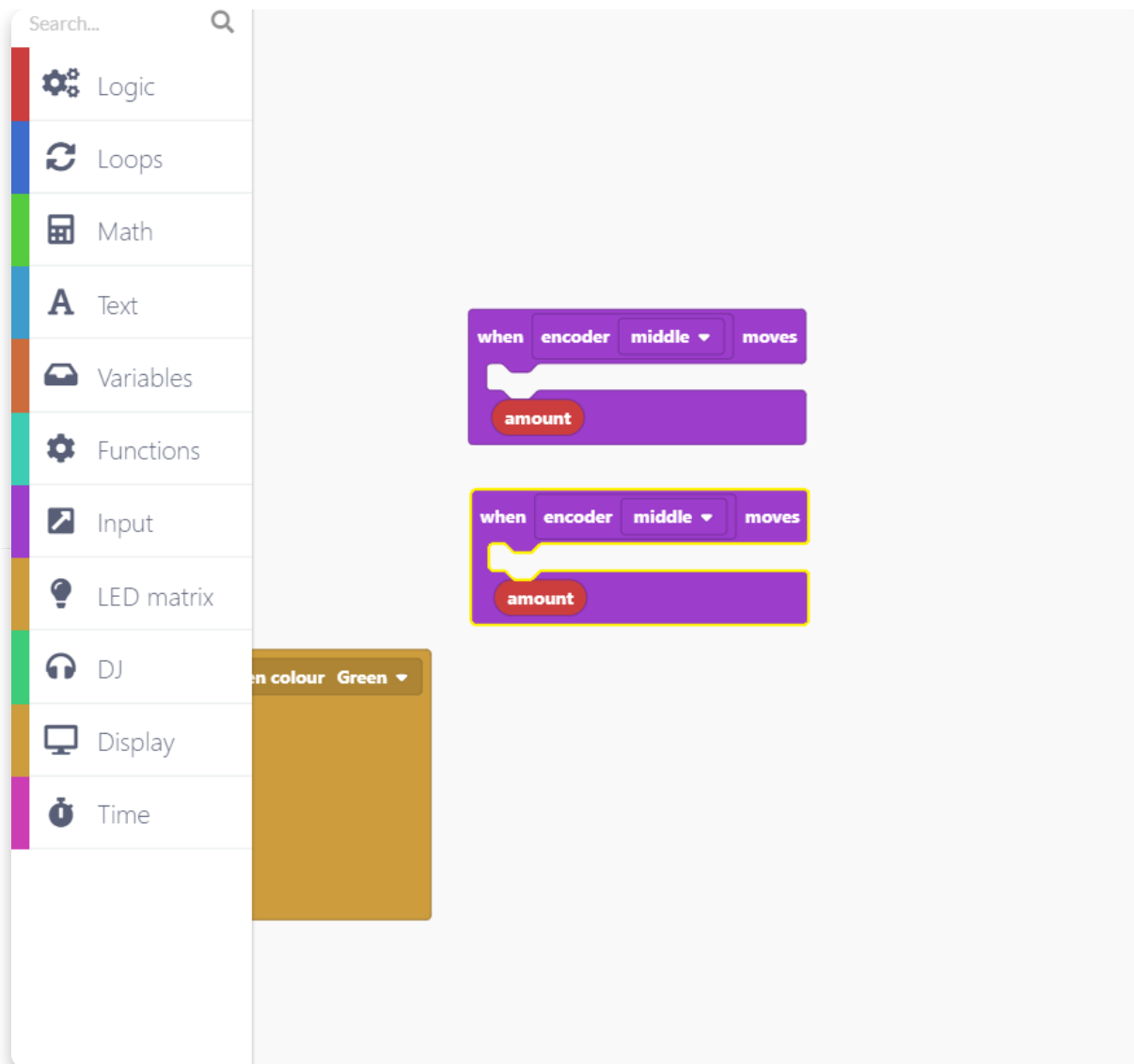


Nun sind wir fast fertig. Es fehlt nur noch eine Kleinigkeit: Wir brauchen eine Eingabequelle, die uns als Stift dienen wird. Lass uns dazu doch die Drehregler verwenden! Ein Regler für die x-Achse und einer für die y-Achse.

Dazu benötigen wir aus der Kategorie „Input“ zweimal den Block "when encoder middle moves".

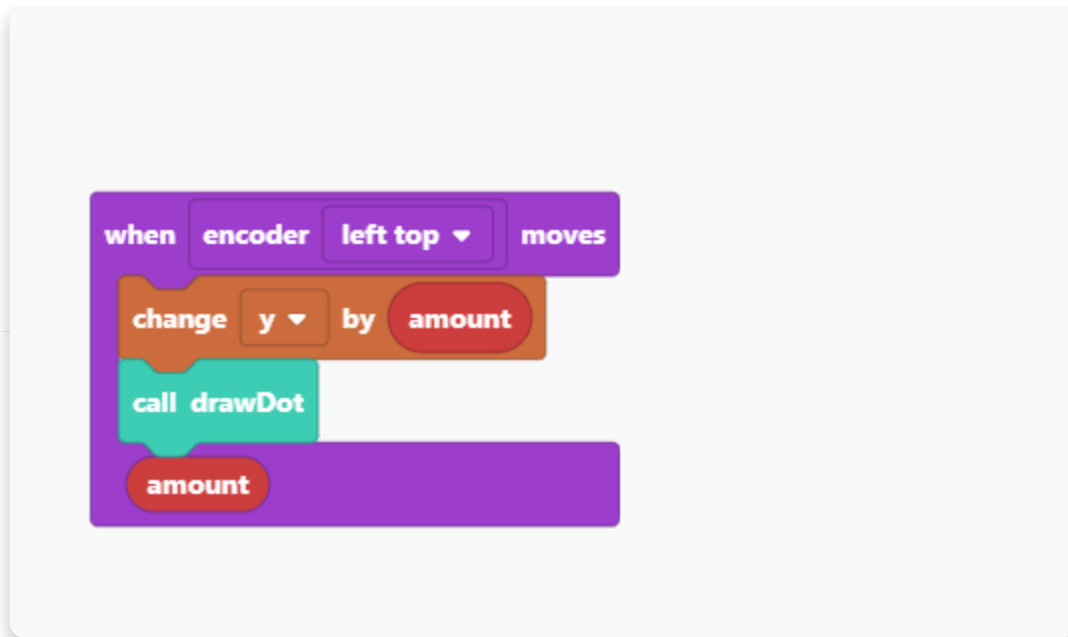


Ziehe zwei dieser Blöcke in den Zeichenbereich und lasse sie fallen:



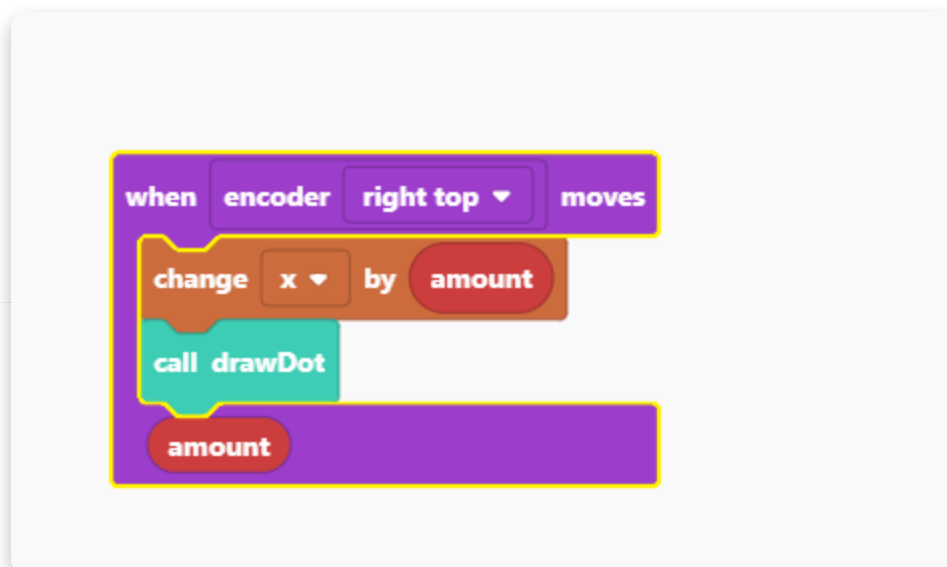
Für die Bewegung auf der y-Achse stellst Du folgendes ein:

- Wähle im Aufklappmenü den Encoder „left top“.
- Nimm aus der Kategorie „Variables“ den Block „change“ und lass ihn in die Encoder-Funktion fallen. Wähle im Aufklappmenü das y aus. Lasse darunter einen Block „change“ aus der Kategorie „Variables“ fallen. Wähle im Aufklappmenü das y aus.
- Ziehe vom unteren Rand der Funktion nun das Feld „amount“ in den „change“ Block.
- Ziehe aus der Kategorie „Functions“ einen „call drawDot“ Block heraus und lasse ihn am unteren Ende der Funktion fallen.



Für die Bewegung auf der x-Achse stellst Du folgendes ein:

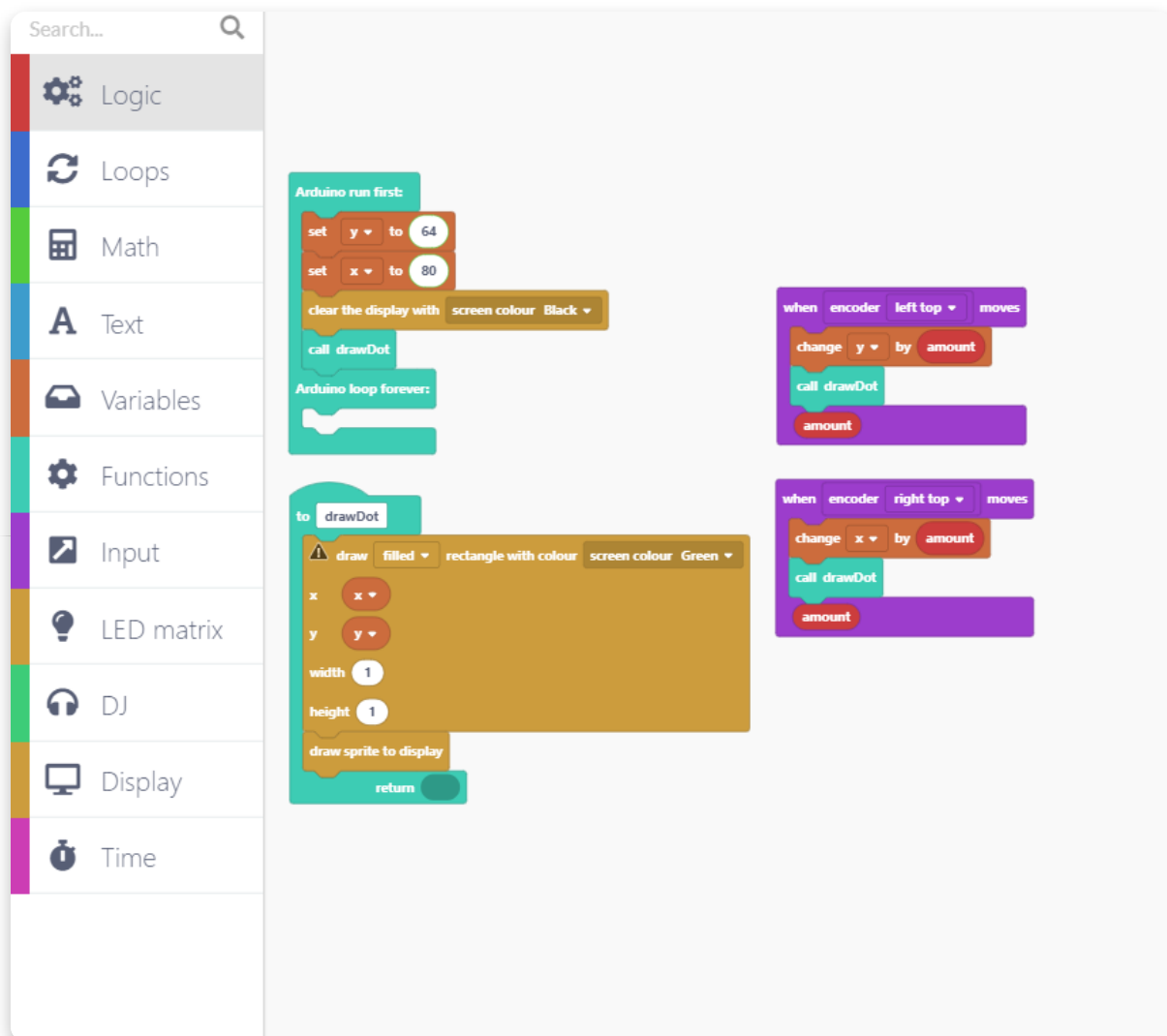
- Wähle im Aufklappmenü den Encoder „right top“.
- Nimm aus der Kategorie „Variables“ den Block „change“ und lass ihn in die Encoder-Funktion fallen. Wähle im Aufklappmenü das x aus. Lasse darunter einen Block „change“ aus der Kategorie „Variables“ fallen. Wähle im Aufklappmenü das x aus.
- Ziehe vom unteren Rand der Funktion nun das Feld „amount“ in den „change“ Block.
- Ziehe aus der Kategorie „Functions“ einen „call drawDot“ Block heraus und lasse ihn am unteren Ende der Funktion fallen.



Das wars. Dein Jay-D ist nun eine leere Leinwand.

Nun musst Du das Projekt nur noch speichern und ihm einen Namen geben, Dann kannst Du es starten. Die Kompilierung kann etwas dauern, da CircuitBlocks erstmalig alle erforderlichen Basisbibliotheken mit übersetzen muss.

So sollte das fertige Projekt am Ende aussehen.



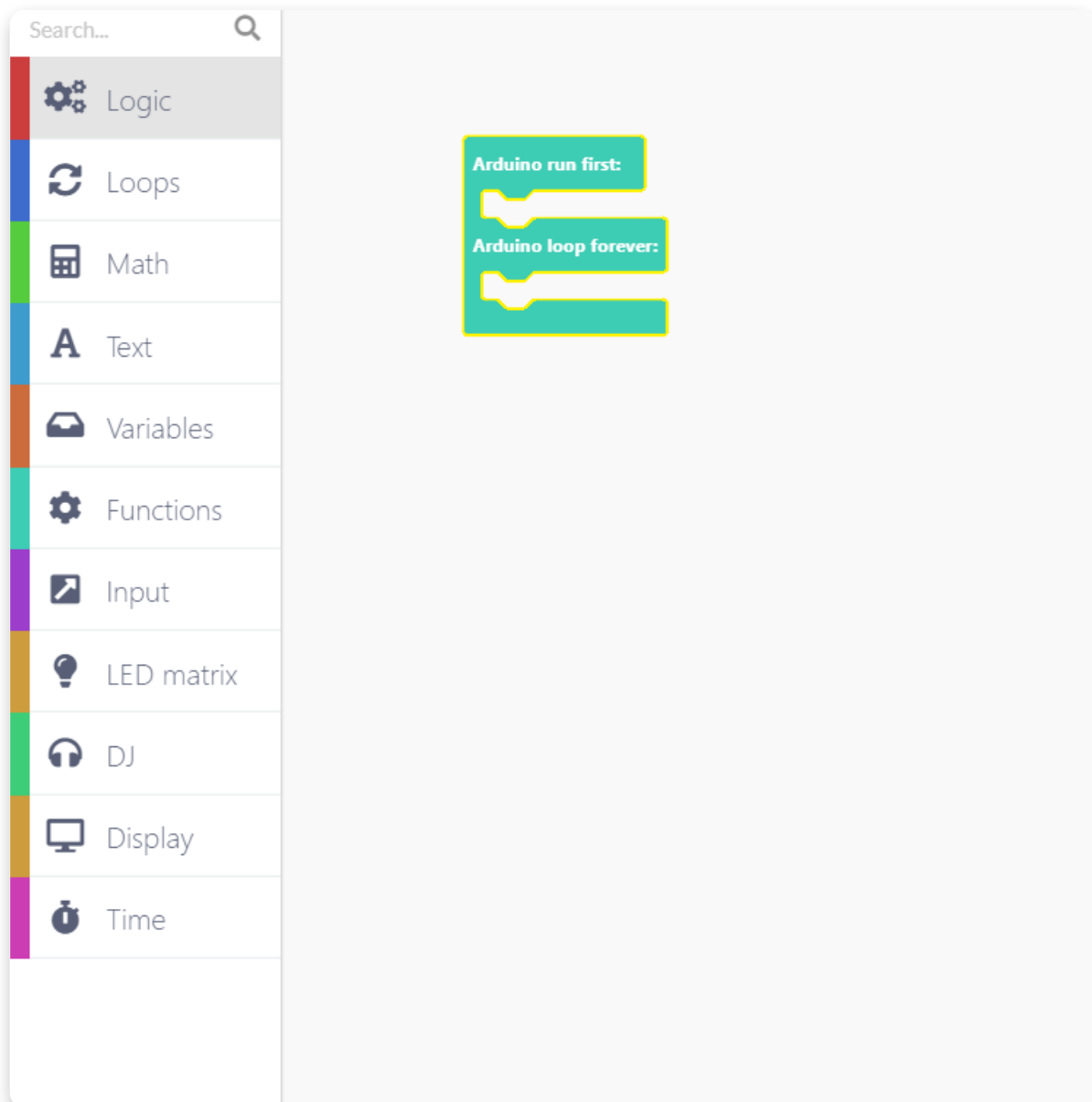
Musik abmischen

Im letztem Abschnitt dieser Anleitung zeigen wir Dir noch ein Beispiel, welches Du mit CircuitBlocks ausprobieren kannst.

Wir programmieren eine DJ-Steuerung mit der Du ein Musikstück mit unterschiedlichen Effekten neu abmischen kannst.

Außerdem werden wir Dir zeigen, wie Du Buchstaben auf das Display bekommst. Das gibt uns dann die Möglichkeit, die Bezeichnung der vorhandenen Soundeffekte auf dem LCD-Display anzuzeigen.

Lass uns zunächst wieder an neues Jay-D Block Projekt anlegen.

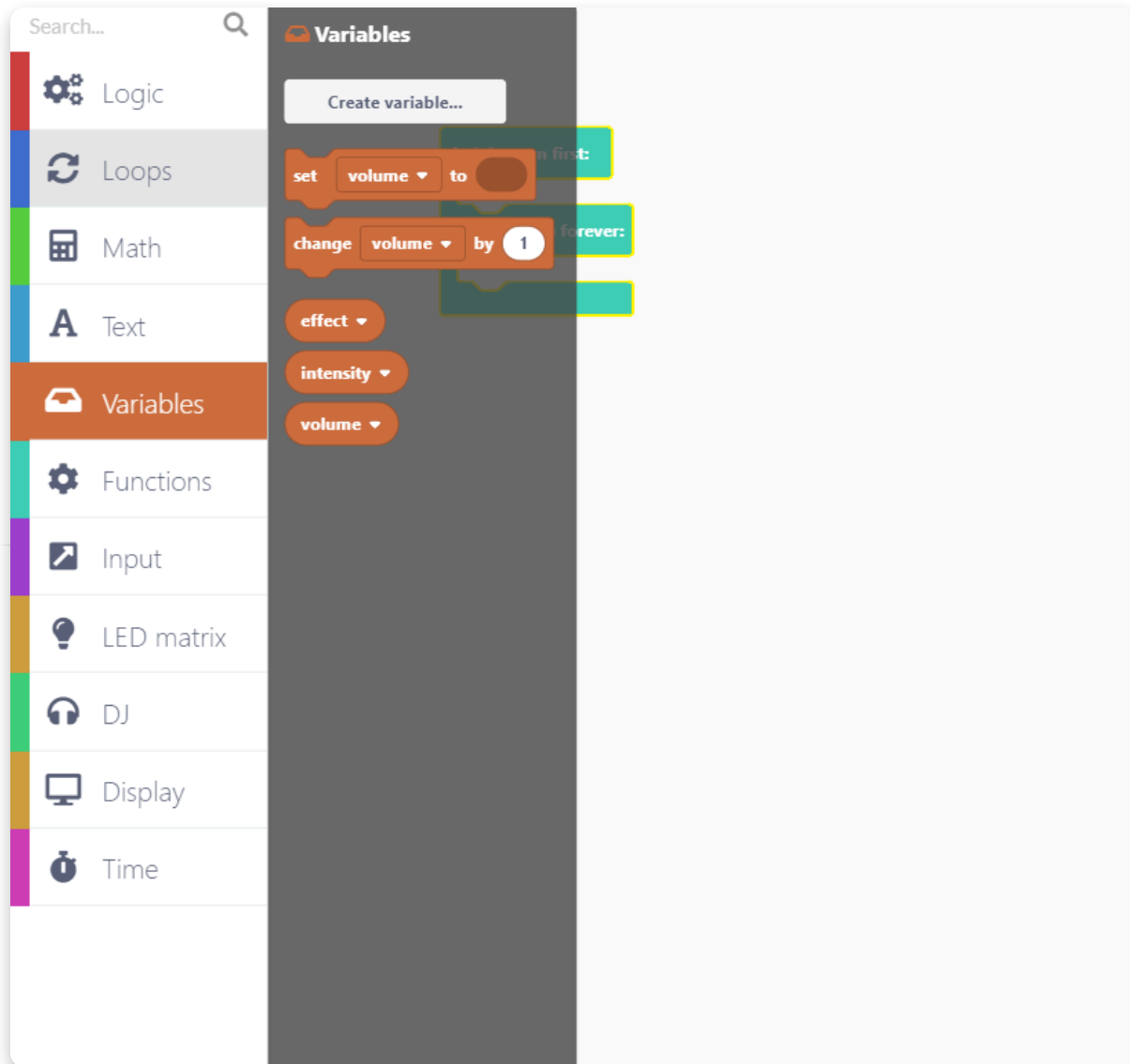


Nun leg bitte drei neue Variablen an. Gib ihnen die Namen: effect, intensity und volume. Diese stehen für: Effekt, Intensität und Lautstärke.

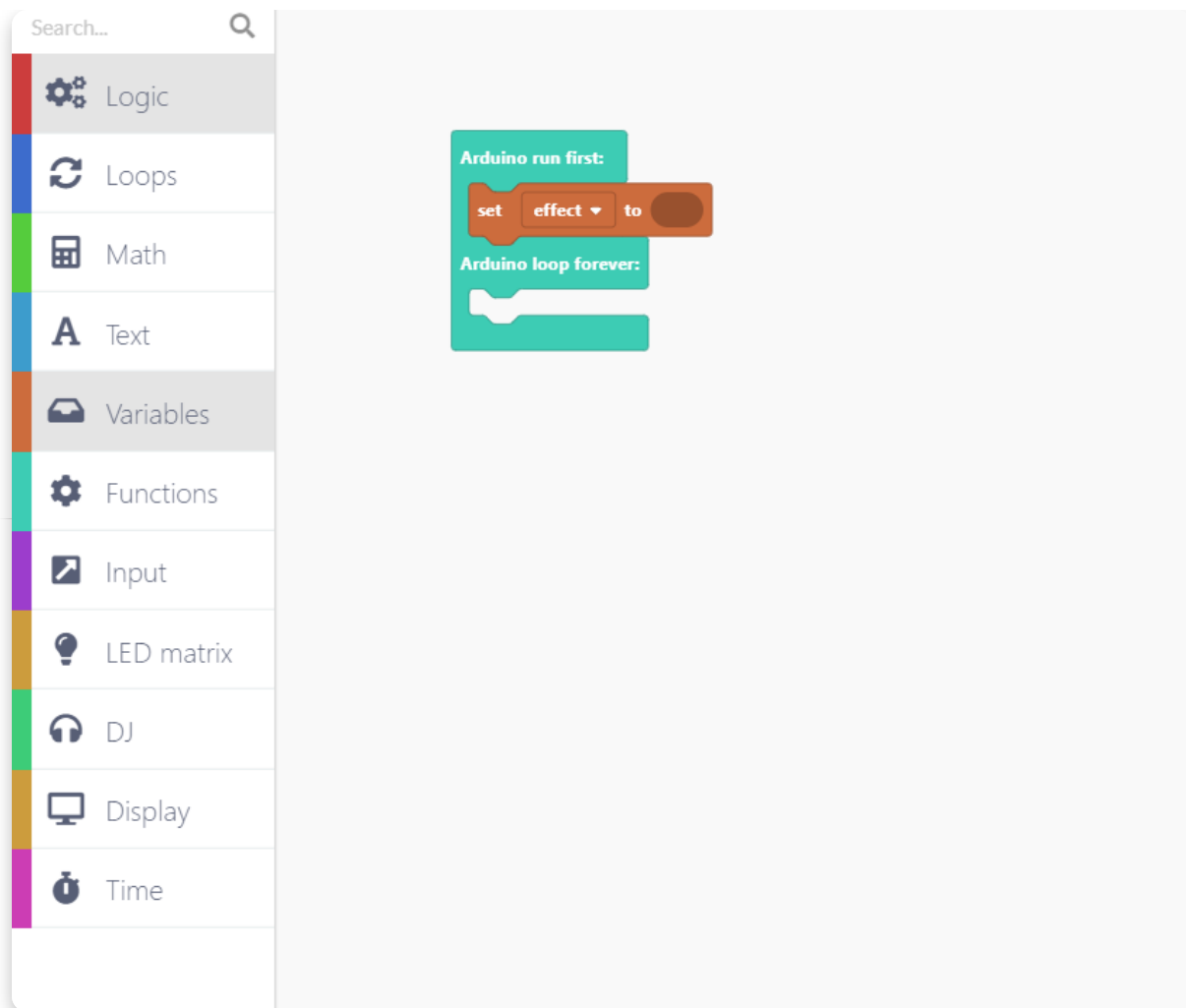
The screenshot shows the CircuitBlocks IDE interface. On the left is a vertical menu with various categories: Logic, Loops, Math, Text, Variables (highlighted in orange), Functions, Input, LED matrix, DJ, Display, and Time. At the top of the menu is a search bar labeled 'Search...'. The main workspace is a light gray area. A dark gray vertical bar on the left side of the workspace contains a 'Create variable...' button. A semi-transparent dialog box is overlaid on the workspace, containing three Arduino-specific blocks: 'Arduino run first:', 'Arduino loop forever:', and an unlabeled block. The dialog box has a yellow border and a dark gray background.

The screenshot shows a dialog box titled 'New variable name:'. It features a text input field with a cursor, a 'Close' button, and a 'Save' button. The dialog box is centered on a dark gray background.

Nachdem Du das gemacht hast, sollten die drei Variablen, wie hier gezeigt, vorhanden sein:

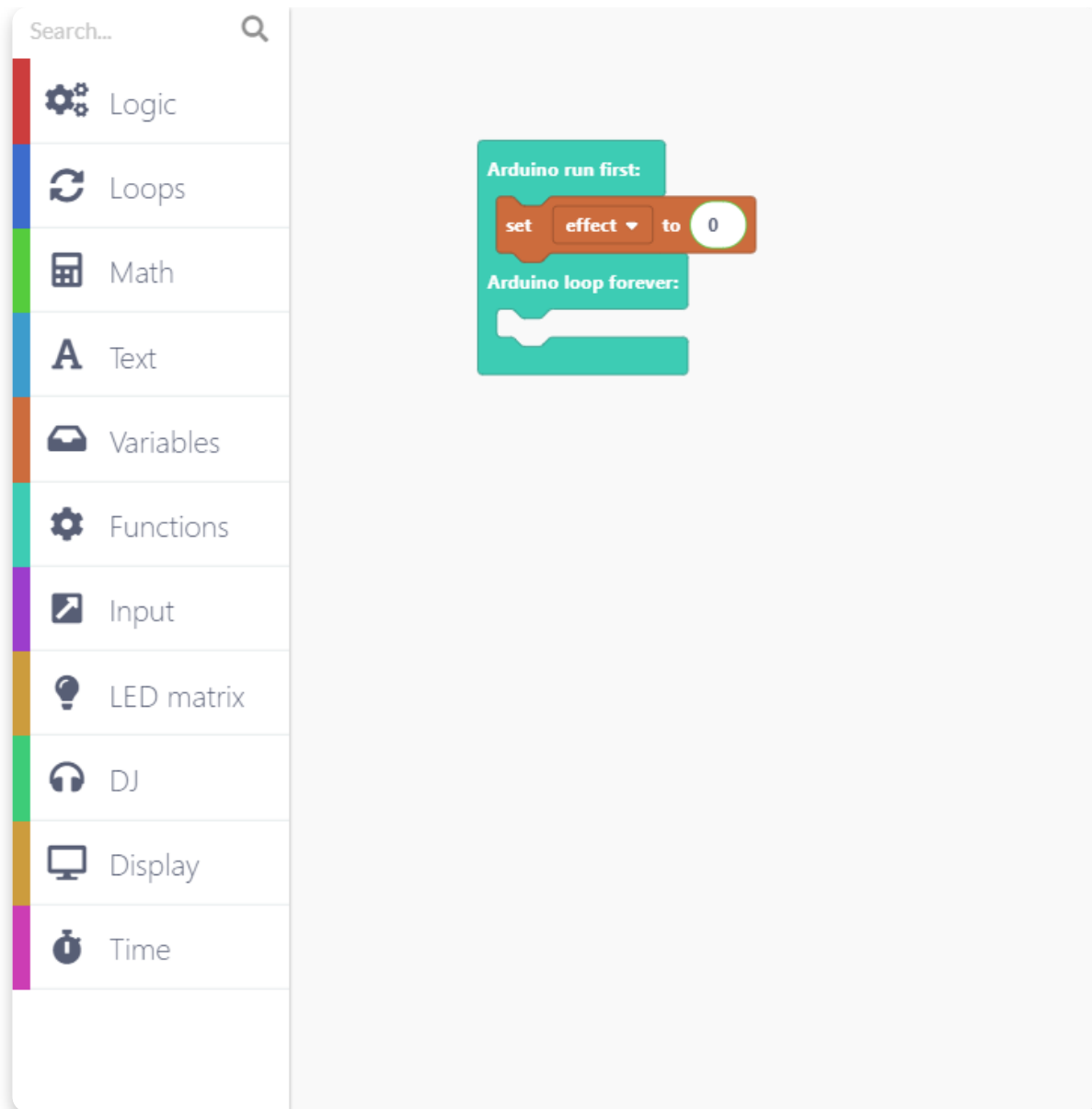


Ziehe den „set“-Block auf den Zeichenbereich und lass ihn in die Lücke unter „Arduino run first:“ fallen. Wähle im Aufklappenmenü den Punkt „effekt“ aus:



Wähle aus der Kategorie „Math“ den Block „123“ aus und füge ihn in den vorigen Block ein. Tausche nun noch die 123 gegen den Wert 0 aus.

The image shows the Circuit Blocks software interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math (highlighted in green), More, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace is titled 'Math' and contains several blocks: a constant block with the value '123', a pi symbol block, an 'e' block, a 'round' block with the value '3.1', a 'random integer from' block with values '1' and '100', a 'random fraction' block, an 'Arithmetic' section with five basic operation blocks (addition, subtraction, multiplication, division, and exponentiation), a 'remainder of' block with values '64' and '10', and a 'Single operand' section with a 'square root' block and the value '9'. A semi-transparent Arduino code editor is visible in the background, showing 'Arduino run first:' and 'Arduino loop forever:' sections.

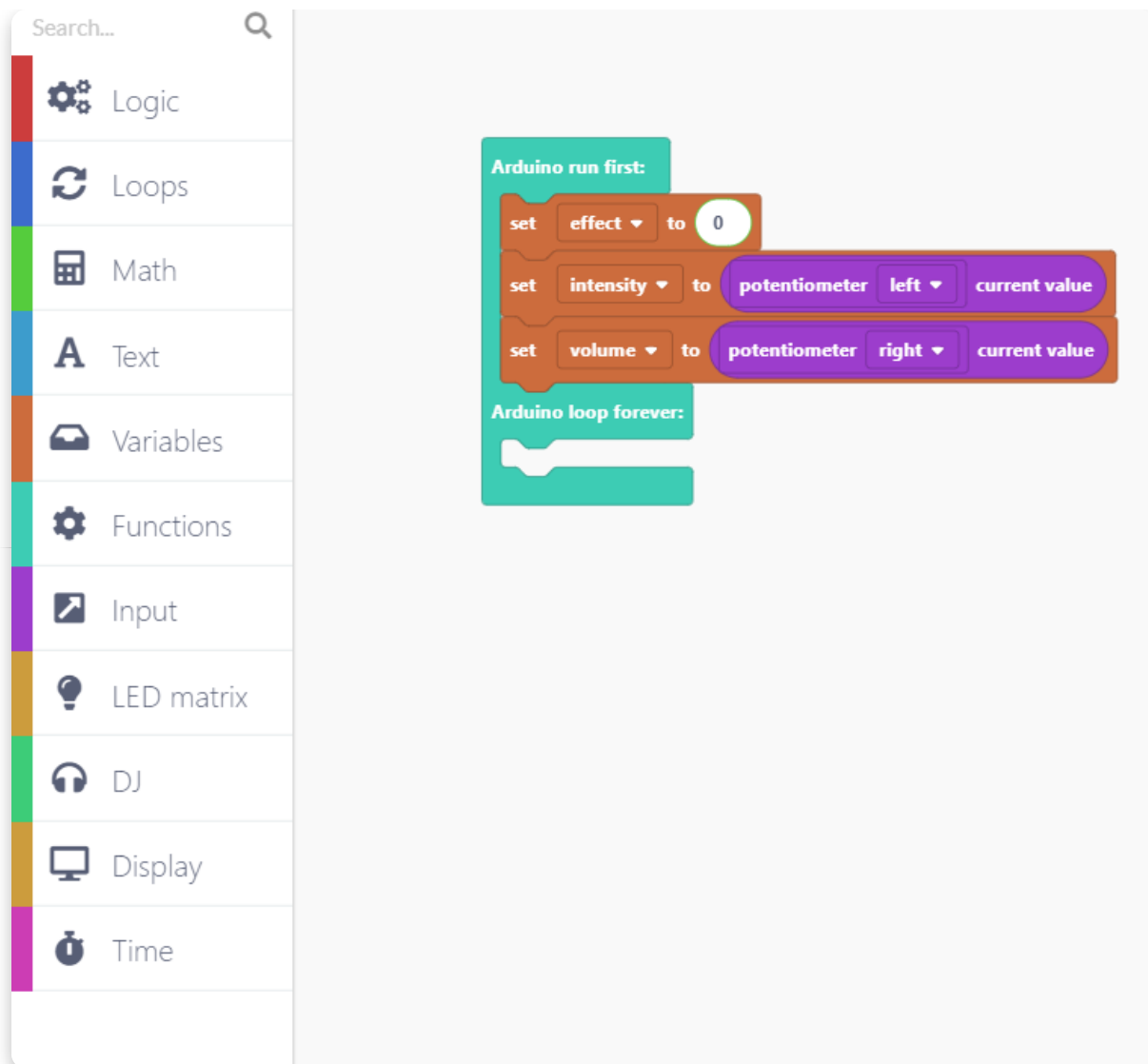


Nun mach bitte das gleich nochmal für die beiden anderen Variablen. In den Aufklappmenüs wählst Du dann entsprechend „intensity“ und für die dritte Variable „volume“ aus.

Intensität und Lautstärke sollen durch die Schieberegler angepasst werden können. Nimm daher aus der Kategorie „Input“ den Block "potentiometer middle current value" und lasse ihn jeweils im Block der Variable für „intensity“ und „volume“ fallen.

The screenshot shows the CircuitBlocks software interface. On the left, a sidebar contains a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input (highlighted in purple), LED matrix, DJ, Display, and Time. The main workspace is titled 'Input' and is divided into three sections: Encoders, Buttons, and Potentiometers. The Encoders section contains a block for 'encoder middle' and a 'when encoder middle moves' block. The Buttons section contains a 'button bottom left' block, a 'when button bottom left gets pressed' block, and a 'when button bottom left gets held for 500 milliseconds' block. The Potentiometers section contains a 'potentiometer middle current value' block, a 'when potentiometer middle moves' block, and a 'potentiometer middle' block. The background of the workspace is dark grey, and the code blocks are purple and white.

Wähle für „intensity“ im Aufklappmenü den Punkt „left“. Im Aufklappmenü für „volume“ nimmst Du aus dem Aufklappmenü den Punkt „right“.

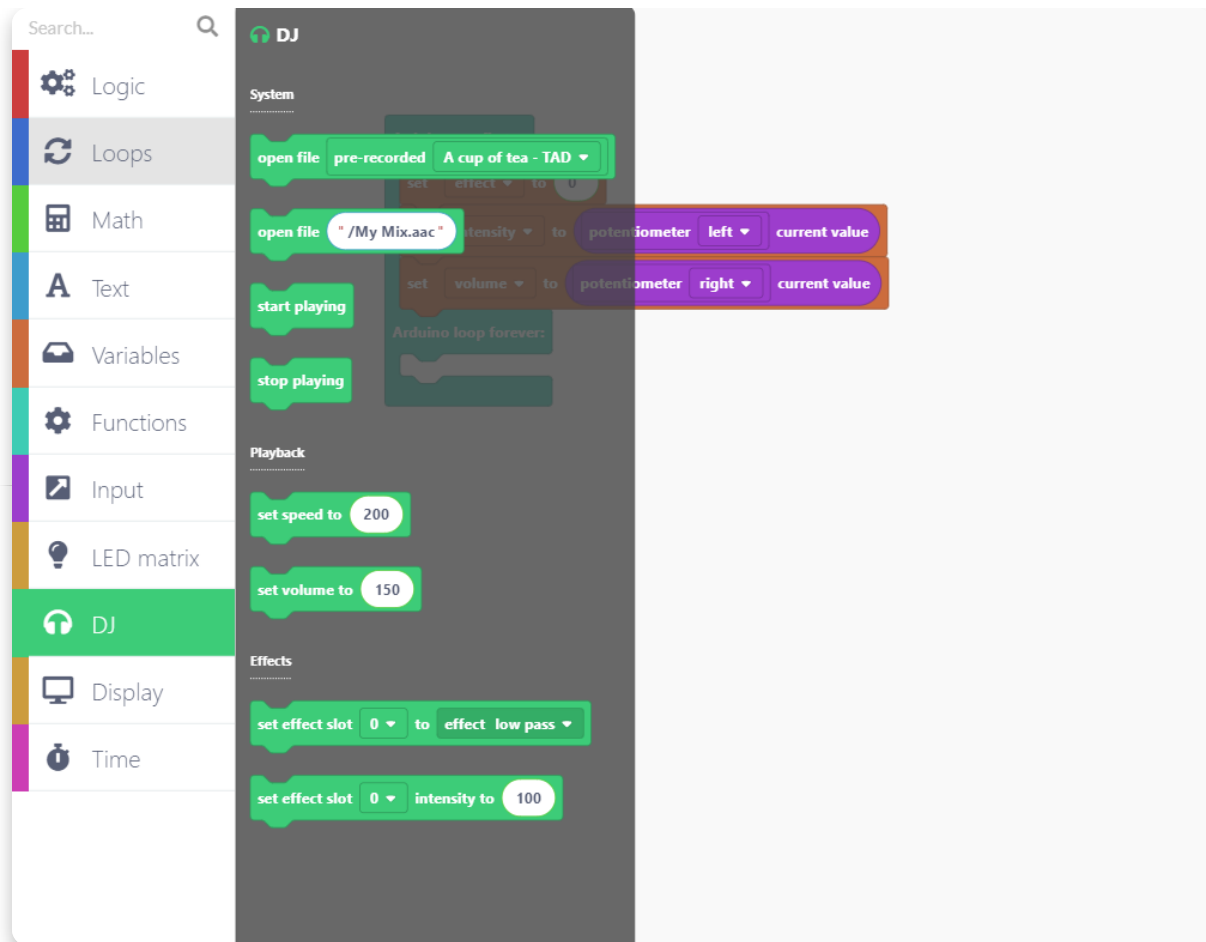


Lass uns eine Datei auf der Speicherkarte öffnen. Das wir der Song sein, den Du neu abmischen möchtest. Suche dazu in der Kategorie „DJ“ den Block "open file pre-recorded".

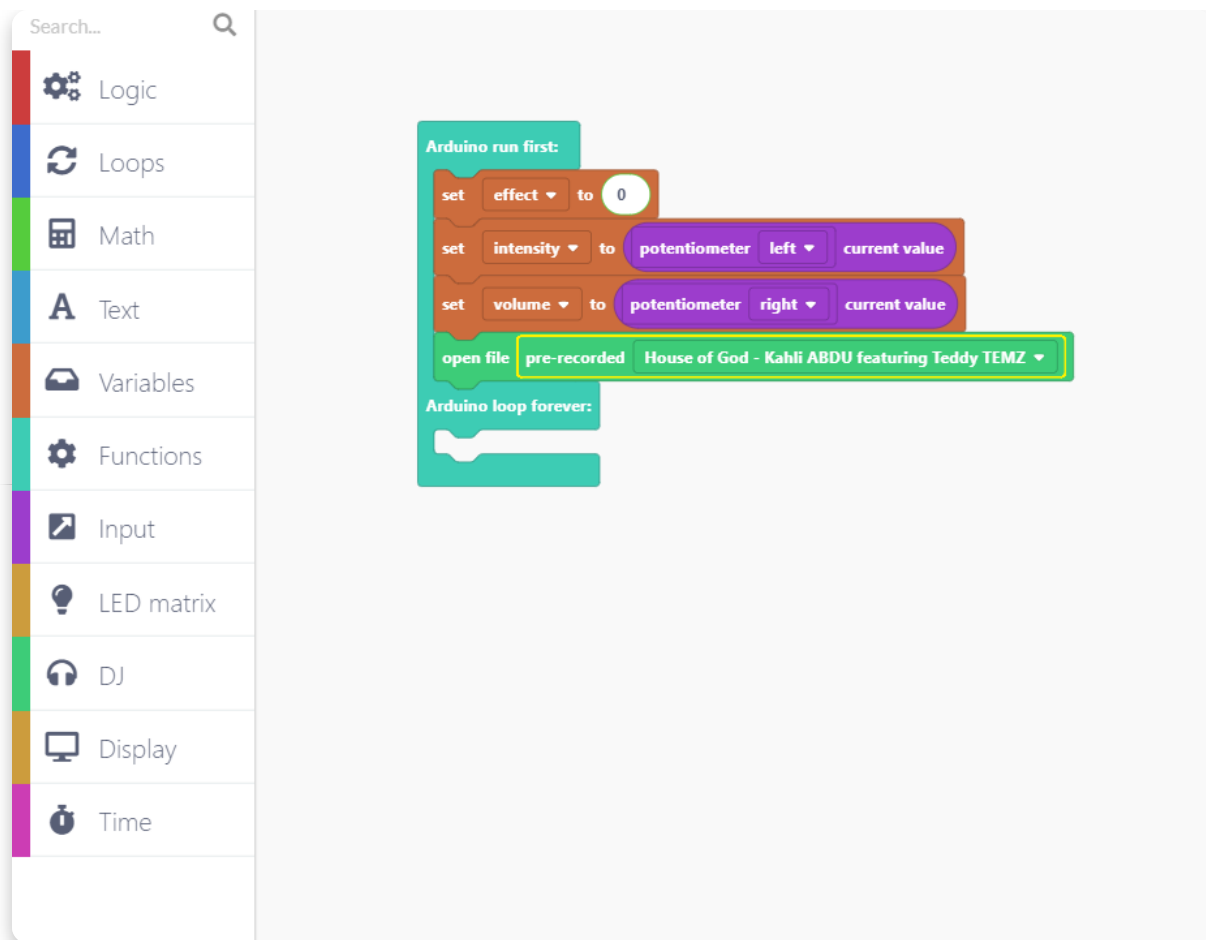
Mit diesem Block kannst Du eine Datei auswählen, die auf der SD-Karte gespeichert ist.

Wenn Du gerne einen Song nehmen möchtest, den Du selber hinzugefügt hast, wähle den Block "open file (Name des Songs)".

Du kannst den Block nun unter den Blöcken fallen lassen, die wir bereits erstellt haben.

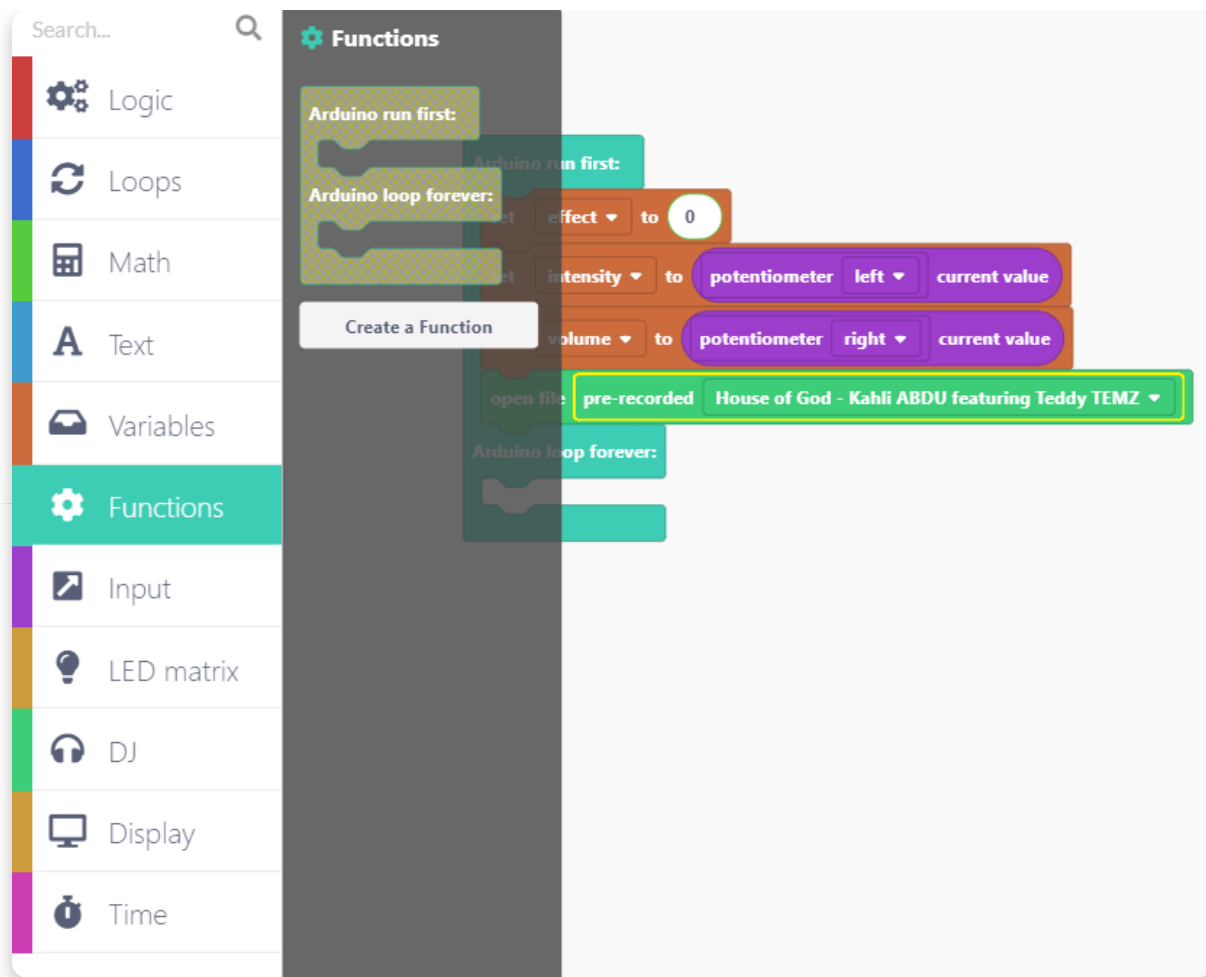


Öffne das Aufklappmenü im „open file...“-Block und such Dir einen Musiktitel aus, den Du neu abmischen möchtest.

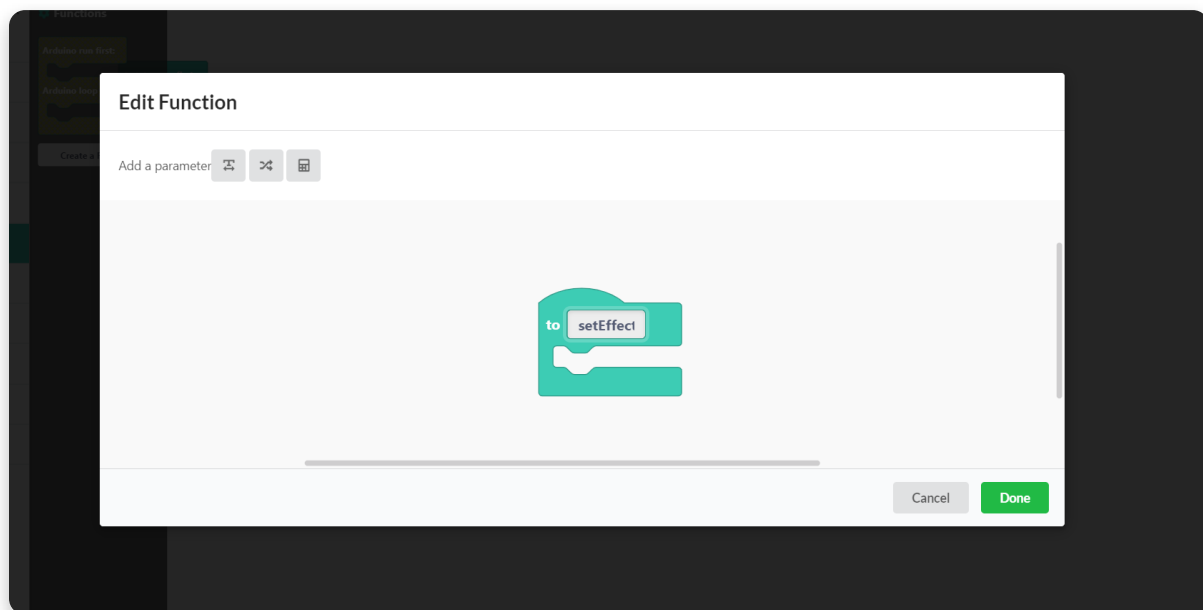


Nachdem Du das getan hast, erstellen wir jetzt eine Funktion, mit denen die Effekte eingestellt werden können.

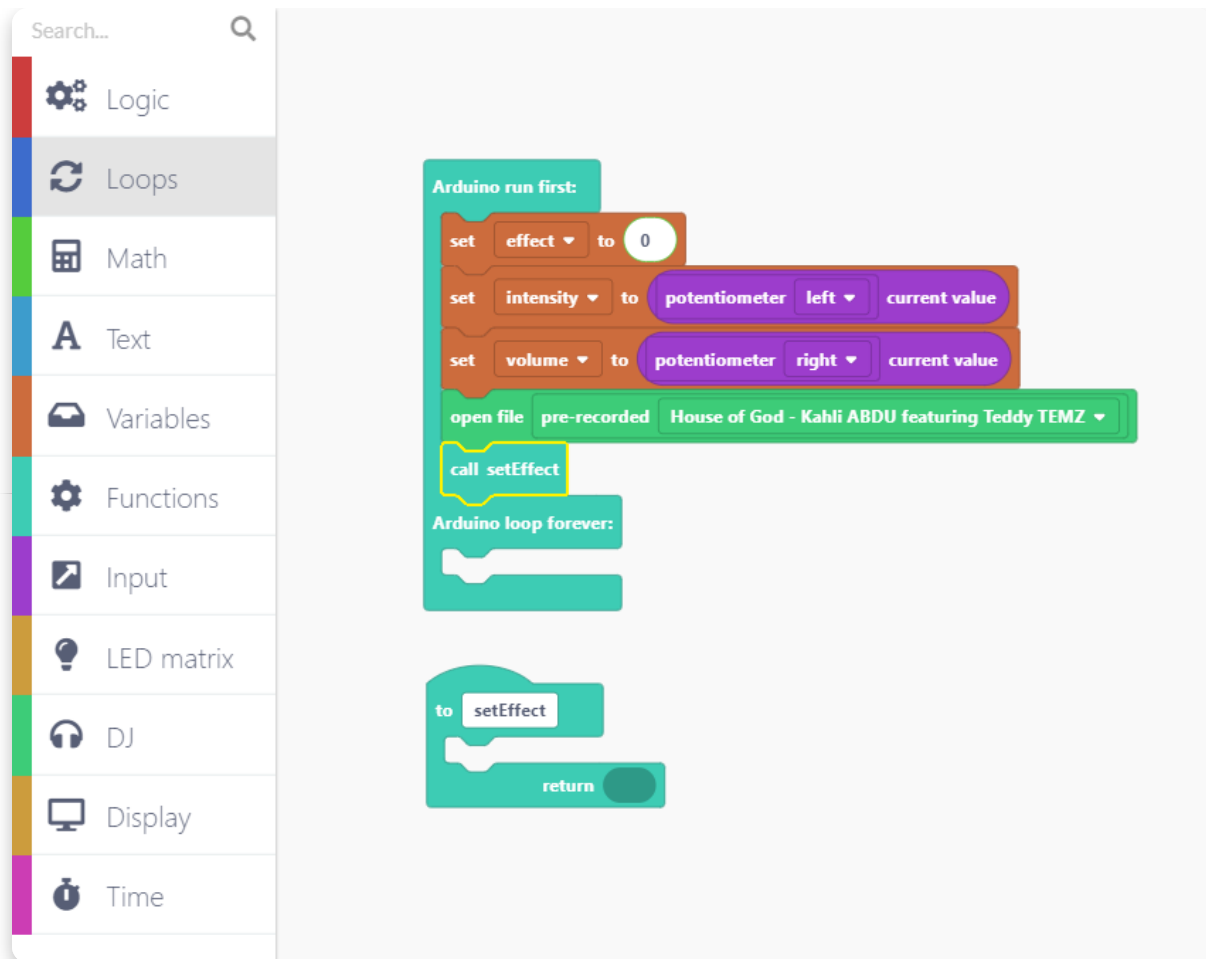
Gehe bitte in die Kategorie „Functions“ und erstelle eine neue Funktion mit „Create a function“.



Gib der neuen Funktion bitte den Namen „setEffect“:



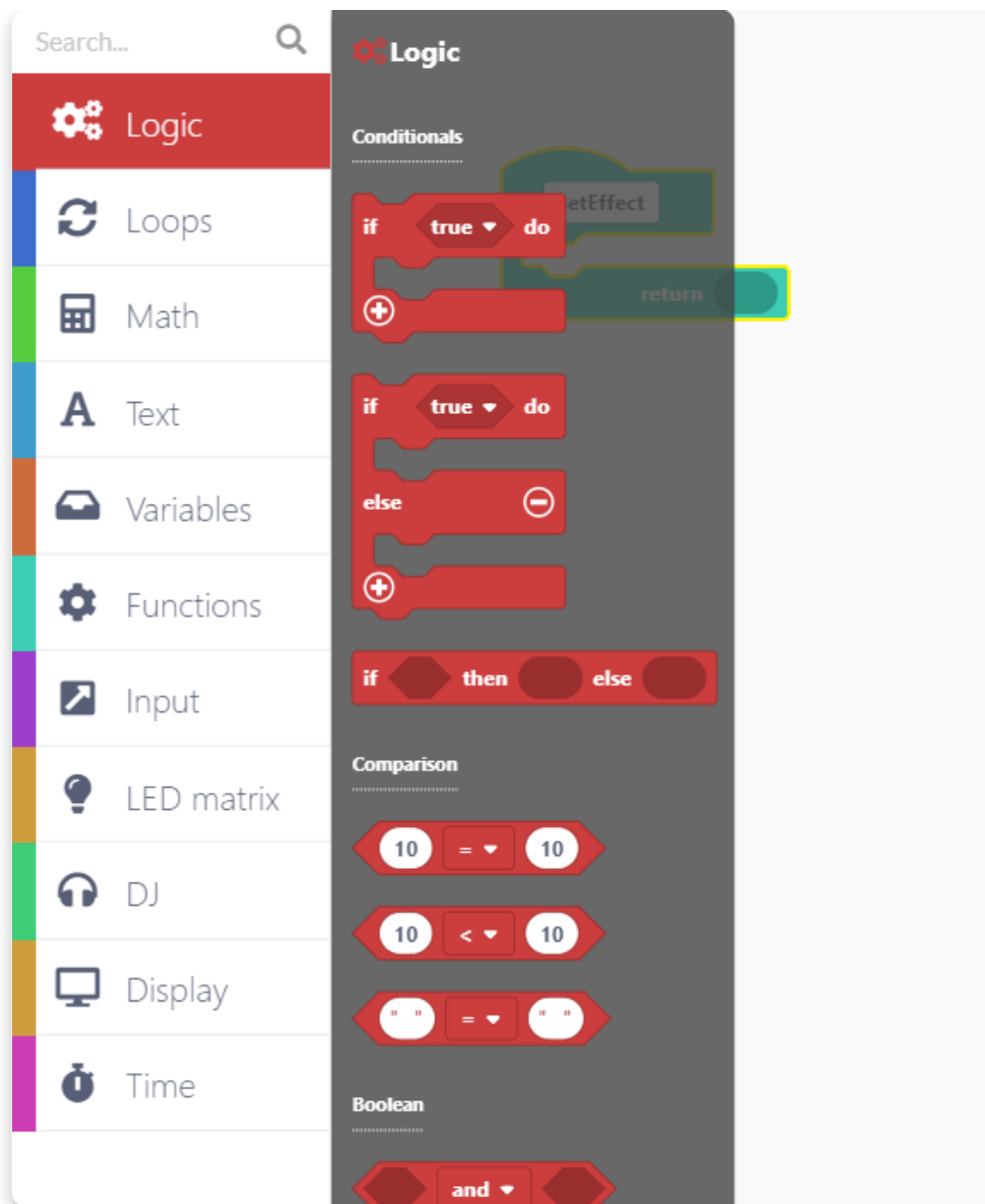
Jetzt erscheint die Funktion unten im Sketch. Vergiss bitte nicht, den Block „call effect“ unter dem Block „open file“ fallen zu lassen. Das sollte dann so aussehen:



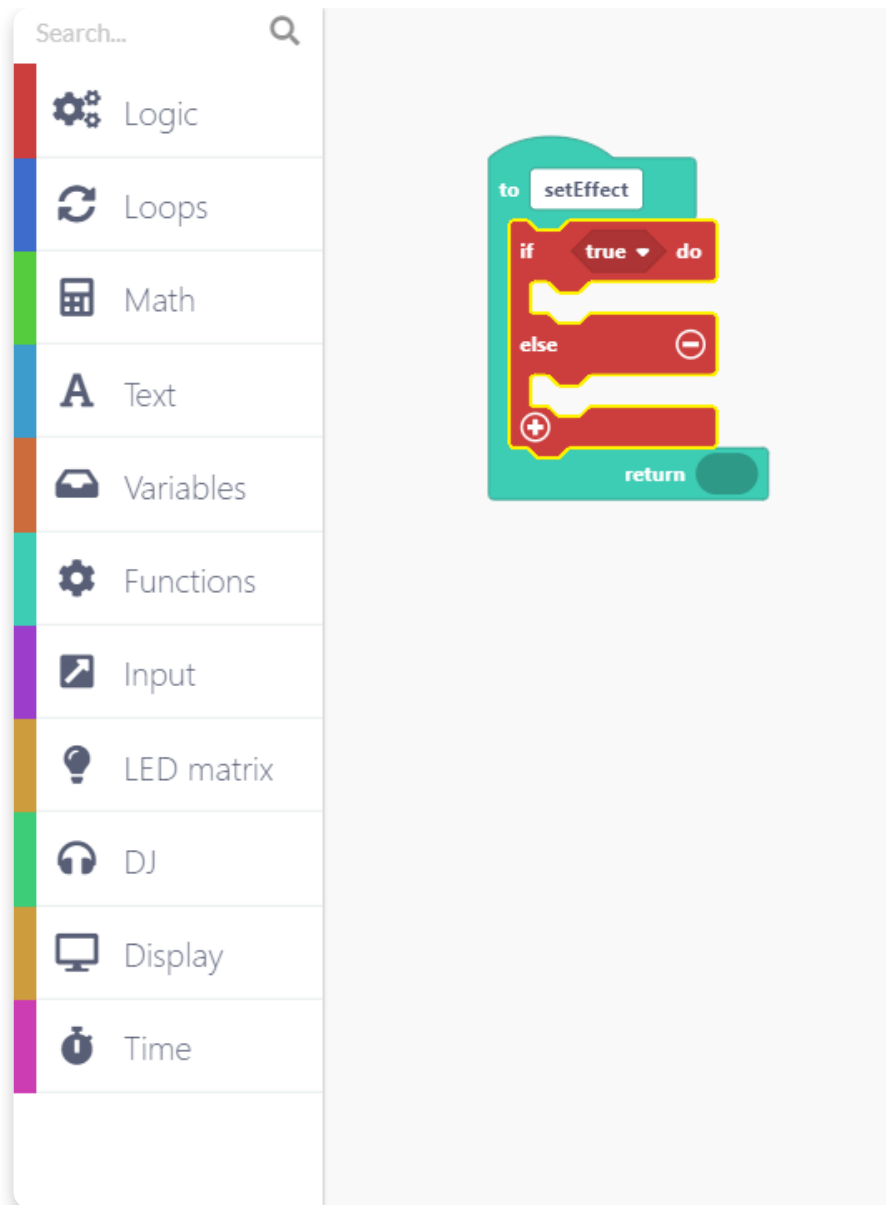
Nun verwenden wir eine logische Funktion (logical function) zum Einstellen der Effekte.

In der Kategorie „Logic“ gibt es fünf verschiedene Blöcke, also schau bitte genau hin:

Öffne die Kategorie „Logic“ und wähle den zweiten Block (den, wo link untereinander „if“ und „else“ steht).

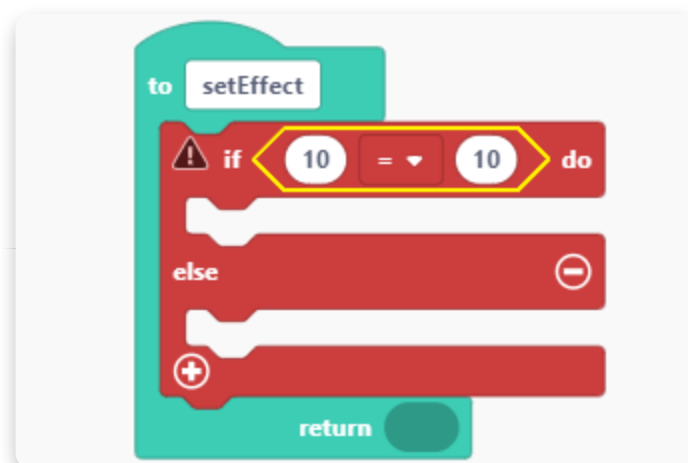


Lass ihn im „setEffekt“-Block fallen.



Anstelle der „true“ Option müssen wir nun einen neuen Logikblock einfügen.

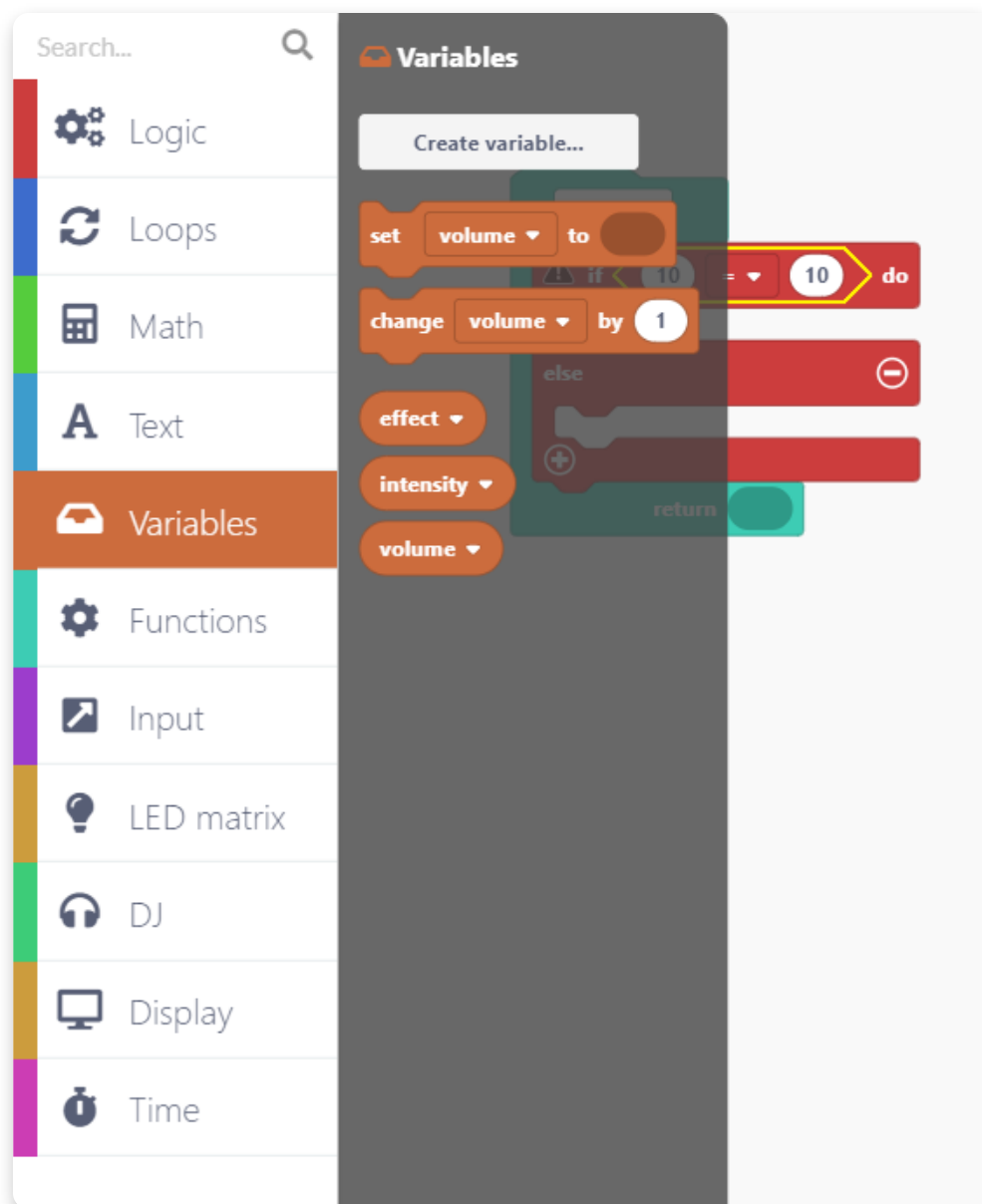
Öffne bitte die Kategorie „Logic“, suche den folgenden Block:



Der benötigte Block vergleicht zwei Zahlenwerte miteinander. Damit wir ihn

verwenden können, müssen wir den ersten Zahlenwert gegen eine „effect“-Variable austauschen. Jede „effect“-Variable hat seinen eigenen Wert. Indem wir die Variable hier einfügen, übernehmen wir für den Vergleich ihren Wert.

Gehe dazu in die Kategorie „Variables“, nimm die „effect“-Variable und lasse sie im linken Vergleichsfeld fallen.

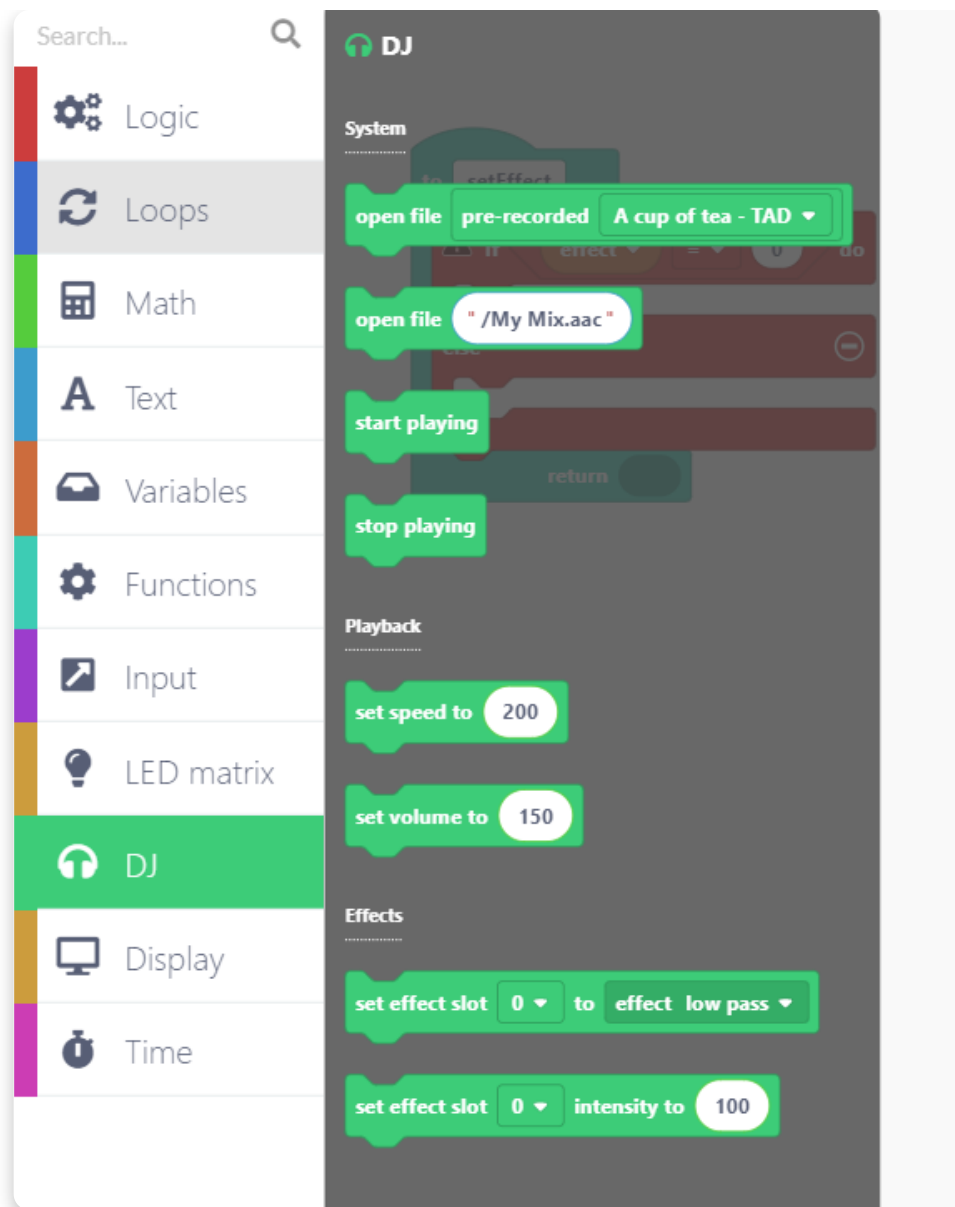


Das Gleichzeichen muss so stehenbleiben. In das Feld rechts schreibst Du anstatt der 10 eine 0.



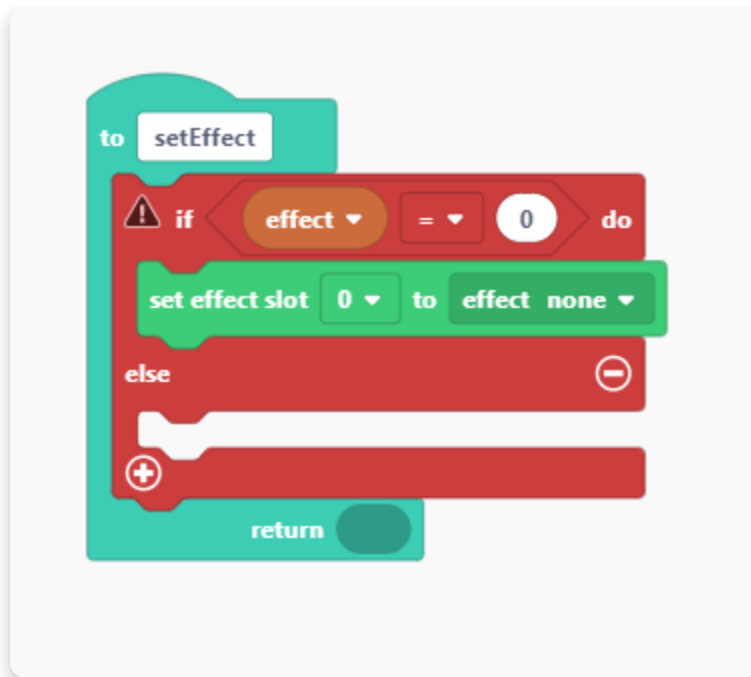
Wenn der Effekt genau 0 entspricht, löst die Vergleichsfunktion eine Aktion aus, die wir nun noch festlegen müssen.

Gehe nun in die Kategorie „DJ“ und such den Block „set effect slot 0 to effect low pass“.



Zieh ihn hierher und lass ihn fallen:

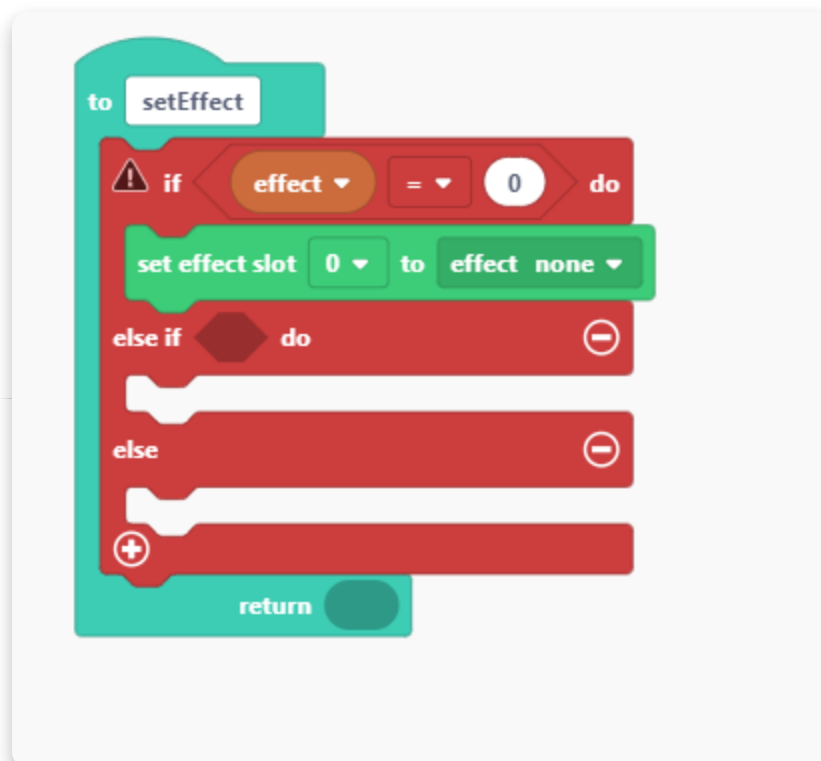
Wähle bitte im ersten Aufklappmenü den Wert „0“ und im zweiten Menü „effect none“.



Die Einstellung für „keinen Effekt“ (also: der Effekt ist ausgeschaltet) haben wir nun fertig.

Das Wiederholen wir nun noch ein paar Mal. Allerdings nun für verschiedene Effekte.

Um den Block um eine neuen Effekt zu erweitern, klicke bitte jedes Mal vorher auf das kleine Pluszeichen am unteren Rand des Blocks:



Wir benötigen (für vier neue Effekte) vier Wiederholungen:

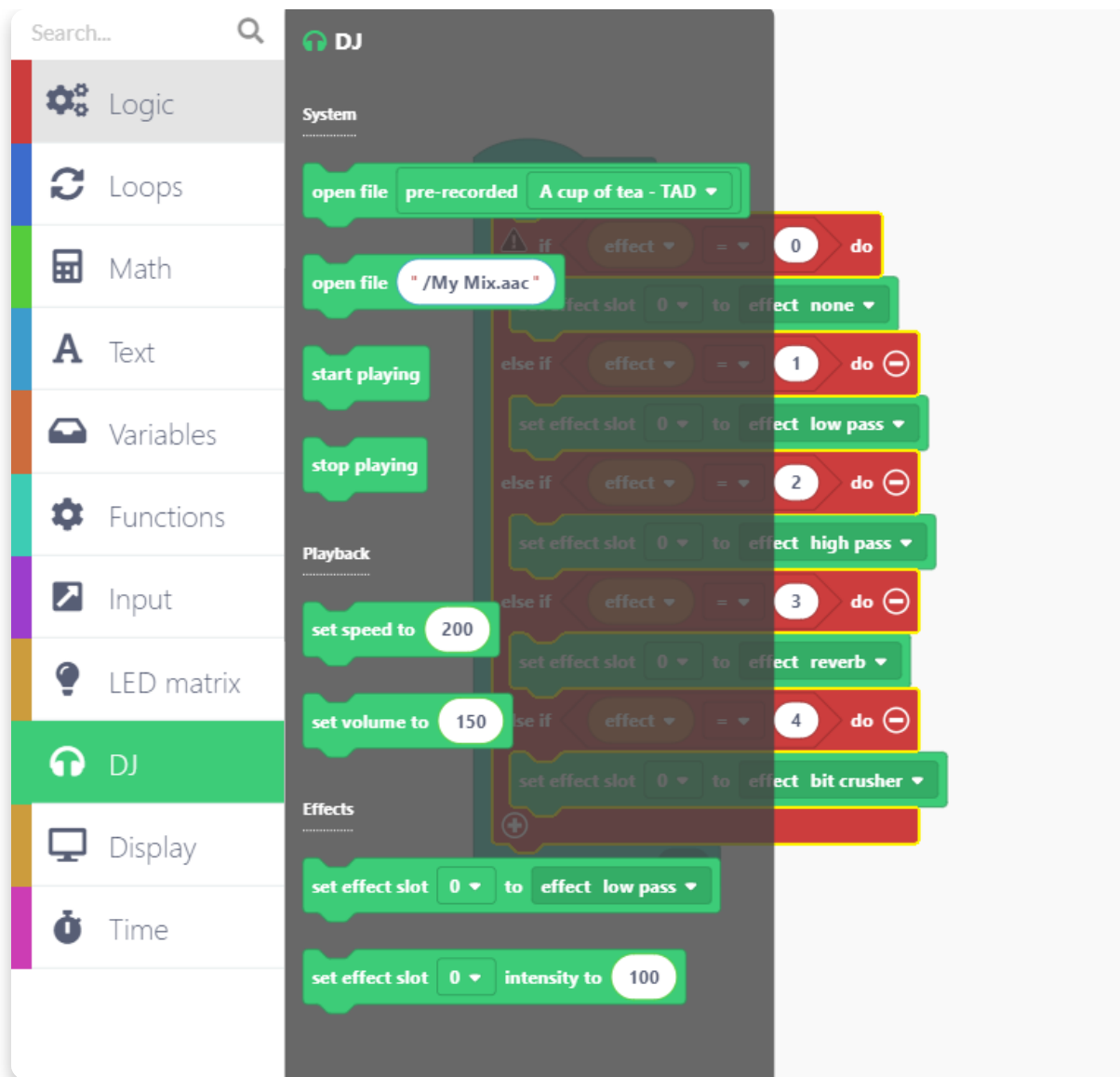
1. low pass
2. high pass
3. reverb
4. bit crusher

Schließe nun die Bearbeitung des Blocks ab, indem Du auf das kleine Minuszeichen neben „else“ klickst.

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace displays a 'to setEffect' block. Inside this block, there is a series of 'if-else if' conditions. Each condition checks the value of 'effect' against a number (0, 1, 2, 3, 4). If the condition is true, it sets 'effect slot 0' to a specific effect: 'none', 'low pass', 'high pass', 'reverb', and 'bit crusher' respectively. The 'else if' blocks have a minus sign in a circle next to them, indicating they are mutually exclusive. The block ends with a 'return' block.

Zum Schluss fügen wir noch einen Block hinzu, der den Effekt mit der Intensität verknüpft. Das machen wir wie vorhin, mit dem zugehörigen „Variable“-Block.

Öffne die Kategorie „DJ“ und suche den Block "set effect slot 0 intensity to 100".



Lasse den Block hier fallen:

The image shows the CircuitBlocks IDE interface. On the left is a search sidebar with a search bar and a magnifying glass icon. Below the search bar are several category buttons: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The 'Loops' category is currently selected. The main workspace on the right contains a 'to setEffect' block. Inside this block, there is a series of 'if' and 'else if' blocks. Each 'if' block checks the value of the 'effect' variable against a number (0, 1, 2, 3, 4). If the condition is met, a corresponding 'set effect slot 0' block is executed, setting the slot to a specific effect: 'effect none', 'effect low pass', 'effect high pass', 'effect reverb', and 'effect bit crusher'. The final 'set effect slot 0' block is highlighted in yellow and has 'intensity to 100' instead of a number. A red '+' icon is visible next to this block. The 'return' block at the bottom of the 'setEffect' block is also visible.

Anstelle der „100“ wird die „intensity“-Variable eingebaut. Dazu gehe bitte in die Kategorie „Variables“ und suche den Block der „intensity“-Variable. Ziehe ihn in den Zeichenbereich und lasse ihn im zuletzt bearbeiteten Block fallen (da, wo bisher die 100 steht).

Wenn Die Zahl dabei aus dem Block herauspringt, ziehe sie bitte einfach in den Bereich wo die Kategorien sind. So wird sie aus dem Sketch entfernt.

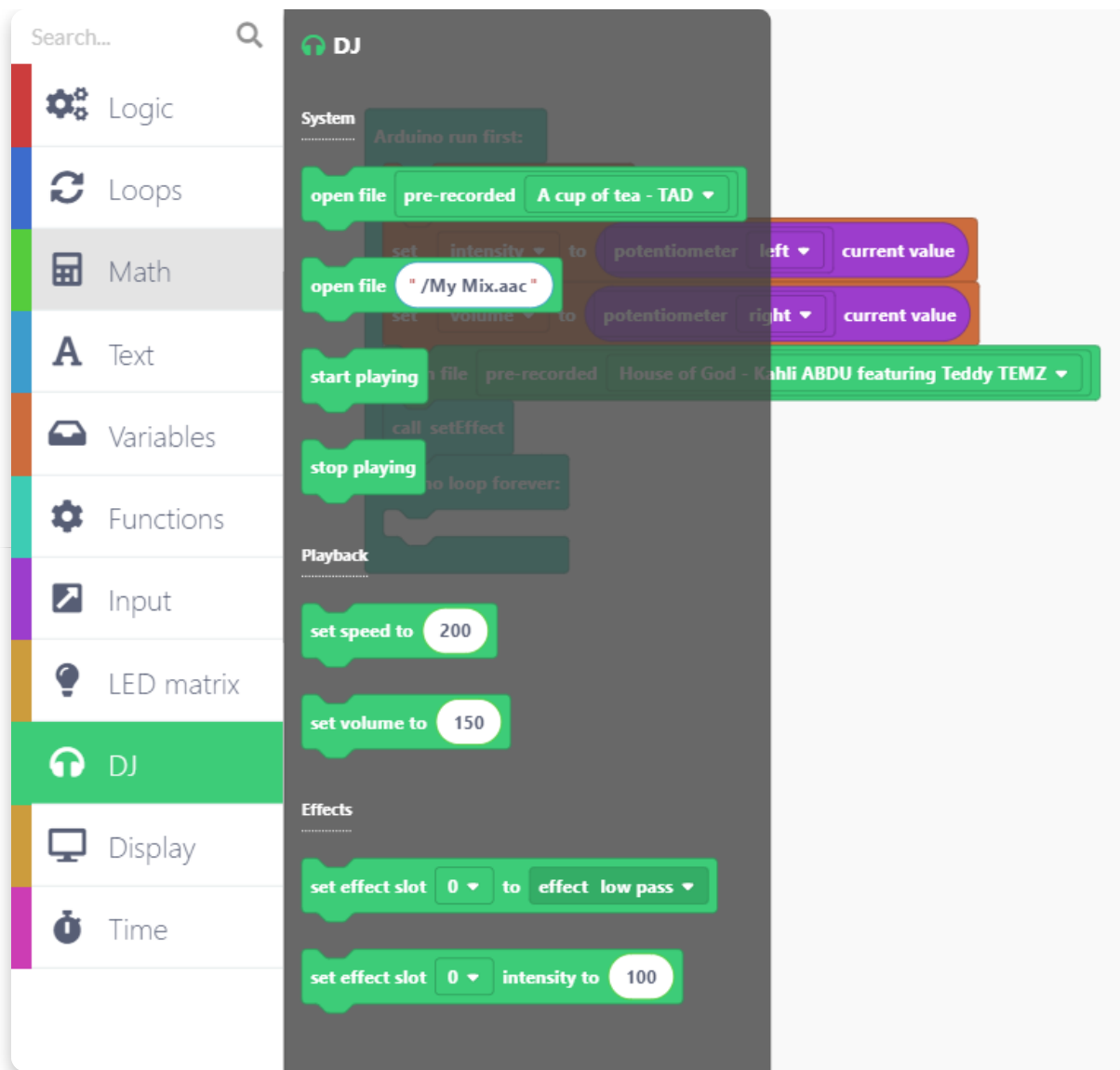
The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables (highlighted), Functions, Input, LED matrix, DJ, Display, and Time. The main workspace is titled 'Variables' and contains a script. The script starts with a 'Create variable...' button, followed by a 'set volume to' block. Below that is a 'change volume by 1' block. The script then enters a series of 'else if' blocks, each checking the 'effect' variable against a number (0, 1, 2, 3, 4). Each 'else if' block is followed by a 'do' block that sets 'effect slot 0' to a specific effect: 'effect none', 'effect low pass', 'effect high pass', 'effect reverb', and 'effect bit crusher'. The final 'else if' block (checking for 4) is followed by a 'do' block that sets 'effect slot 0 intensity to 100'. The script ends with a 'return' block.

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The 'Loops' category is currently selected. The main workspace displays a function block named 'setEffect'. The function's logic is as follows: it starts with an 'if' block checking if the 'effect' variable is equal to 0. If true, it sets 'effect slot 0' to 'effect none'. If the 'effect' variable is equal to 1, it sets 'effect slot 0' to 'effect low pass'. If the 'effect' variable is equal to 2, it sets 'effect slot 0' to 'effect high pass'. If the 'effect' variable is equal to 3, it sets 'effect slot 0' to 'effect reverb'. If the 'effect' variable is equal to 4, it sets 'effect slot 0' to 'effect bit crusher'. Finally, it sets 'effect slot 0' intensity to the 'intensity' variable and then returns.

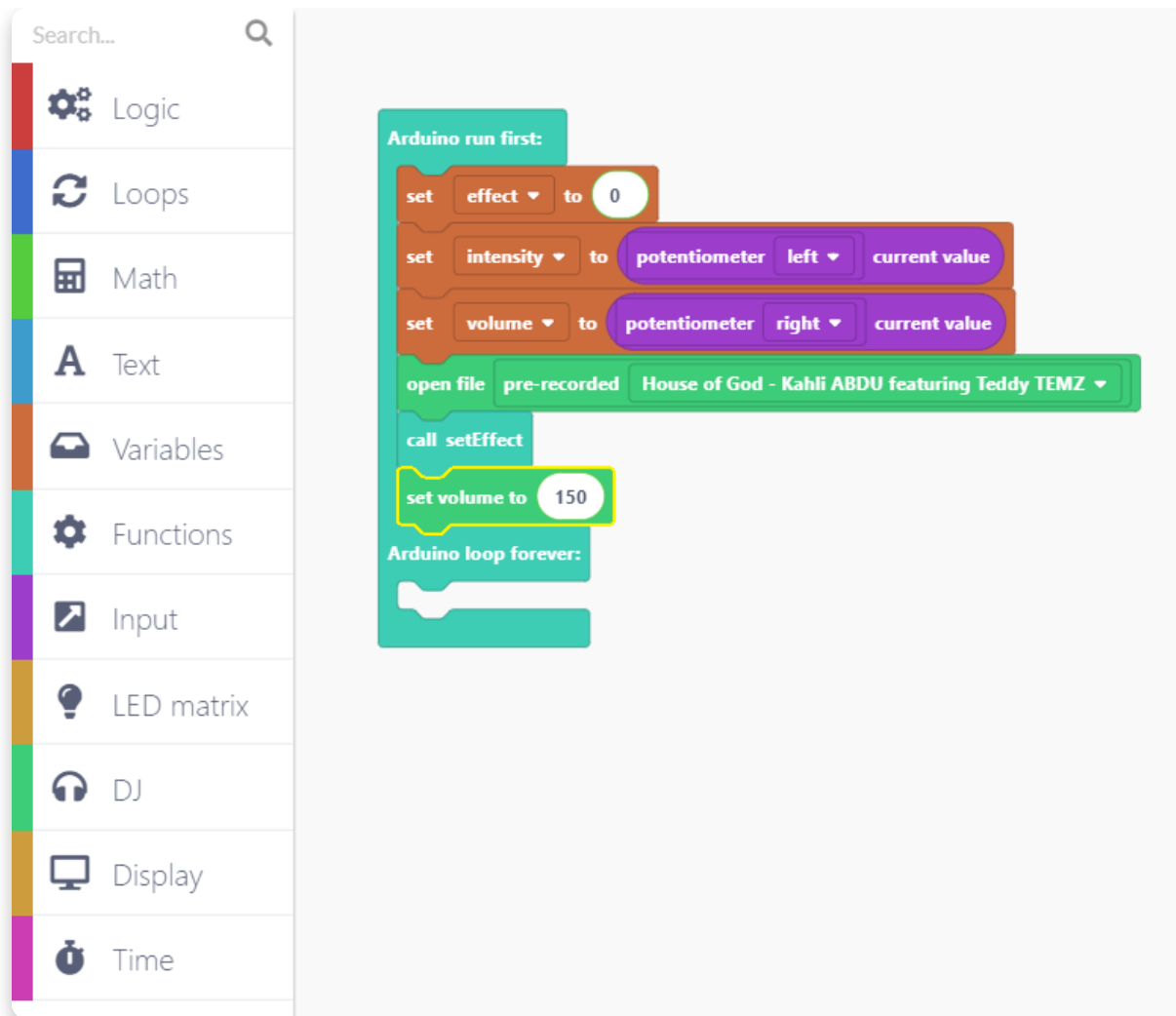
Die Funktion „setEffect“ ist nun fertig!

Lass uns jetzt zum ersten (Arduino run first) Block zurückgehen und den Lautstärke-Block hinzufügen.

Öffne daz die Kategorie „DJ“ und suche den Block „set volume to“.



Lasse ihn bitte hier fallen:



The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace contains an Arduino code block with the following structure:

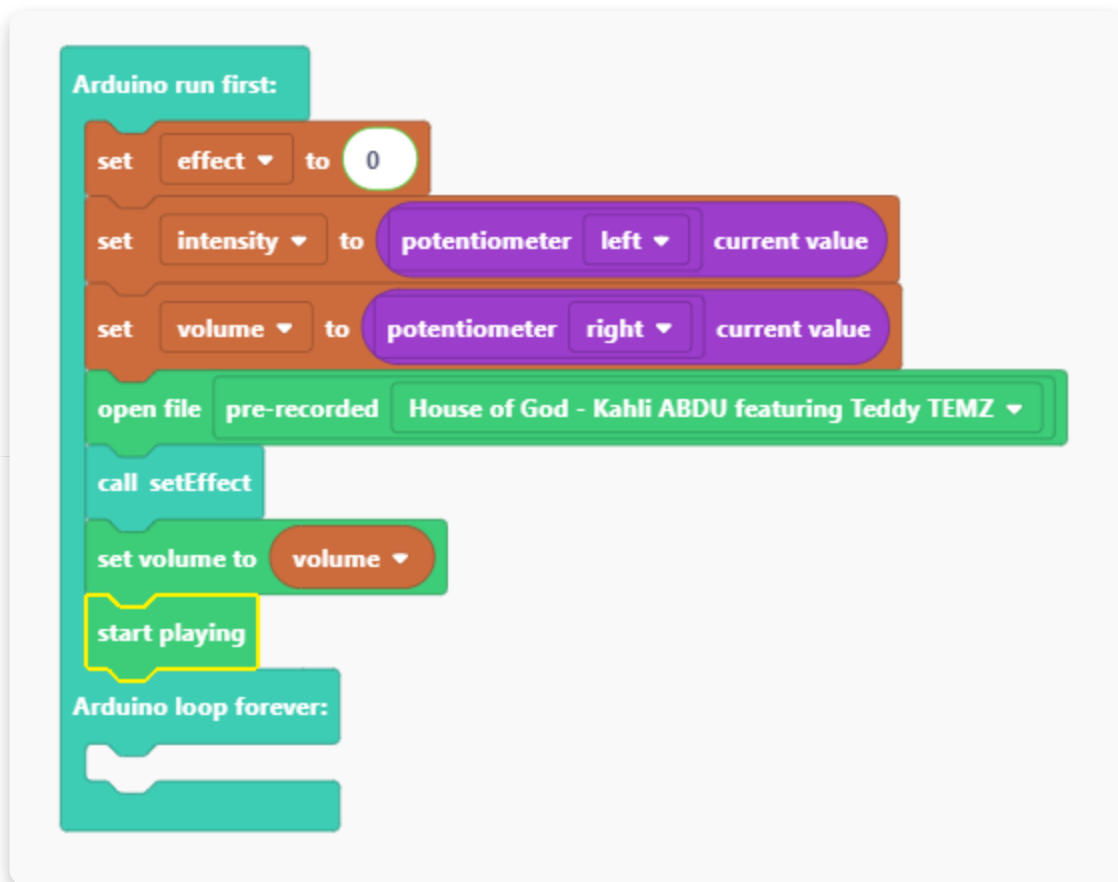
- Arduino run first:**
 - set effect to 0
 - set intensity to potentiometer left current value
 - set volume to potentiometer right current value
 - open file pre-recorded House of God - Kahli ABDU featuring Teddy TEMZ
 - call setEffect
 - set volume to 150 (highlighted with a yellow box)
- Arduino loop forever:**
 - (Empty loop body)

Genau wie bei der „intensity“-Variable müssen wir nun den numerischen Wert gegen die „volume“-Variable austauschen:



Öffne wieder die Kategorie „DJ“ und suche den Block „start“ playing“.

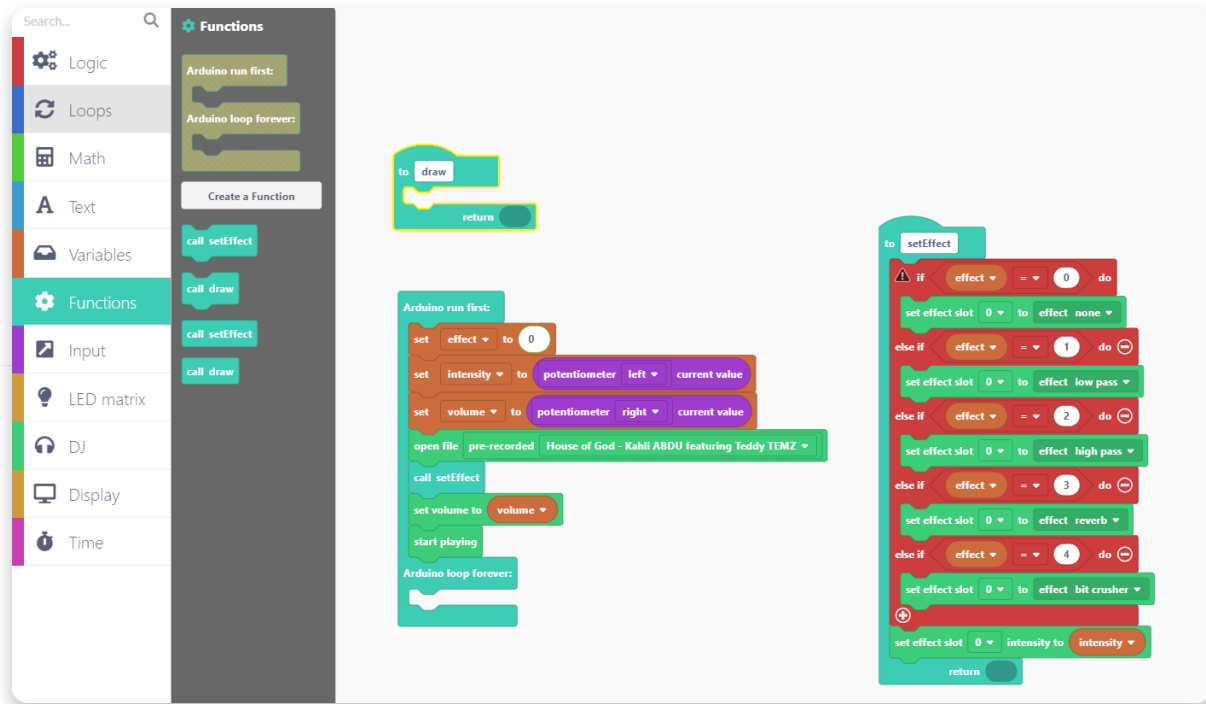
Lass ihn bitte hier fallen:



Jetzt müssen wir noch eine weitere Funktion erstellen.

Lass sie uns „draw“ (Zeichnen) nennen weil Sie dazu dienen wird, die Bezeichnung des Effekts und die Intensität auf dem LED-Display darzustellen.

Nach dem Du sie erstellt hast, erscheint sie in Deinem Sketch.



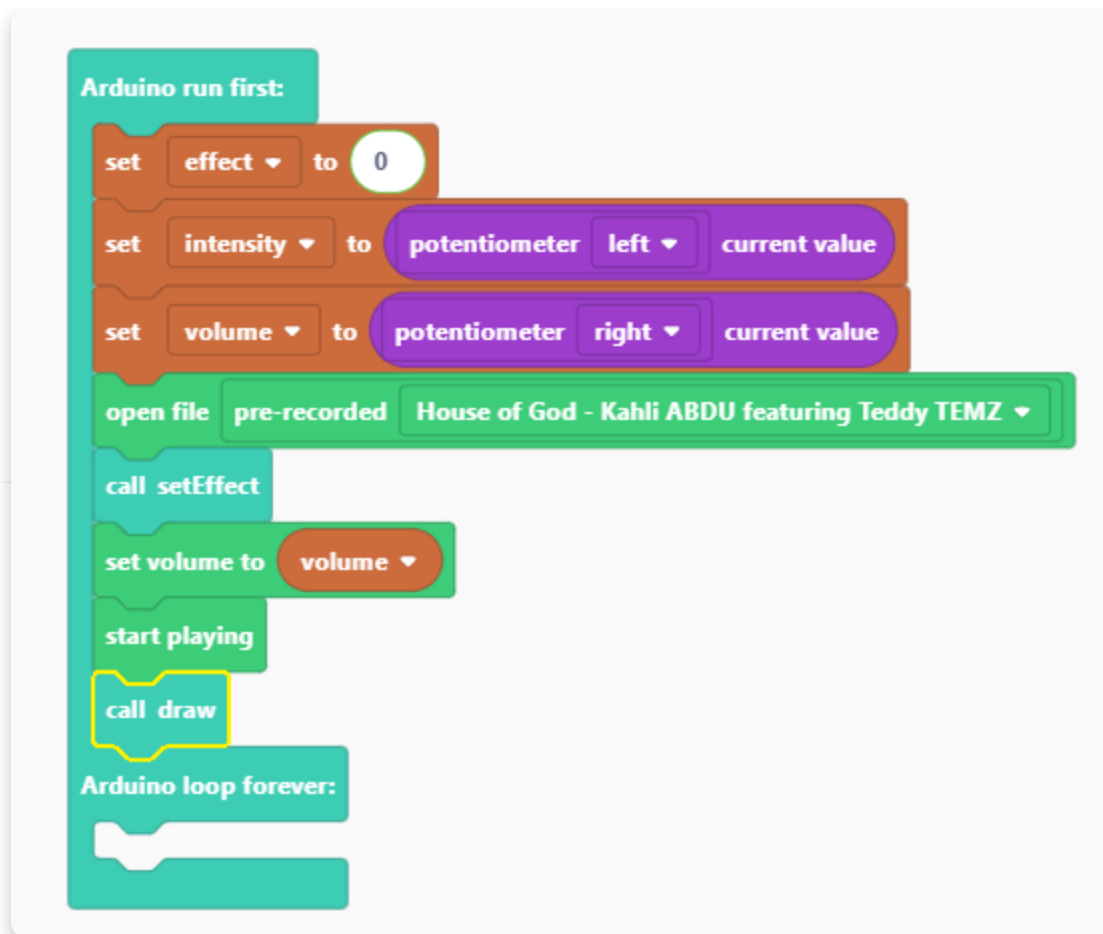
The screenshot shows the CircuitMuss IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions (highlighted), Input, LED matrix, DJ, Display, and Time. The 'Functions' panel is open, showing a 'draw' function block. The main workspace contains a sketch with the following code:

```
Arduino run first:  
  set effect to 0  
  set intensity to potentiometer left current value  
  set volume to potentiometer right current value  
  open file pre-recorded House of God - Kahli ABDU featuring Teddy TEMZ  
  call setEffect  
  set volume to volume  
  start playing  
Arduino loop forever:  
  to draw  
  return
```

The 'draw' function is defined as follows:

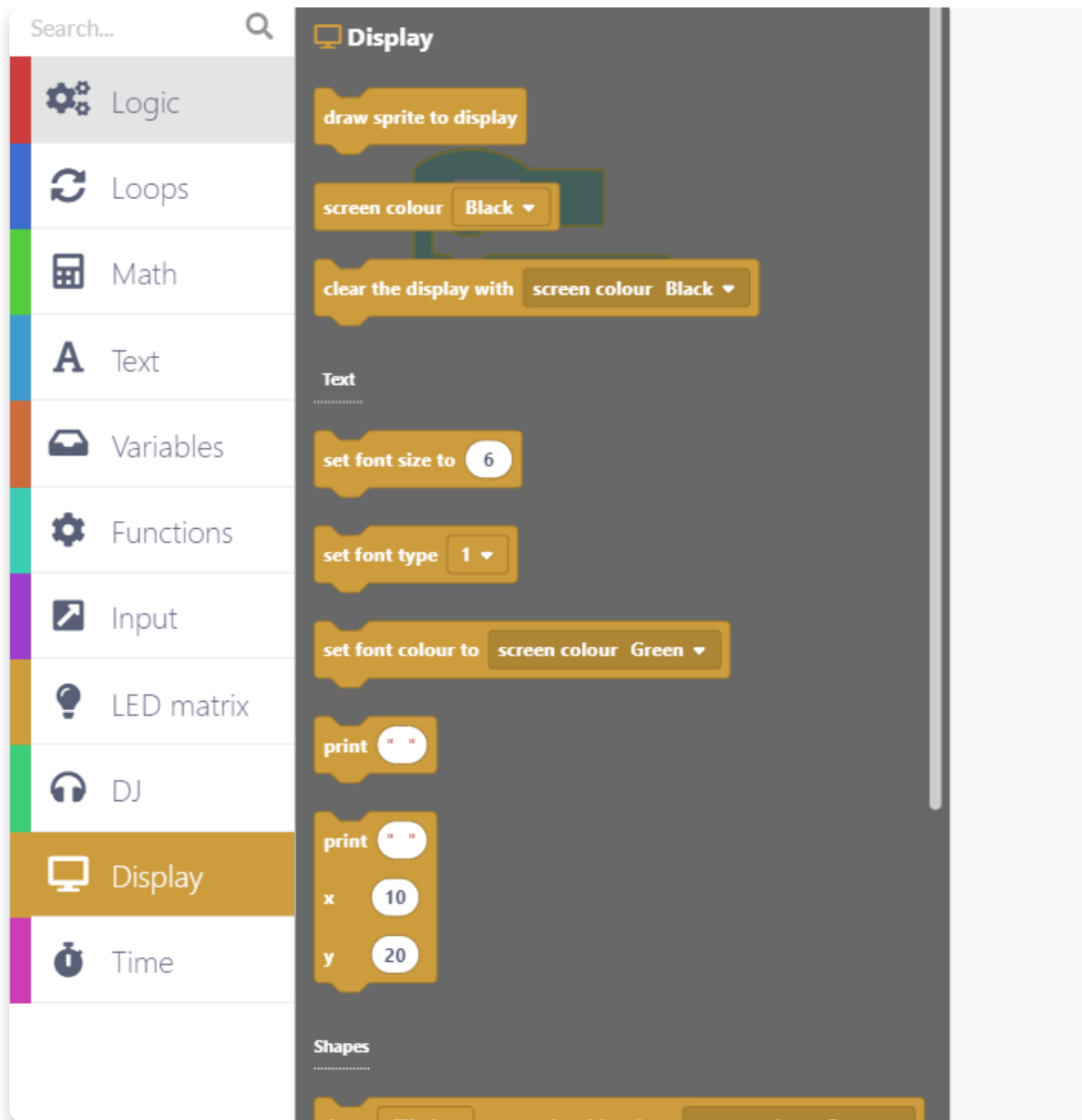
```
to setEffect  
  if effect == 0 do  
    set effect slot 0 to effect none  
  else if effect == 1 do  
    set effect slot 0 to effect low pass  
  else if effect == 2 do  
    set effect slot 0 to effect high pass  
  else if effect == 3 do  
    set effect slot 0 to effect reverb  
  else if effect == 4 do  
    set effect slot 0 to effect bit crusher  
  set effect slot 0 intensity to intensity  
  return
```

Vergiss bitte nicht, den „call draw“ Block hier fallen zu lassen:

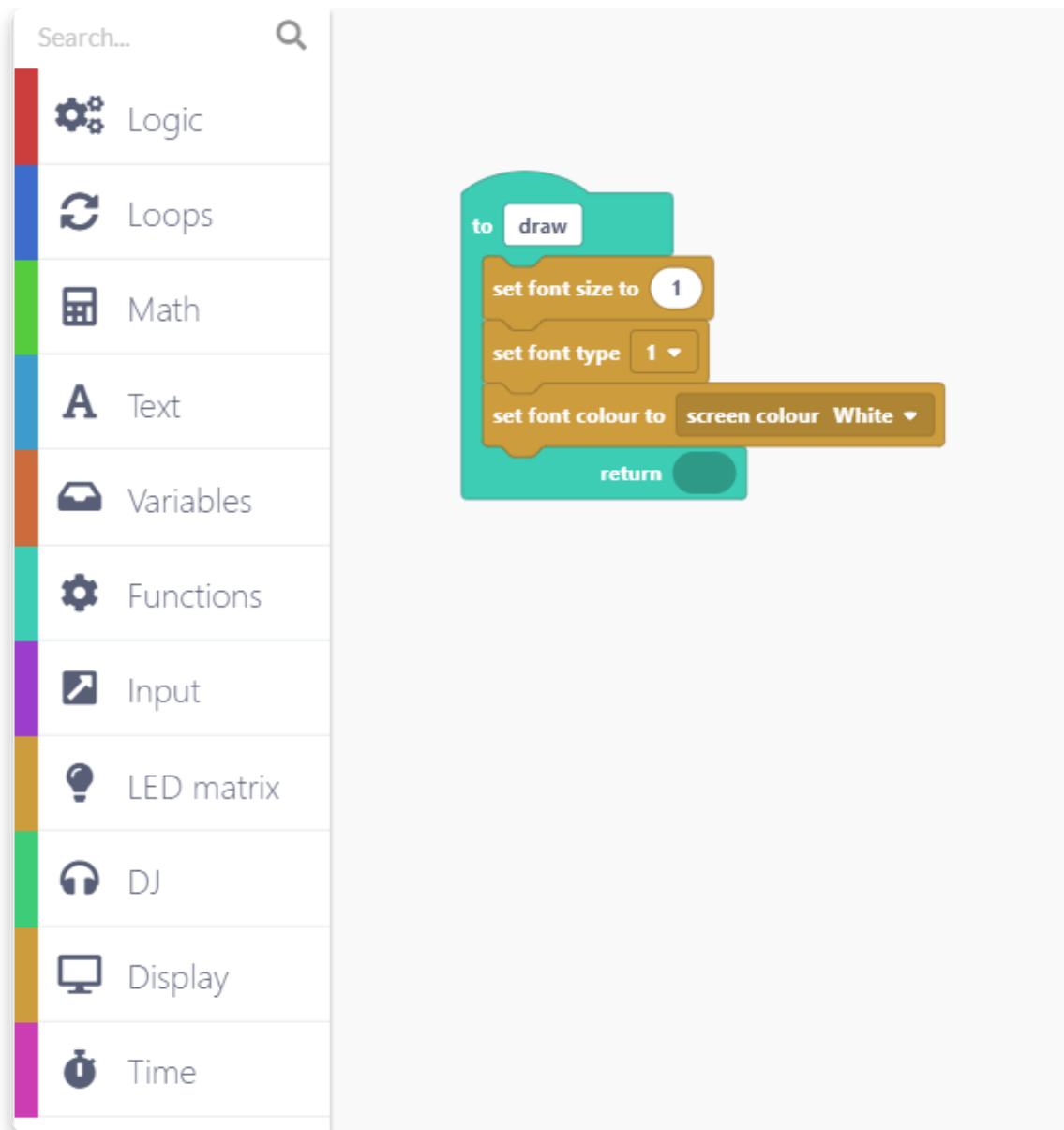


Jetzt bearbeiten wir die „draw“-Funktion.

Die Schriftgröße, Schriftart und Farbe können wir mit Blöcken aus der Kategorie „Display“ festlegen.



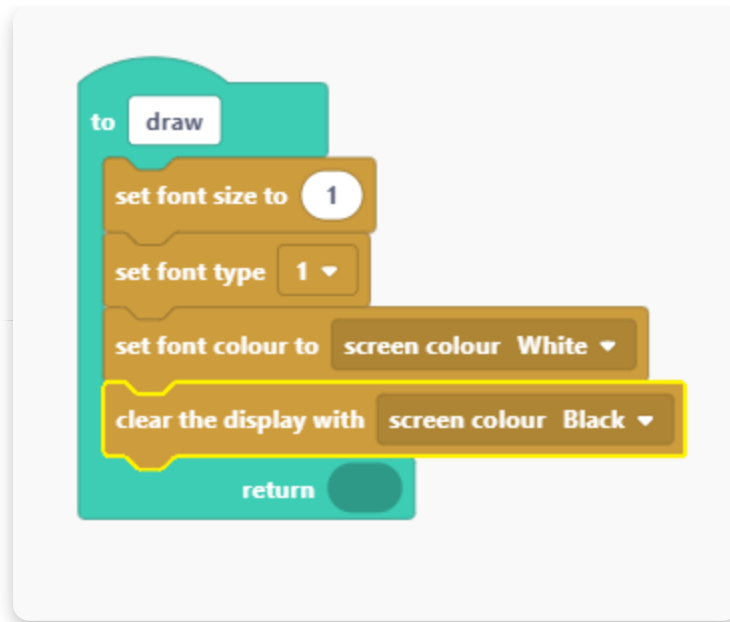
Wähle bitte die Einstellungen für den Bildschirmtext wie auf dem nächsten Bild:



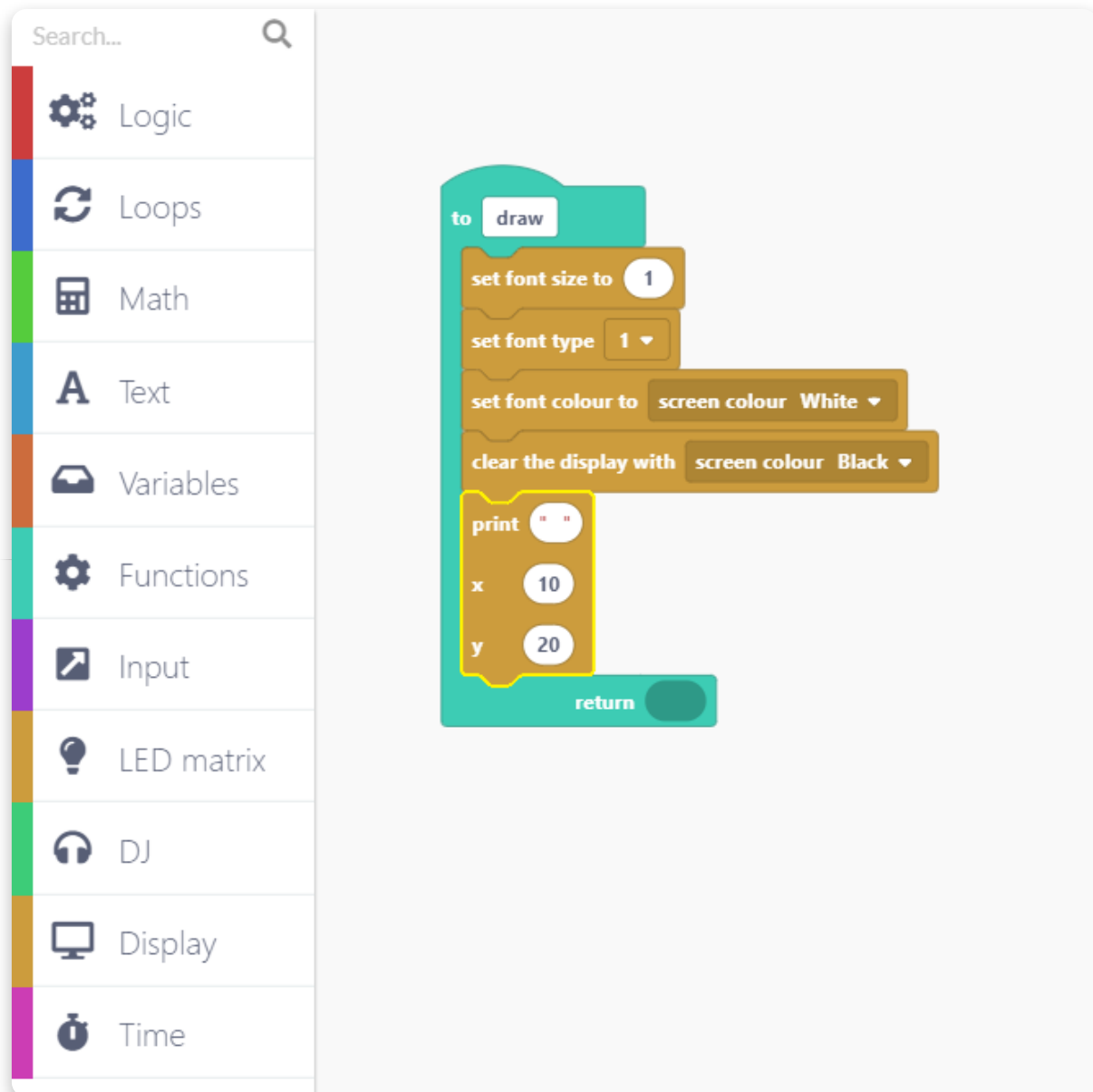
Da wir für den Text die Farbe Weiß ausgesucht haben sollte der Bildschirmhintergrund am Besten schwarz werden.

Suche in der Kategorie „Display“ den Block "clear the display with screen color Black".

Nimm den Block und lasse ihn bitte hier fallen:

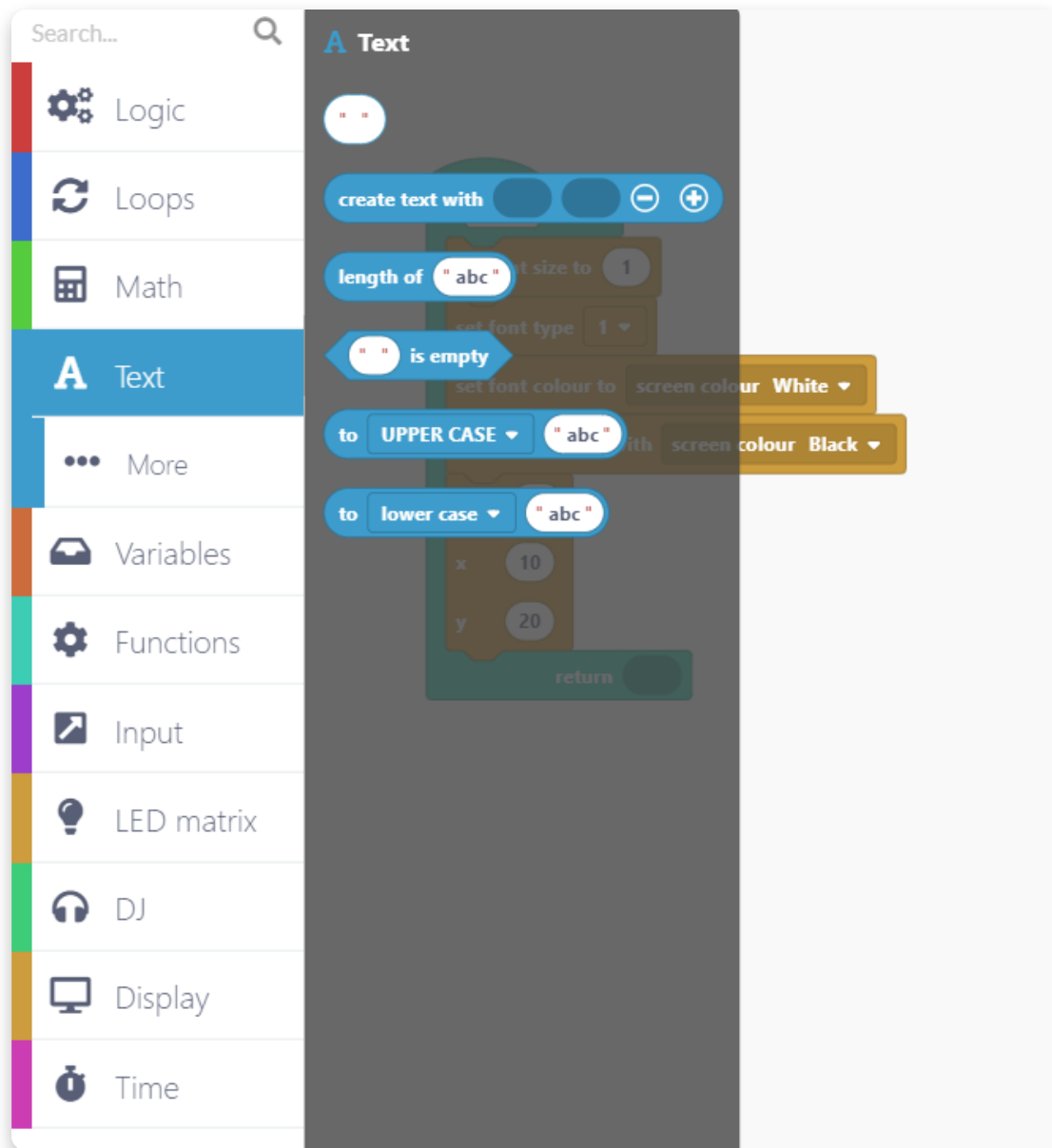


Nimm Dir aus der Kategorie „Display“ nun den „print“-Block:

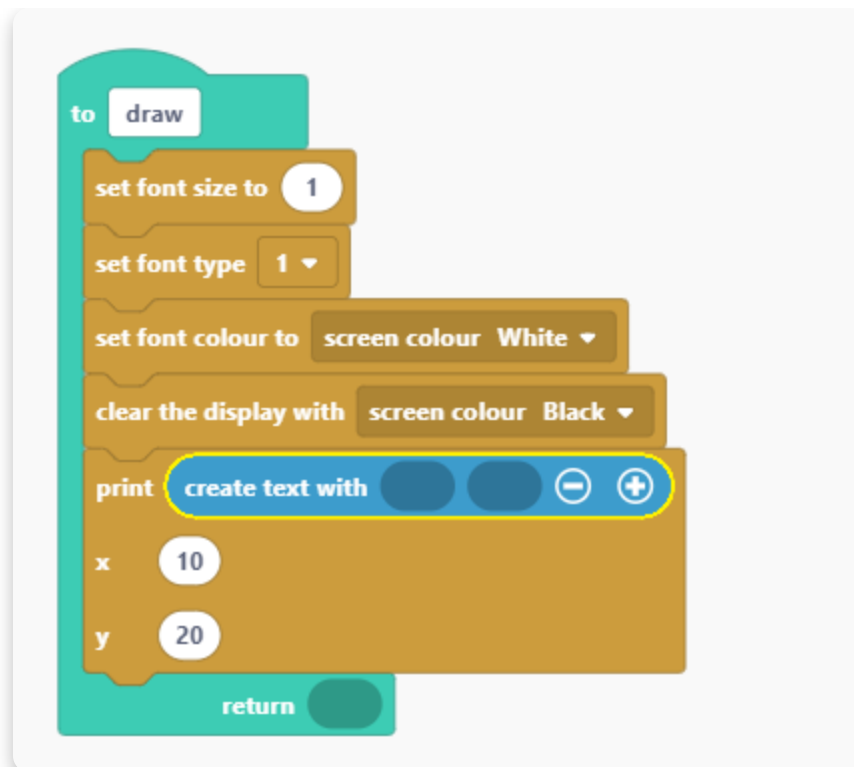


Den verändern wir nun so, dass er das Wort „Volume“ (also: Lautstärke) und auch den gerade eingestellten Wert der Lautstärke auf das LCD-Display schreibt.

Suche dazu aus der Kategorie „Text“ den Block "create text with".

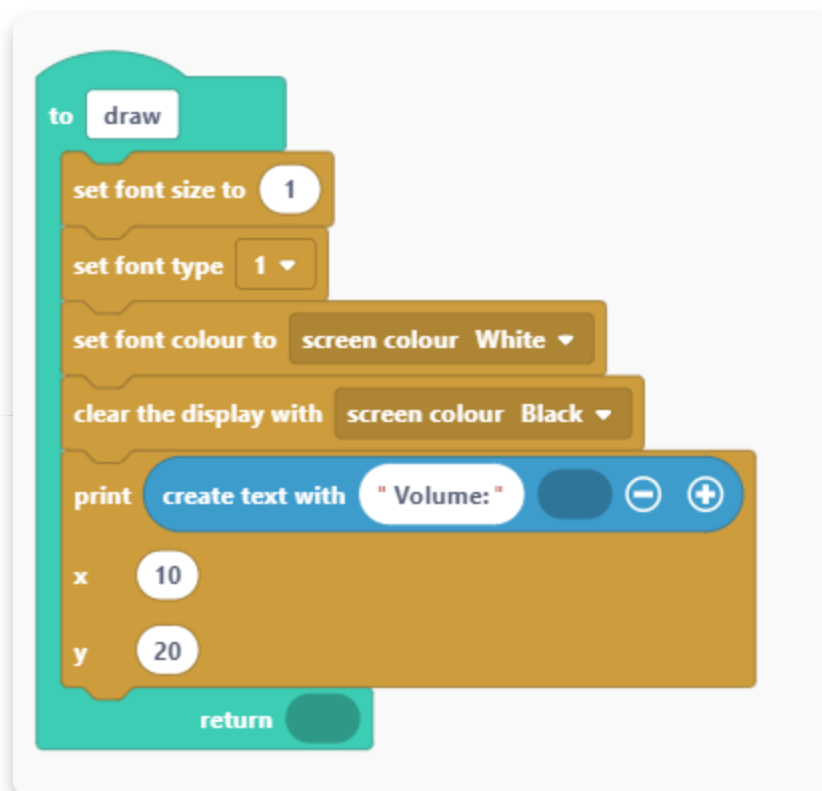


Nimm den Block und lasse ihn bitte hier fallen:



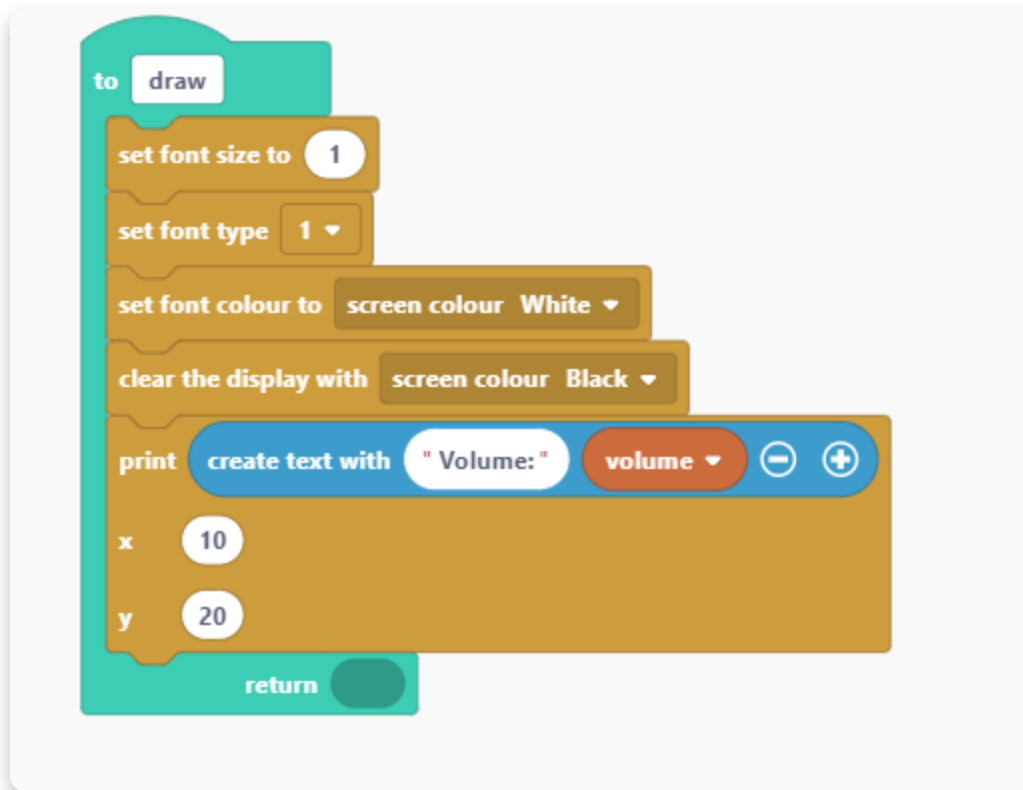
Öffne nochmal die Kategorie „Text“ und nimm den ersten, leeren Block (mit den beiden „“) und lass ihn in den zuvor verschobenen Block fallen.

Schreibe dann das Wort „Volume:“ hinein. Das sollte dann so aussehen:



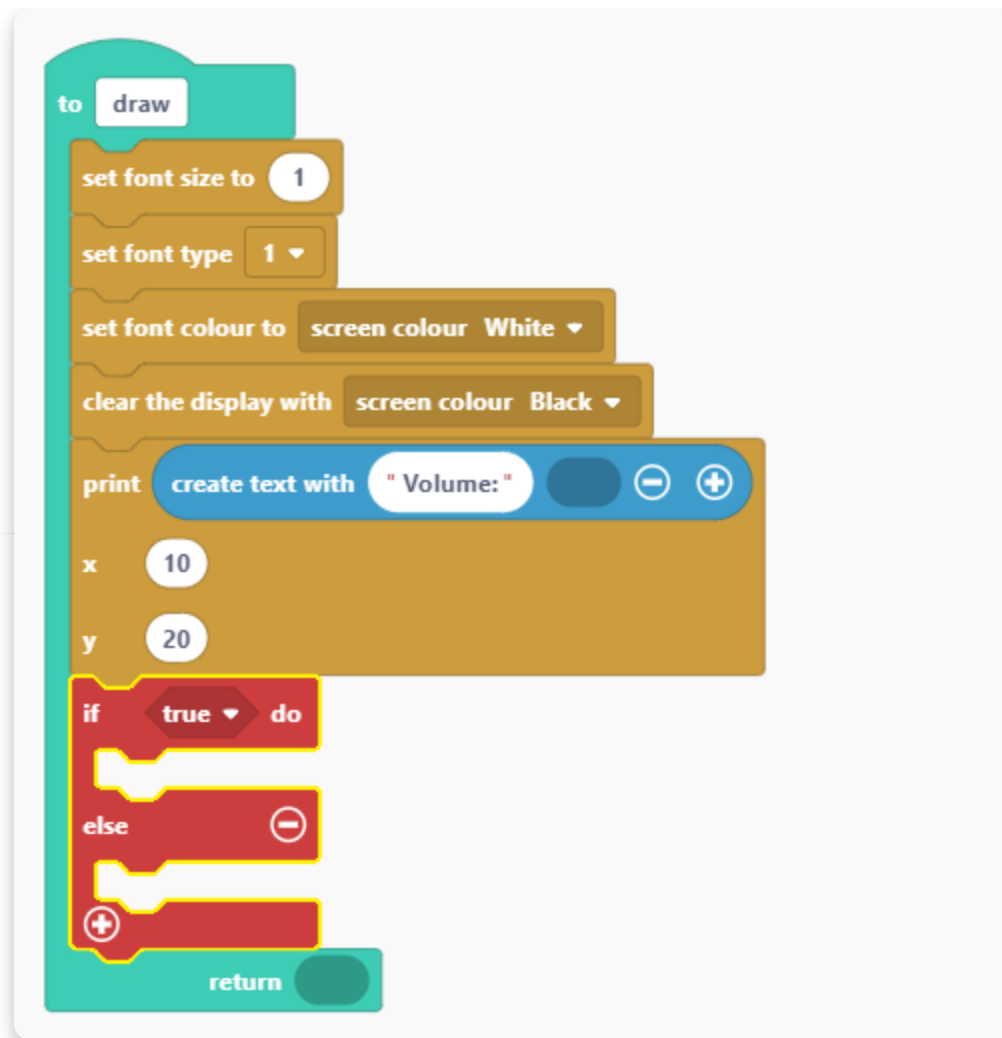
Ziehe nun die „Volume“ Variable in das Feld rechts neben den „Volume:“ Text. Damit wird später der Lautstärkewert auf dem Display angezeigt werden.

Die Koordinaten x und y geben an, an welcher Stelle auf dem Display der Text angezeigt werden wird. Trage hier bitte für x den Wert 10 und für y den Wert 20 ein.

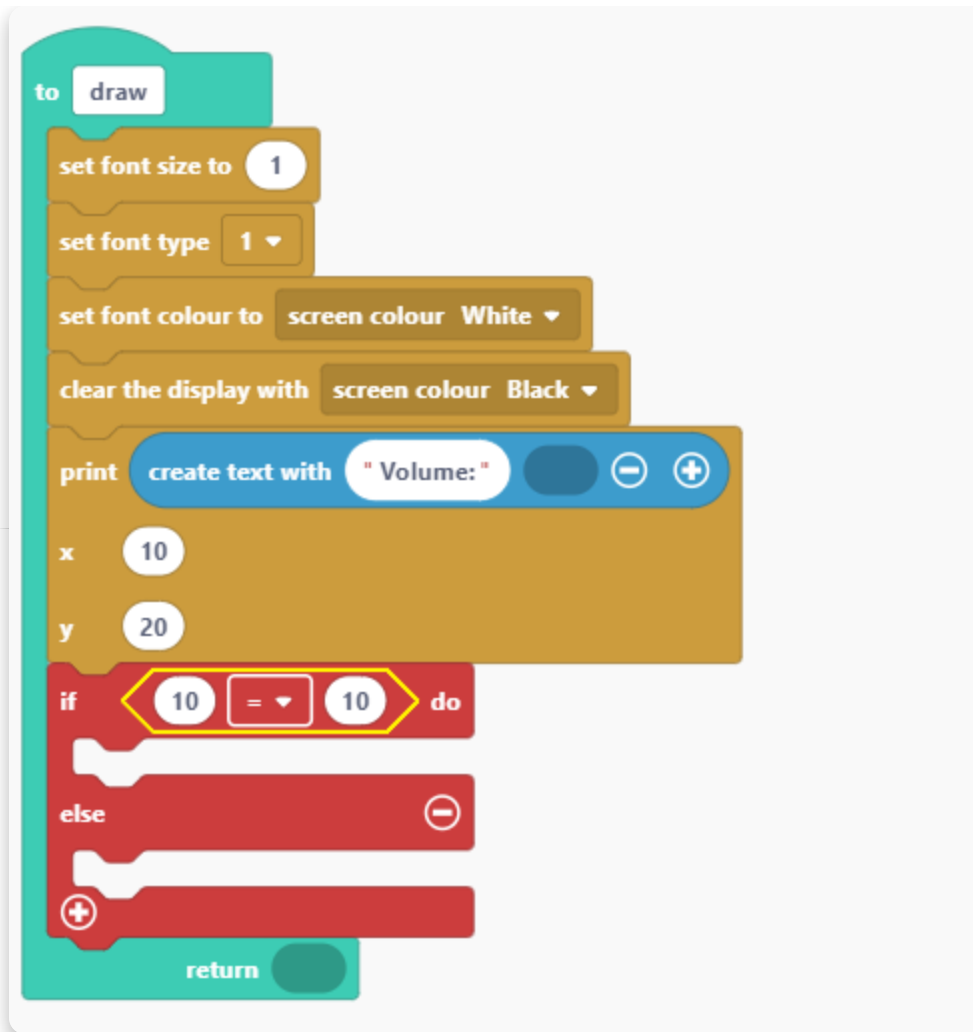


Um die Bezeichnung des Effekts anzuzeigen müssen wir eine „if-else“ Bedingung verwenden. Das Vorgehen ist vergleichbar mit dem bei den Effekten.

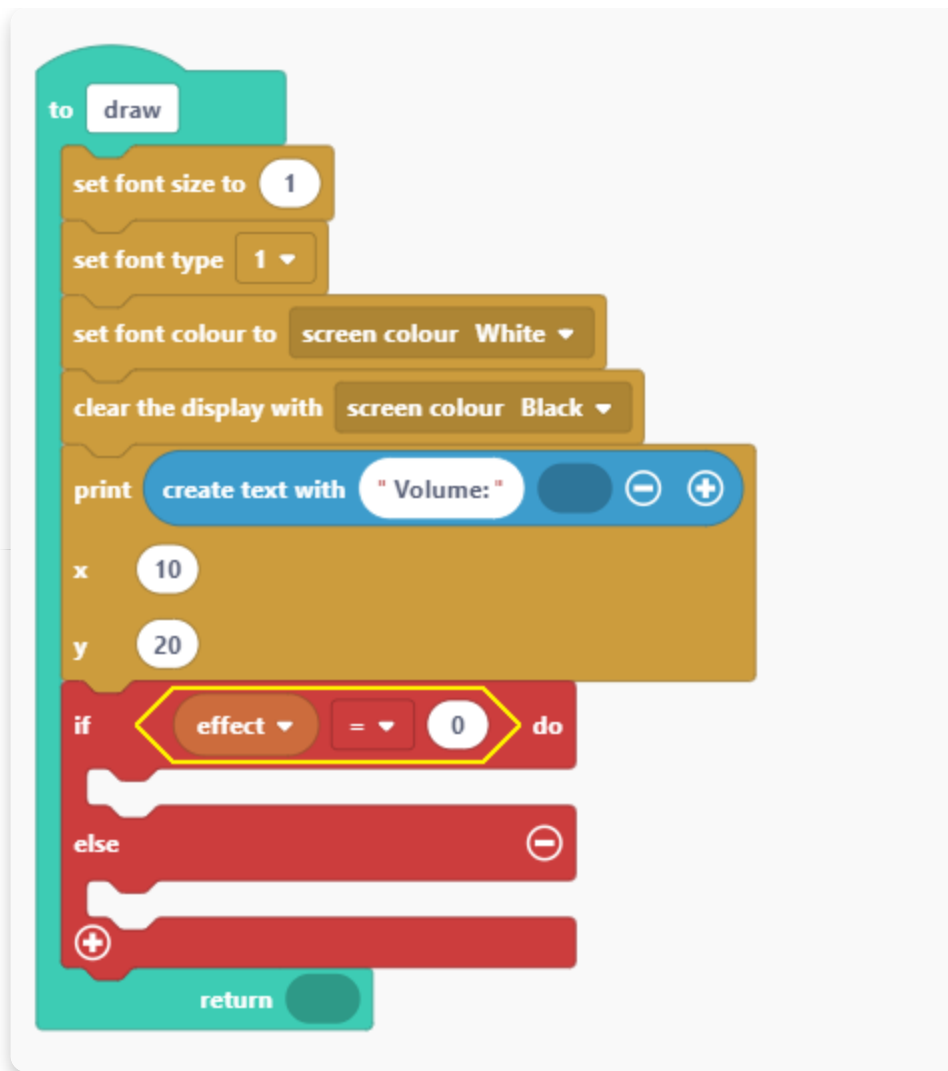
Bitte lass den „if-else“ Block hier fallen:



Anstelle der „true“-Bedingungen wählen wir aus dem Aufklappmenü die „ist gleich“ Bedingungen. Diese wird durch das Symbol „=" dargestellt.



Anstelle des ersten Zahlenwerts lassen wir hier die Variable „effekt“ hineinfallen. Ändere bitte dann den zweiten Wert (der weiter rechts steht) auf 0.



Hat der Effekt den Wert 0, so wird dann auf dem Display angezeigt, dass kein Effekt an ist.

Diese Bedingung ergänzen wir als nächstes.

Wir benötigen dazu eine neue Variable, die wir „effectname“ nennen. Sie wird später verwendet um den Namen des Effekts anzuzeigen.

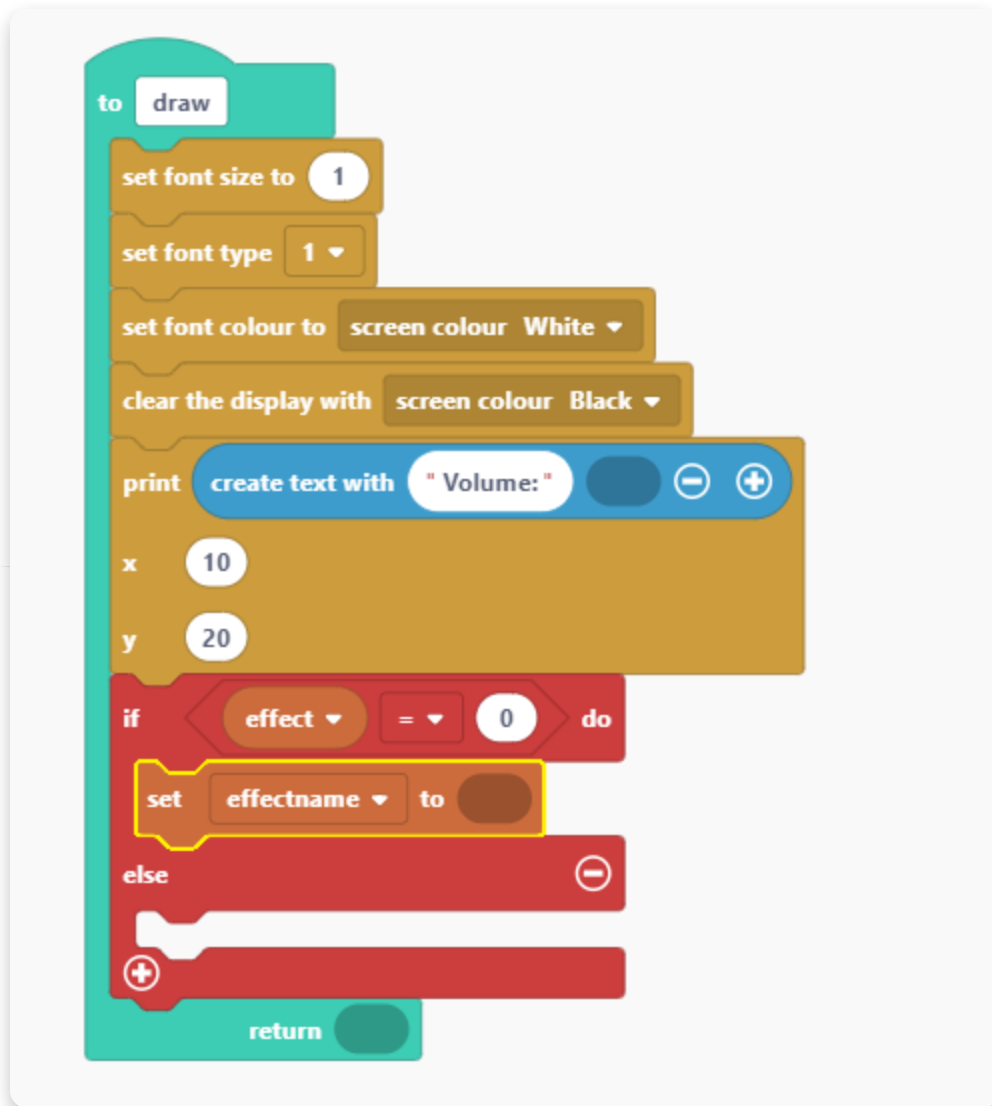
Erstelle bitte nun die neue Variable „effectname“:

The screenshot shows the CircuitMess IDE interface. On the left is a vertical menu with categories: Logic, Loops, Math, Text, Variables (highlighted in orange), Functions, Input, LED matrix, DJ, Display, and Time. The main workspace is titled 'Variables' and contains a script:

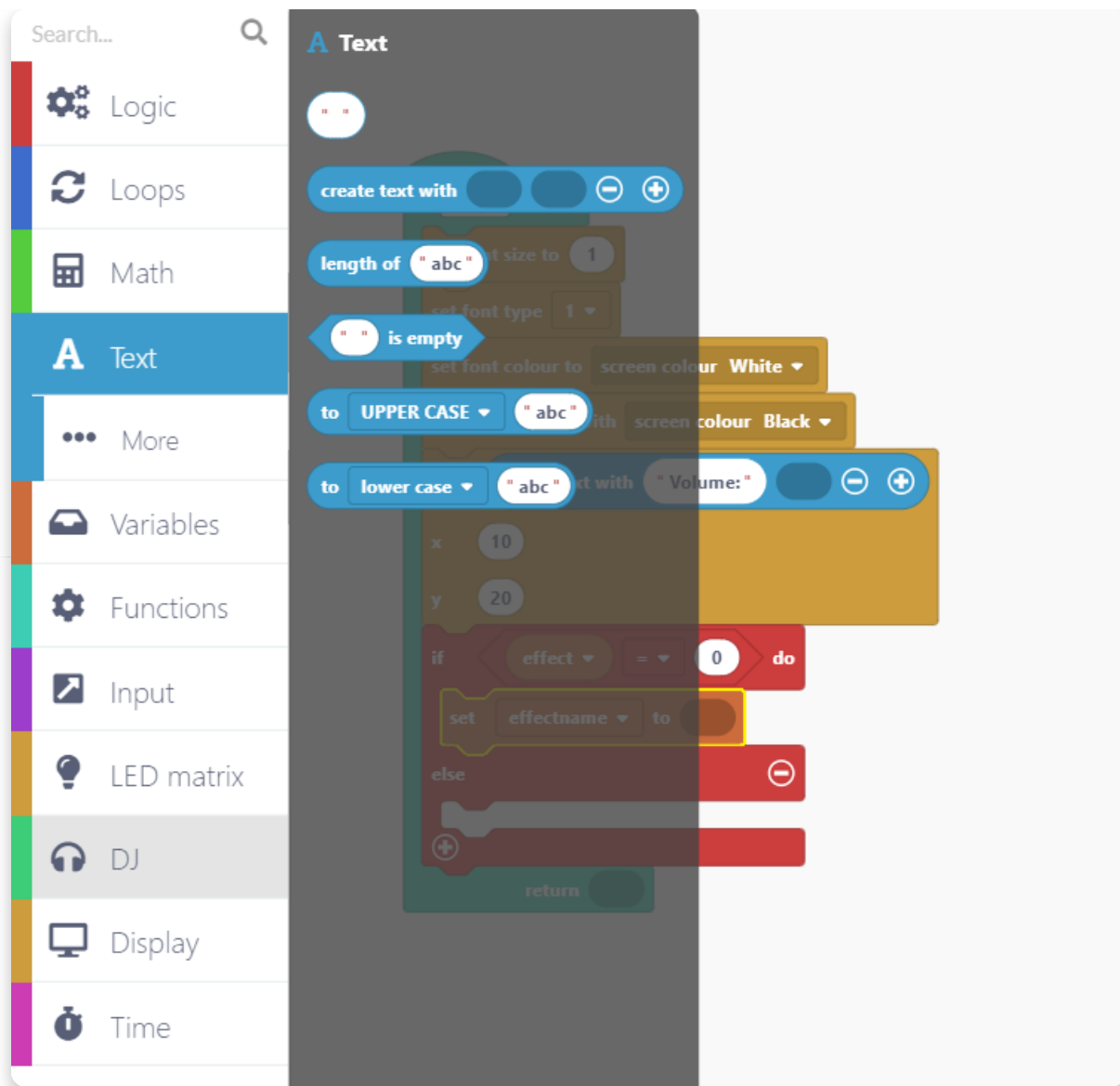
- Buttons for 'effect', 'intensity', and 'volume' are visible.
- Script blocks include: 'set volume to', 'change volume by 1', 'set font colour to screen colour White', 'clear the display with screen colour Black', 'print create text with "Volume:"', and an 'if effect = 0 do' block with '+' and '-' buttons.

The screenshot shows a dialog box titled 'New variable name:' with a text input field containing 'effectname'. Below the input field are two buttons: 'Close' and 'Save'. The background shows a script with 'set intensity to potentiometer left current value' and 'set volume to potentiometer right current value' blocks.

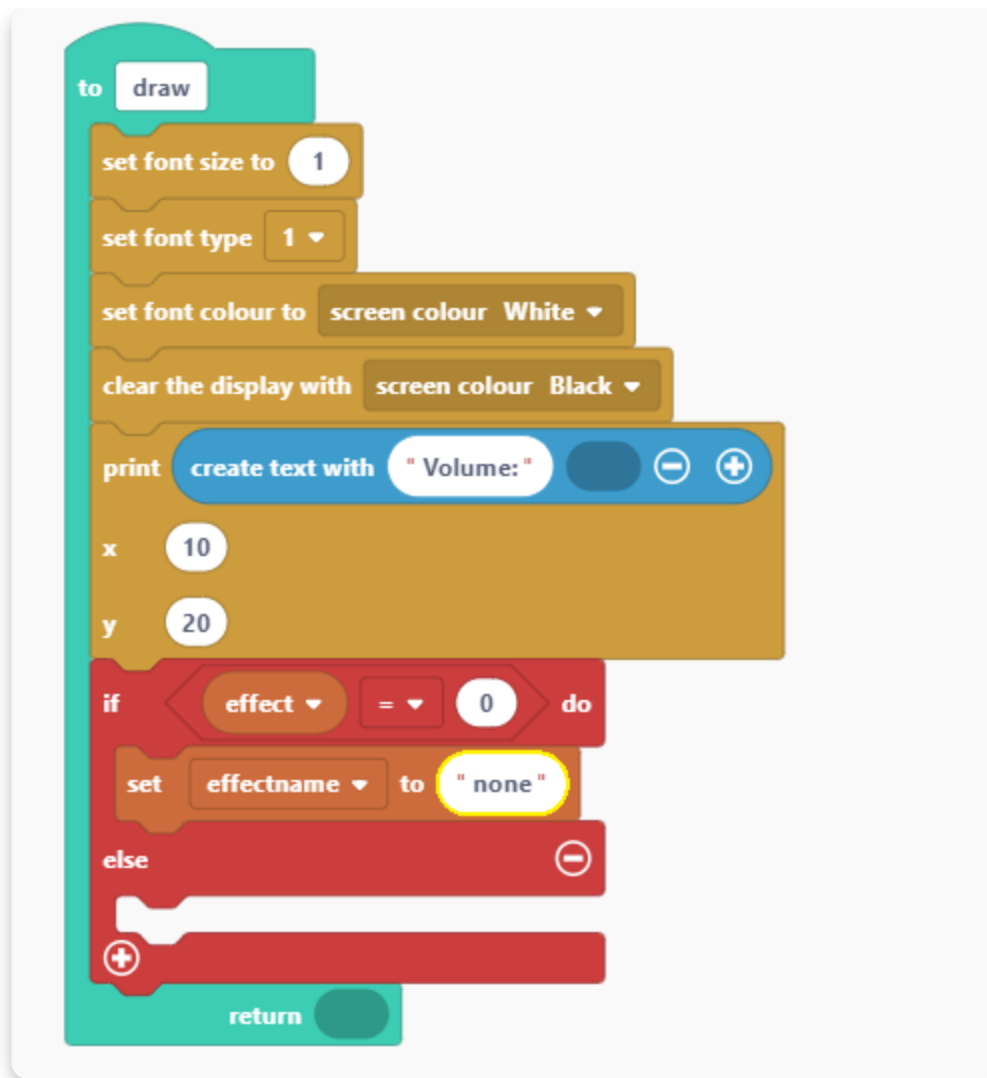
Lass den Block "set effectname to" bitte hier fallen:



Nimm nun aus der Kategorie einen Block für Text (die weiße Blase mit den beiden „“).



Schreibe bitte den Text „none“ hinein. Wenn der Effekt nun den Wert 0 besitzt, wird auf dem Display „none“ (also: kein Effekt) angezeigt.



Diesen Schritt wiederholen wir nun für alle Effekte. Erweitere den Block indem Du auf das kleine Pluszeichen am unteren Rand klickst.

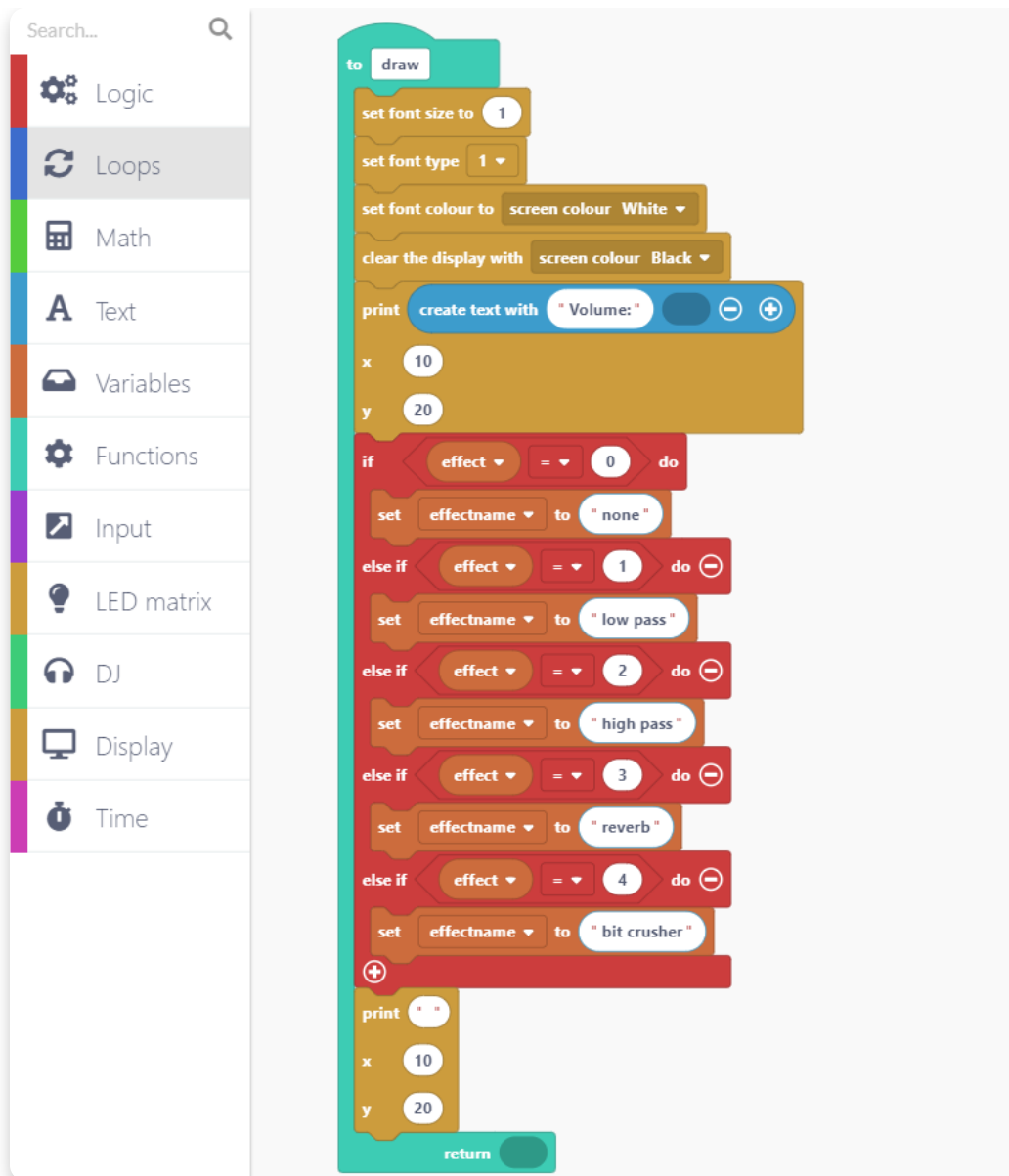
Wenn Du fertig bist, klicke bitte auf das kleine Minuszeichen.

The screenshot shows the Circuit Blocks coding environment. On the left is a sidebar with a search bar and categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace contains a 'to draw' block with the following code:

```
to draw
  set font size to 1
  set font type 1
  set font colour to screen colour White
  clear the display with screen colour Black
  print create text with "Volume:"
  x 10
  y 20
  if effect = 0 do
    set effectname to "none"
  else if effect = 1 do
    set effectname to "low pass"
  else if effect = 2 do
    set effectname to "high pass"
  else if effect = 3 do
    set effectname to "reverb"
  else if effect = 4 do
    set effectname to "bit crusher"
  return
```

Nachdem Du nun alle Bedingungen erstellt hast, müssen wir nun den Block für die Anzeige der Effekt-Bezeichnung anlegen.

Dazu gehst Du bitte in die Kategorie „Display“ und nimmst diesen Block:

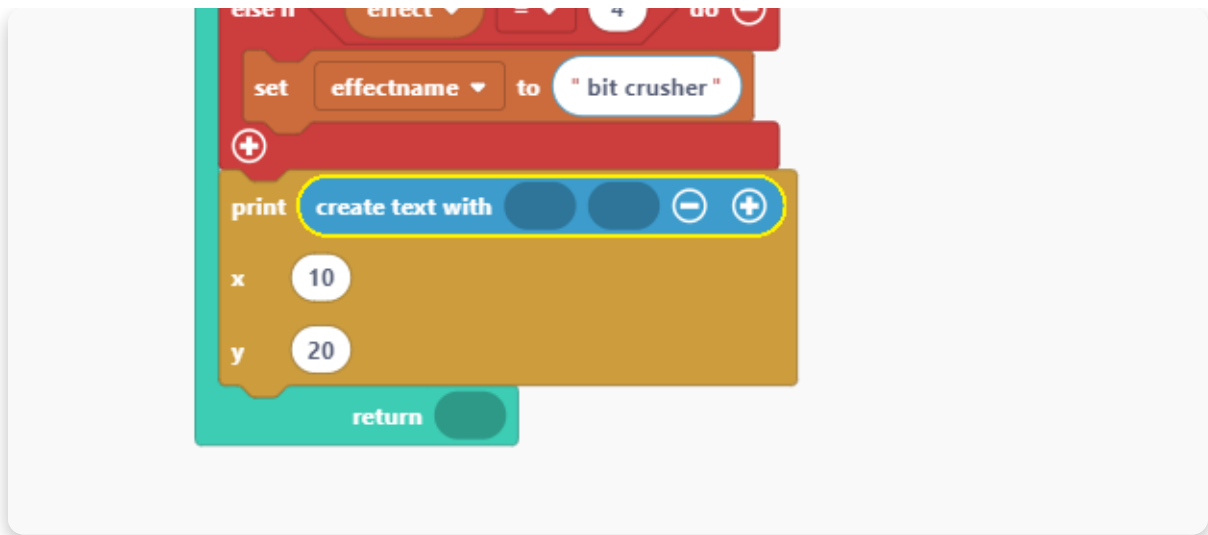


The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The 'Text' category is selected. The main workspace displays a 'to draw' function block with the following code:

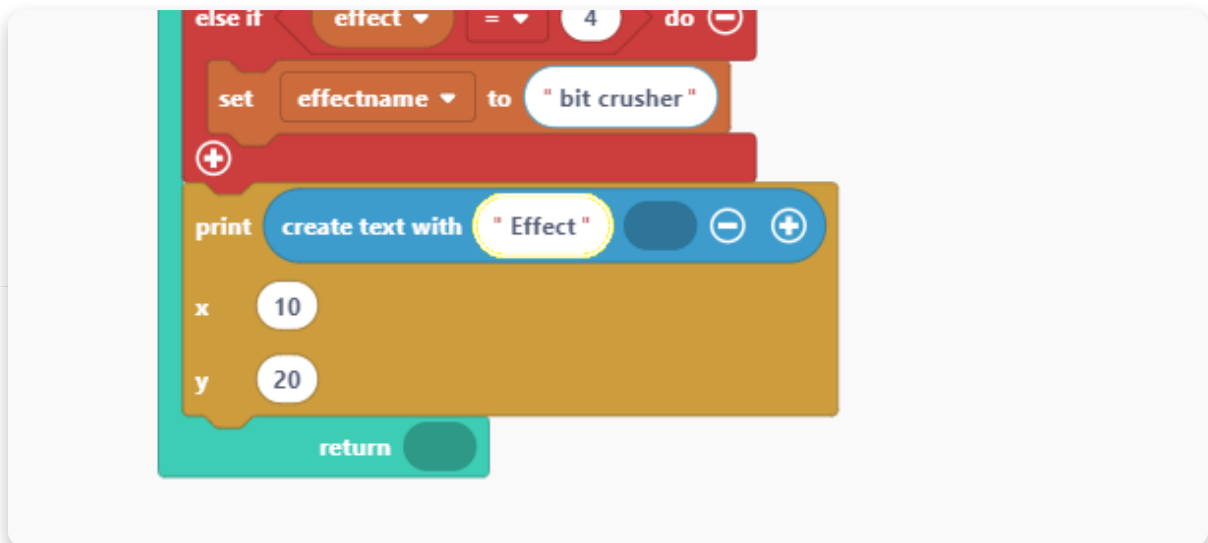
```
to draw
  set font size to 1
  set font type 1
  set font colour to screen colour White
  clear the display with screen colour Black
  print create text with " Volume: "
  x 10
  y 20
  if effect = 0 do
    set effectname to "none"
  else if effect = 1 do
    set effectname to "low pass"
  else if effect = 2 do
    set effectname to "high pass"
  else if effect = 3 do
    set effectname to "reverb"
  else if effect = 4 do
    set effectname to "bit crusher"
  print " "
  x 10
  y 20
return
```

Um die Effekt-Bezeichnung einzufügen, gehe bitte in die Kategorie „Text“ und wähle den Block "create text with" aus.

Lass den Block dann bitte hier fallen:

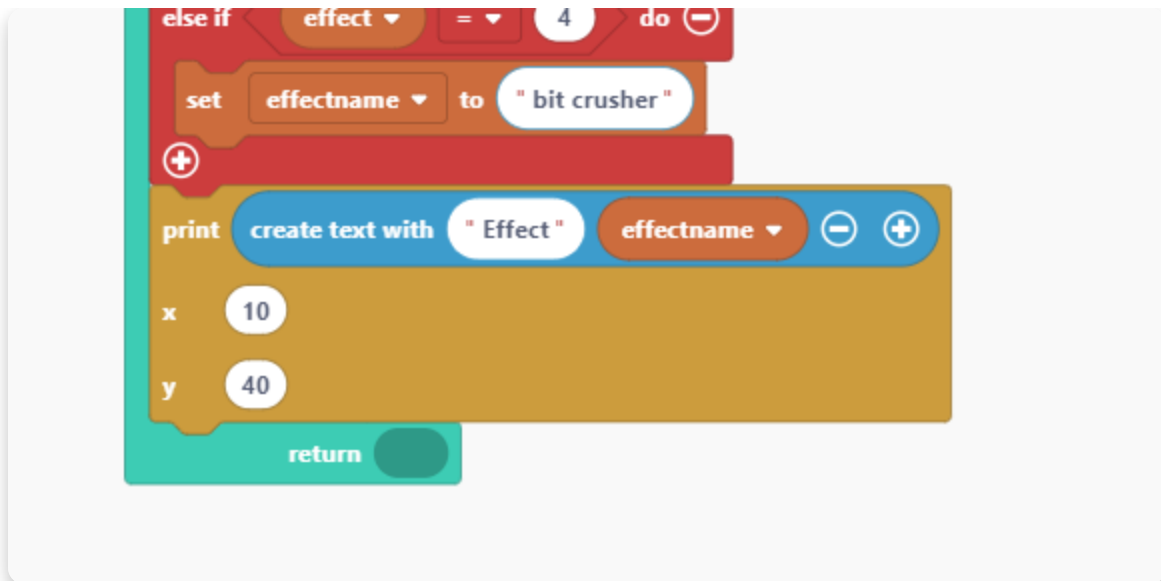


Füge nun aus der Kategorie „Text“ ein leeres Textfeld ein. Dort hinein schreibst Du dann bitte: „Effect:“.



Lasse nun die Variable „effectname“ in das Feld rechts vom Textfeld fallen.

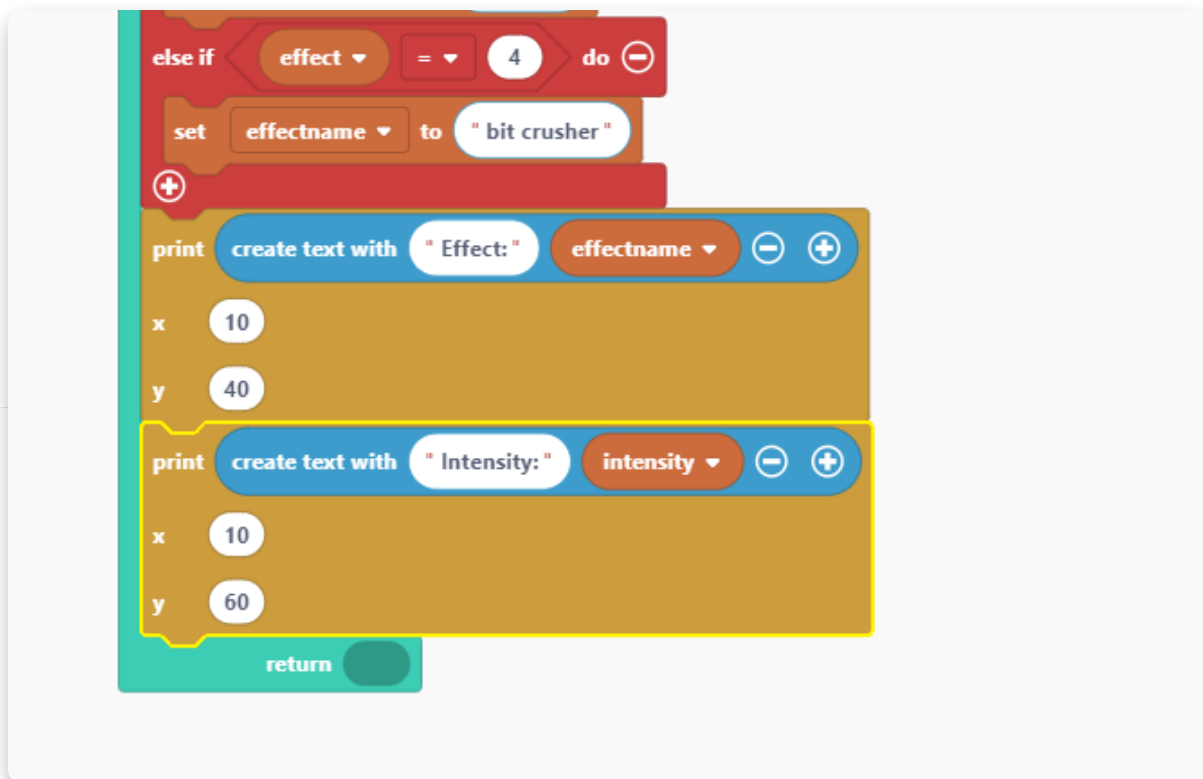
Und setze dann bitte die Werte der Koordinaten für x auf 10 und für y auf 40.



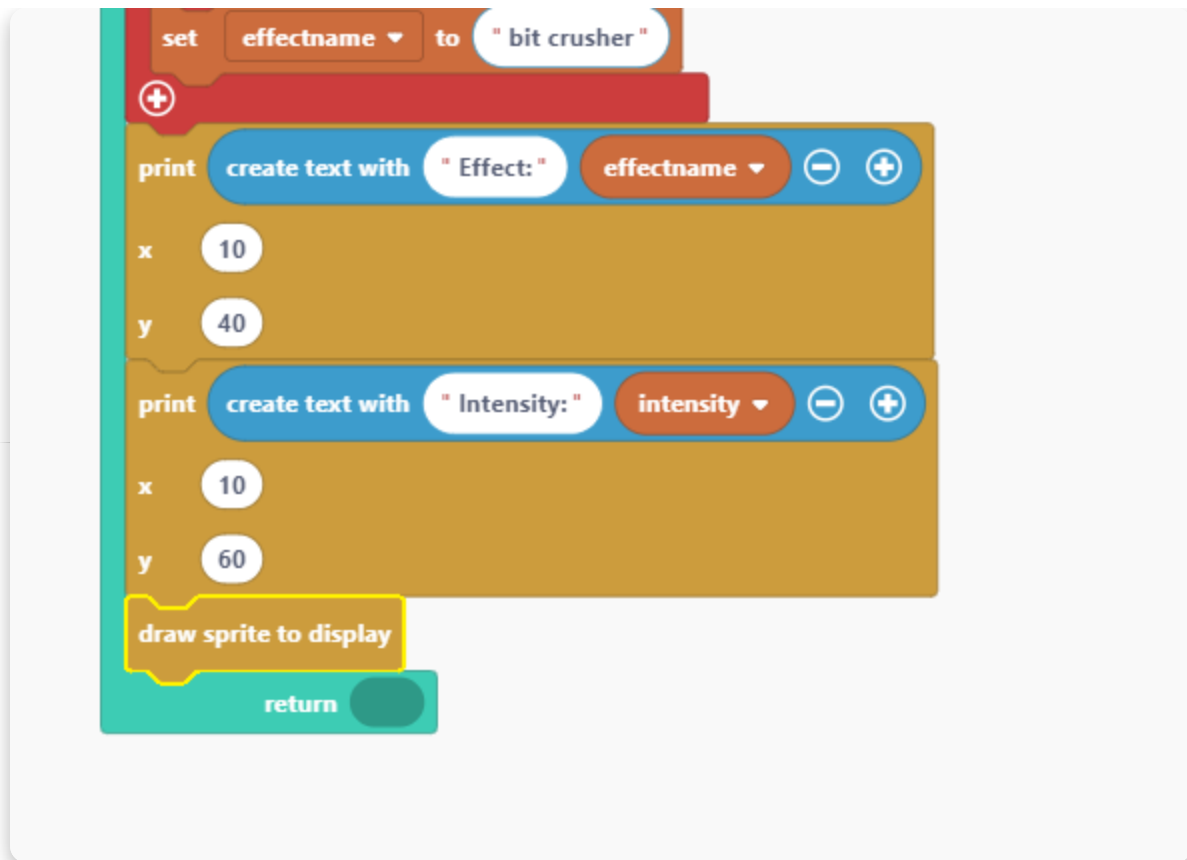
Prima. Mit dieser Funktion sollte später der Name des Effektes hinter dem Lautstärkewert auf dem Display Deines Jay-D angezeigt werden.

Zum Schluss muss nun noch die Anzeige des „intensity“ Wertes programmiert werden.

Nimm dazu einen weiteren „print“-Block und passe ihn, wie hier gezeigt, an:



Zum Anschluss der „draw“-Funktion nehmen wir nun noch aus der Kategorie „Display“ den Block "draw sprite to display".



Die fertige Funktion sollte bei Dir nun so aussehen:


```
to draw
  set font size to 1
  set font type to 1
  set font colour to screen colour White
  clear the display with screen colour Black
  print create text with "Volume:"
  x 10
  y 20
  if effect = 0 do
    set effectname to "none"
  else if effect = 1 do
    set effectname to "low pass"
  else if effect = 2 do
    set effectname to "high pass"
  else if effect = 3 do
    set effectname to "reverb"
  else if effect = 4 do
    set effectname to "bit crusher"
  print create text with "Effect:" effectname
  x 10
  y 40
  print create text with "Intensity:" intensity
  x 10
  y 60
  draw sprite to display
return
```

Und so sollte Dein ganzer Sketch nun aussehen:

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace contains two code blocks:

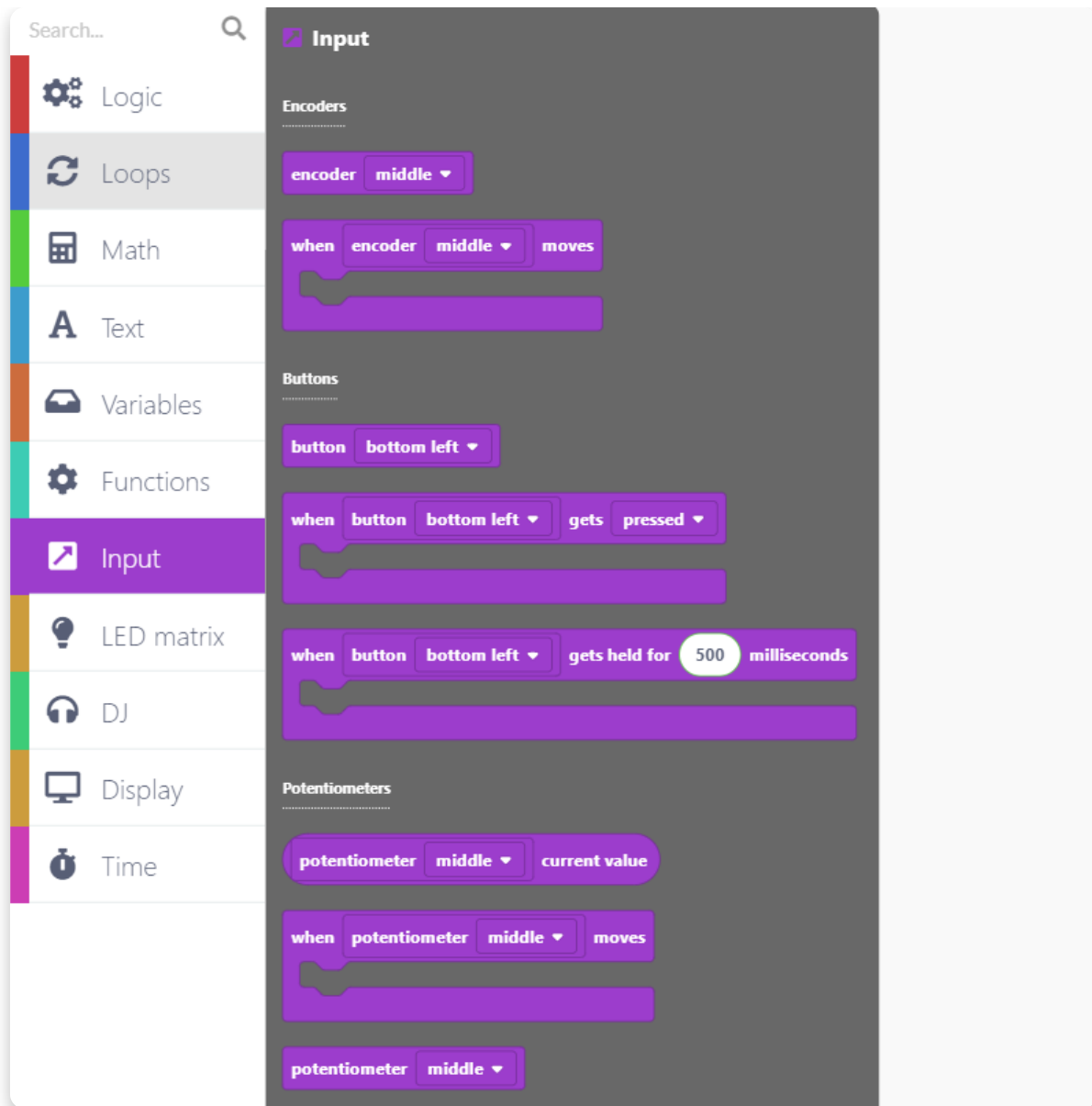
- Arduino run first:**
 - set effect to 0
 - set intensity to potentiometer left current value
 - set volume to potentiometer right current value
 - open file pre-recorded House of God - Kabli ABDU featuring Teddy TEMZ
 - call setEffect
 - set volume to volume
 - start playing
 - call draw
 - Arduino loop forever:
- to setEffect:**
 - if effect == 0 do
 - set effect slot 0 to effect none
 - else if effect == 1 do
 - set effect slot 0 to effect low pass
 - else if effect == 2 do
 - set effect slot 0 to effect high pass
 - else if effect == 3 do
 - set effect slot 0 to effect reverb
 - else if effect == 4 do
 - set effect slot 0 to effect bit crusher
 - set effect slot 0 intensity to intensity
 - return
- to draw:**
 - set font size to 1
 - set font type 1
 - set font colour to screen colour White
 - clear the display with screen colour Black
 - print create text with Volume
 - x 10
 - y 20
 - if effect == 0 do
 - set effectname to none
 - else if effect == 1 do
 - set effectname to low pass
 - else if effect == 2 do
 - set effectname to high pass
 - else if effect == 3 do
 - set effectname to reverb
 - else if effect == 4 do
 - set effectname to bit crusher
 - print create text with Effect: effectname
 - x 10
 - y 40
 - print create text with Intensity: intensity
 - x 10
 - y 60
 - draw sprite to display
 - return

Was nun noch gemacht werden muss ist die Programmierung des mittleren Drehreglers für die Anpassung von Lautstärke und Effekt-Intensität.

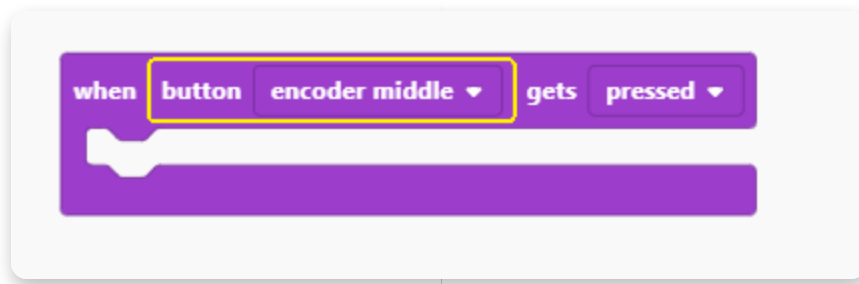
Lass uns mit den Effekten beginnen. Das soll später mit dem Druck auf den mittleren Drehregler passieren.

Also nehmen wir dazu aus der Kategorie „Input“ den Block "when button bottom left gets pressed".

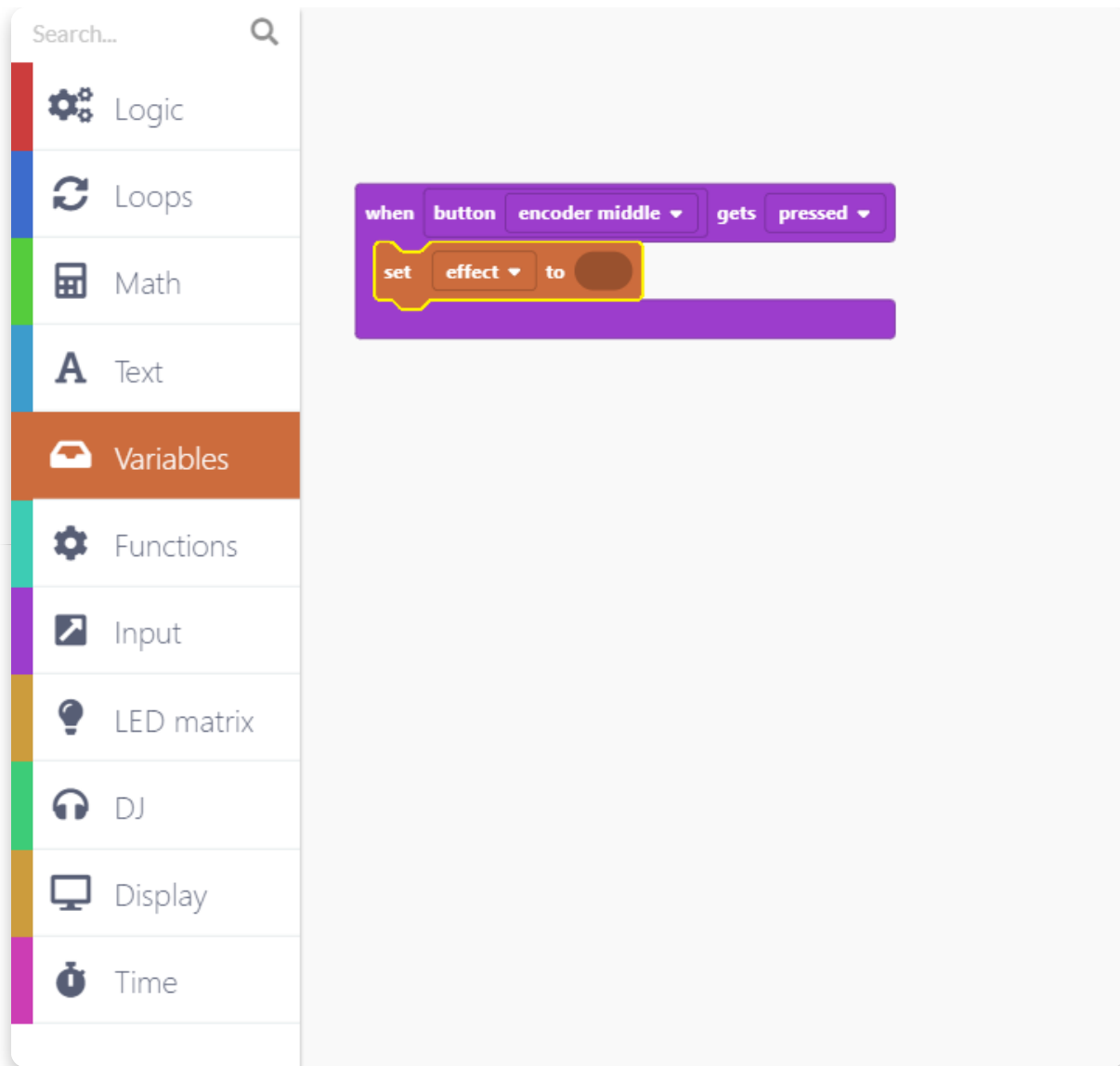
Zieh den Block in den Zeichenbereich und lass ihn fallen.



Wähle aus dem ersten Aufklappmenü nun den Punkt „encoder middle“ und im zweiten Menü den Punkt „pressed“ (also: gedrückt).



Um auszuwählen, was beim Drich auf den Drehregler passieren soll, holen wir uns diesen Block aus der Kategorie „Variables“:



Da wir ja insgesamt fünf mögliche Zustände für „effect“ haben (none, low pass, high pass, reverb, und bit crusher), nehmen wir uns aus der Kategorie „Math“ den Block "remainder". Dieser Block teilt die Anzahl der Tastendrücke durch 5 (für die 5 möglichen Zustände).

The screenshot shows the CircuitBlocks IDE interface. On the left is a sidebar with a search bar and a list of categories: Logic, Loops, Math, Text, Variables, Functions, Input, LED matrix, DJ, Display, and Time. The main workspace contains a single code block. The block starts with a 'when' trigger: 'when button encoder middle gets pressed'. This is followed by a 'set effect to remainder of 64 ÷ 10' block. The 'remainder of' block has two input fields: the first contains '64' and the second contains '10'.

Füge dazu den Block „division“ in das erste Feld des „remainder“-Blocks ein:

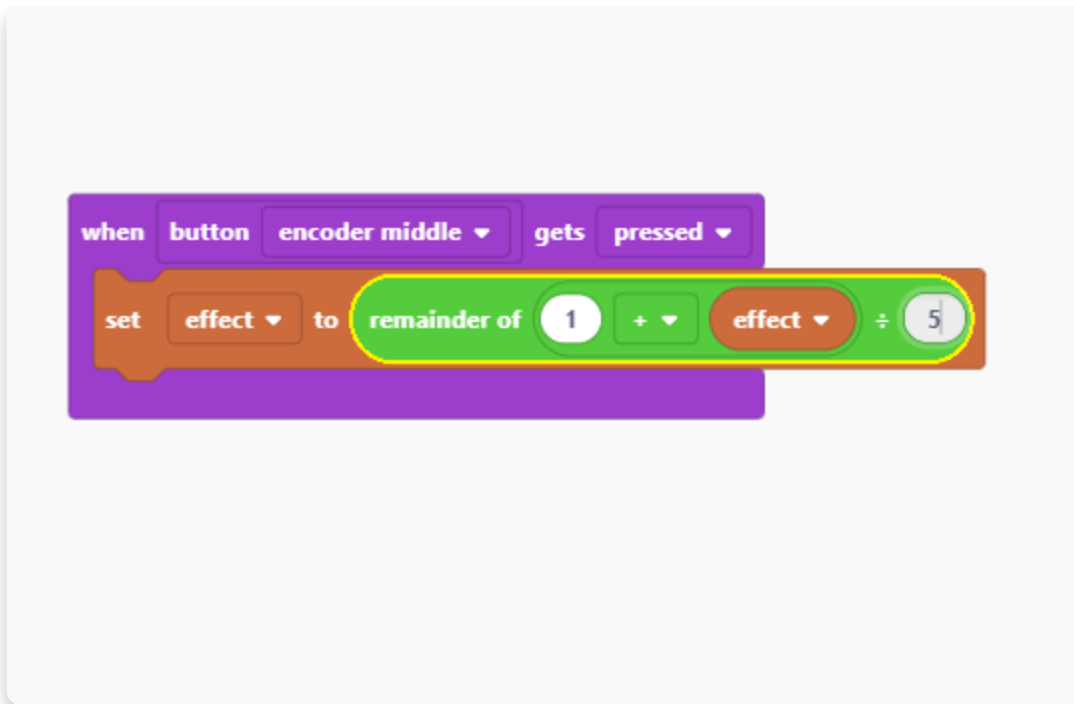
Search...

- Logic
- Loops
- Math
- Text
- Variables
- Functions
- Input
- LED matrix
- DJ
- Display
- Time

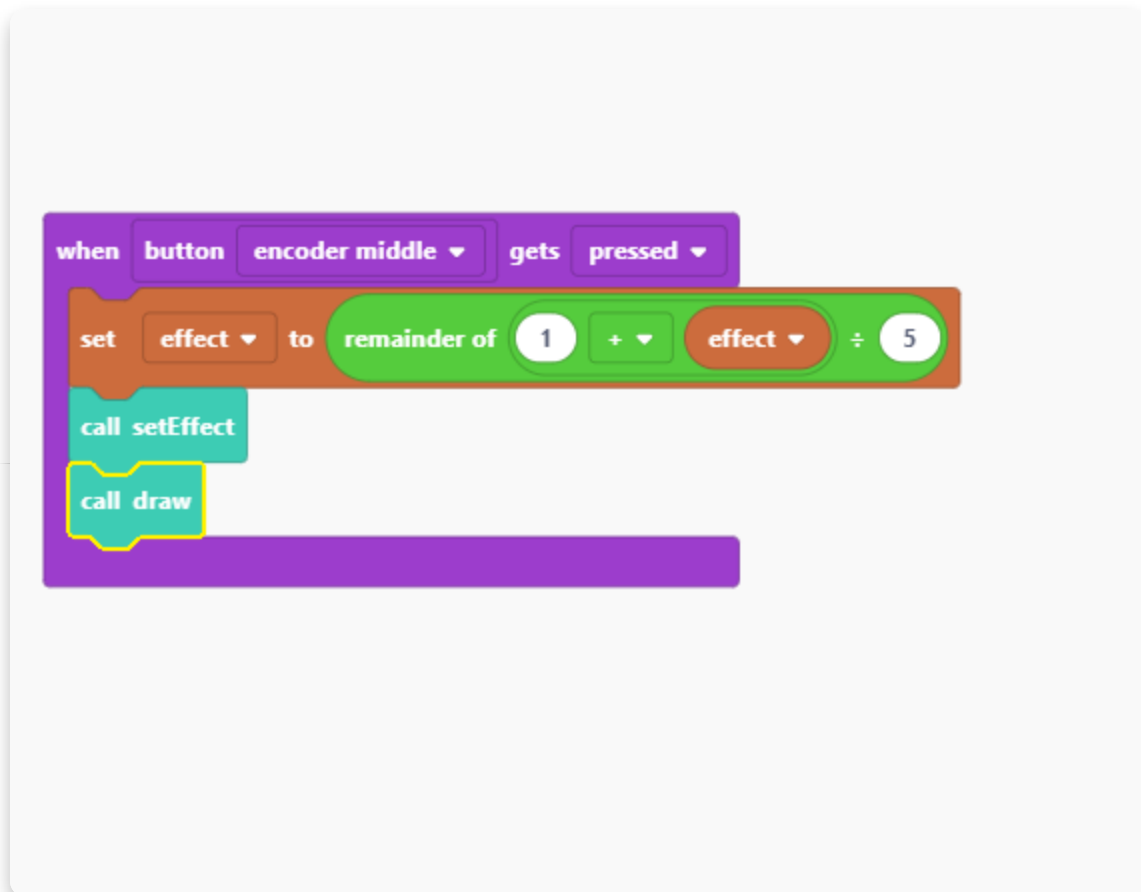
when button encoder middle gets pressed

set effect to remainder of 1 + 1 ÷ 10

Nun bitte noch die Variable „effect“ in das zweite Feld einfügen.



Füge jetzt bitte zwei der Funktionen hinzu, die wir bereits zuvor erstellt haben :
setEffect und draw.

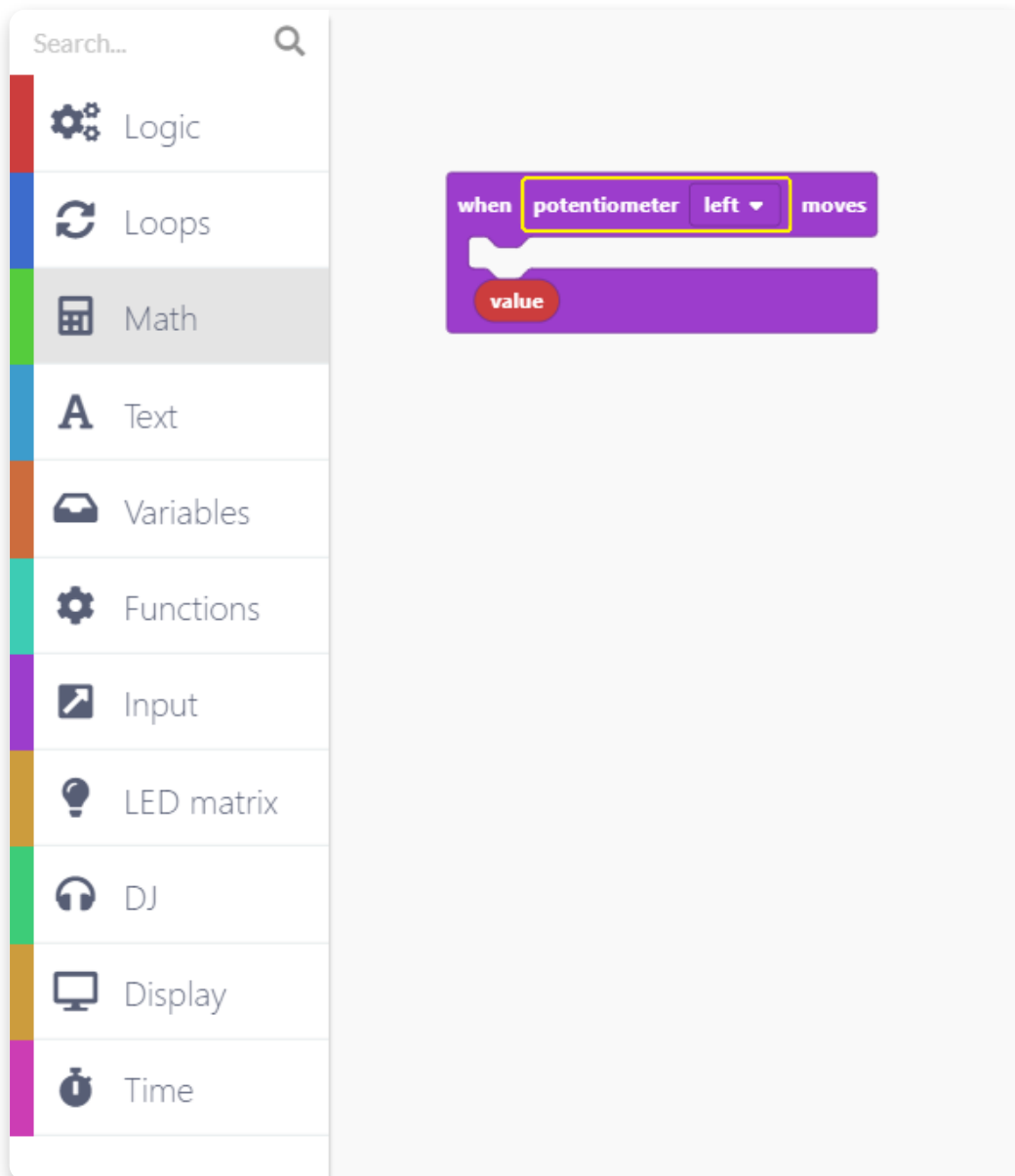


Mit dieser Bedienfunktion sind wir nun fertig.

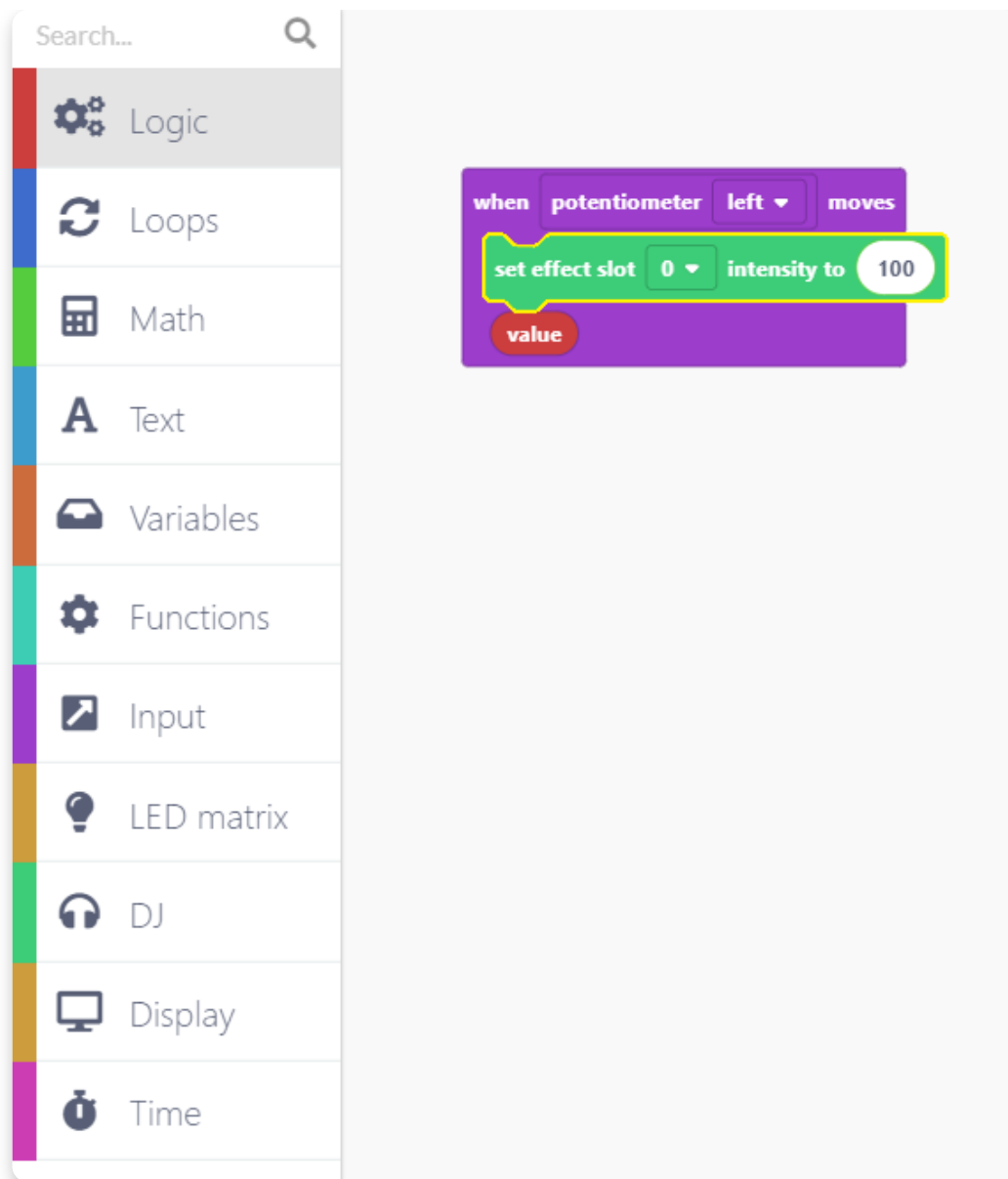
Lass uns nun die Steuerung der Effekt-Intensität mit dem linken Potentiometer

erstellen.

Öffne die Kategorie „Input“ und suche den Block „potentiometer“. Wähle im Aufklappmenü „left“ für das linke Potentiometer:



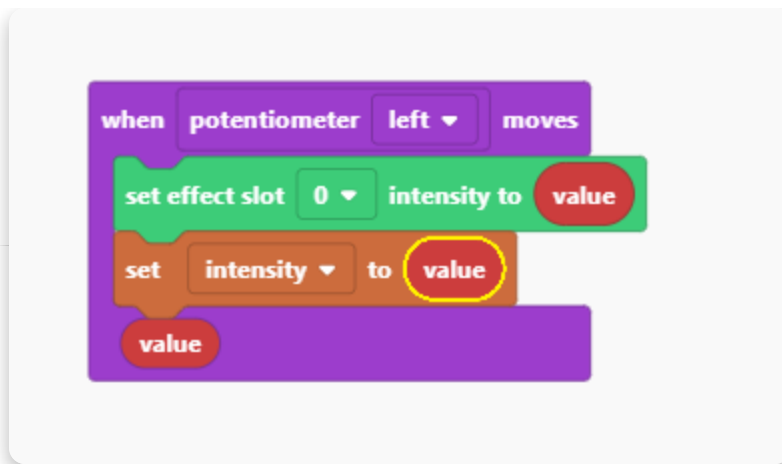
Jetzt benötigen wir noch einen Block aus der Kategorie „DJ“ der die Intensität steuert:



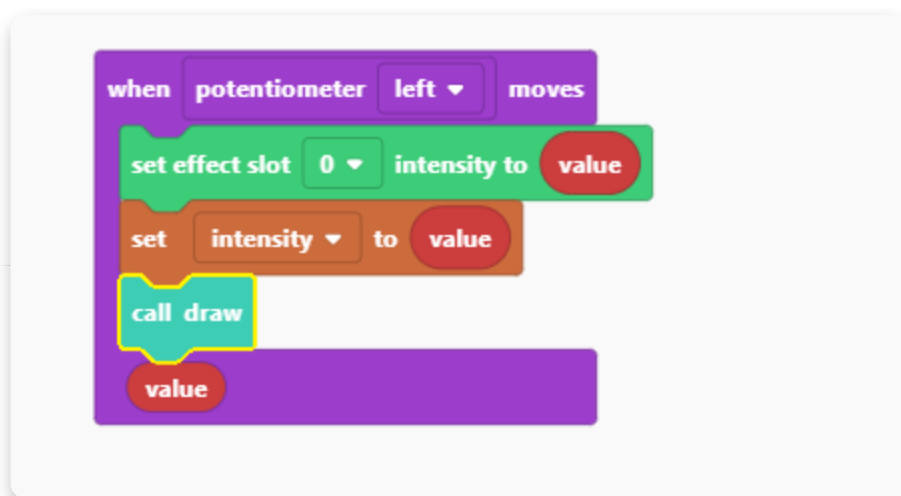
Ziehe nun vom unteren Ende des Input-Blocks den „value“-Block in das Feld, in der bis jetzt die Zahl 100 steht.



Füge nun eine Funktion hinzu, welche die Intensität auf den verknüpften (mit dem Potentiometer eingestellten) Wert setzt:



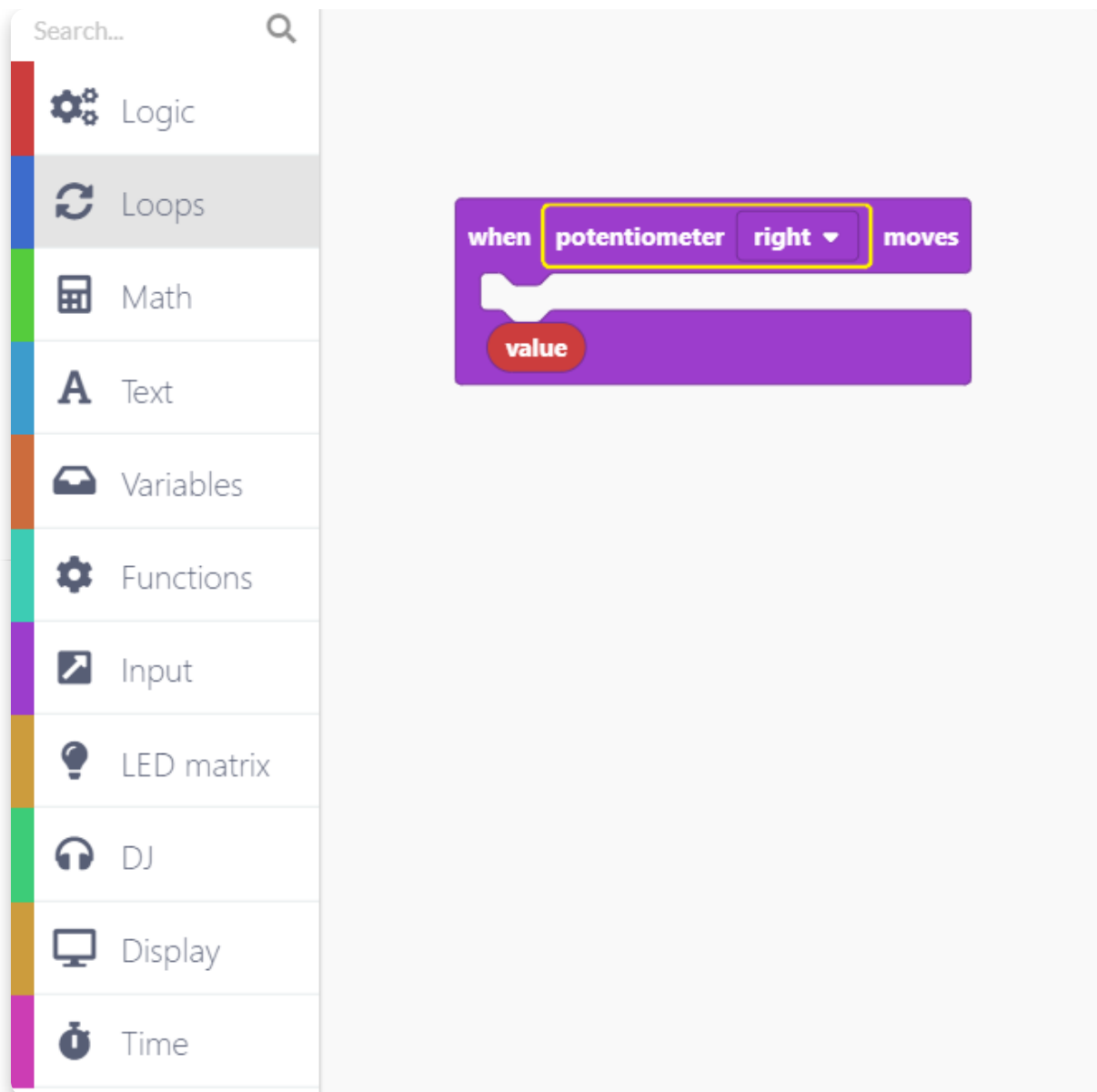
Als letztes fügen wir noch den Block „call draw“ zum Aufruf des Programmabschnitts „draw“ hinzu.



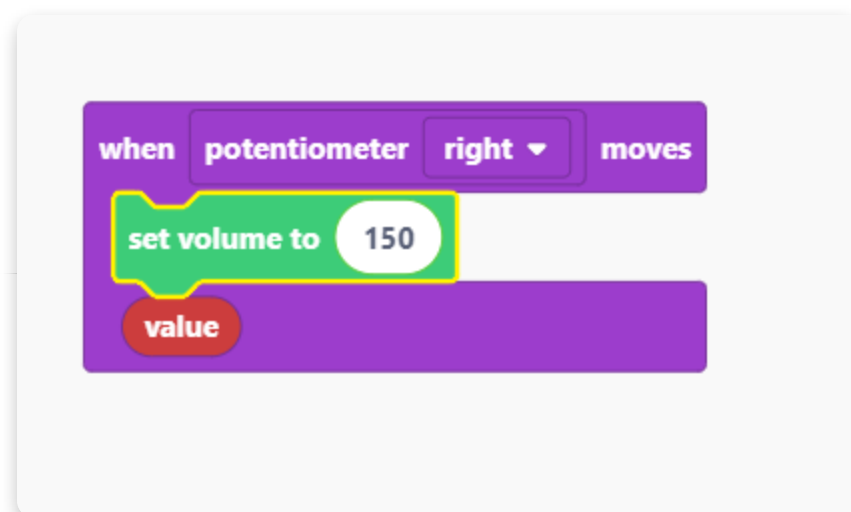
Die Abfrage des Potentiometers ist nun fertig und Dein Sketch sollte nun so aussehen:

```
Arduino run first:  
set effect to 0  
set intensity to potentiometer left current value  
set volume to potentiometer right current value  
open file pre-recorded House of God - Rahil ABRU featuring Teddy 12342  
call setEffect  
set volume to volume  
start playing  
call draw  
Arduino loop forever  
  
to: setEffect  
if effect == 0 do  
  set effect slot 0 to effect none  
else if effect == 1 do  
  set effect slot 0 to effect low pass  
else if effect == 2 do  
  set effect slot 0 to effect high pass  
else if effect == 3 do  
  set effect slot 0 to effect reverb  
else if effect == 4 do  
  set effect slot 0 to effect bit crusher  
set effect slot 0 intensity to intensity  
return  
  
to: draw  
set font size to 1  
set font type to 1  
set font colour to screen colour White  
clear the display with screen colour Black  
print create text with "Volume:"  
x 10  
y 20  
if effect == 0 do  
  set effectname to "none"  
else if effect == 1 do  
  set effectname to "low pass"  
else if effect == 2 do  
  set effectname to "high pass"  
else if effect == 3 do  
  set effectname to "reverb"  
else if effect == 4 do  
  set effectname to "bit crusher"  
print create text with "Effect:" effectname  
x 10  
y 40  
print create text with "Intensity:" intensity  
x 10  
y 60  
draw sprite to display  
return  
  
when button encoder middle gets pressed  
set effect to remainder of 1 + effect : 5  
call setEffect  
call draw  
  
when potentiometer left moves  
set effect slot 0 intensity to value  
set intensity to value  
call draw  
value
```

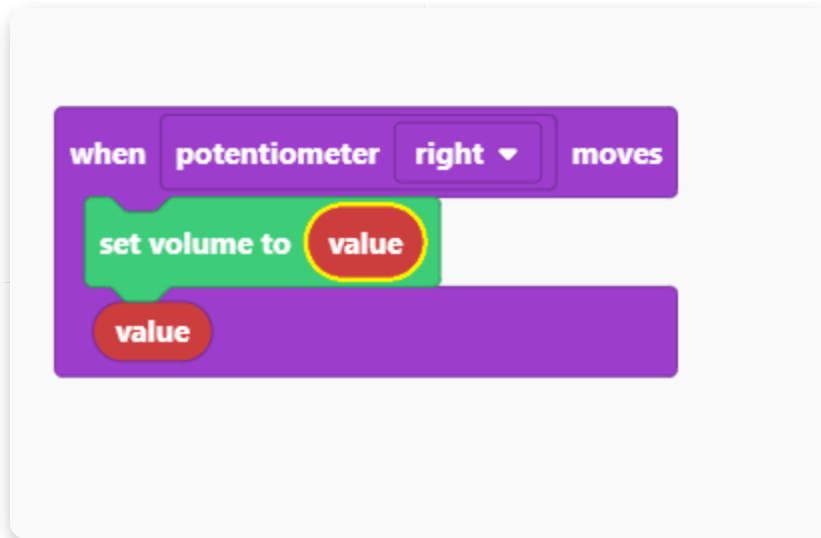
Verbleibt zum Schluss nun noch ein weiterer Block zur Lautstärkeeinstellung.



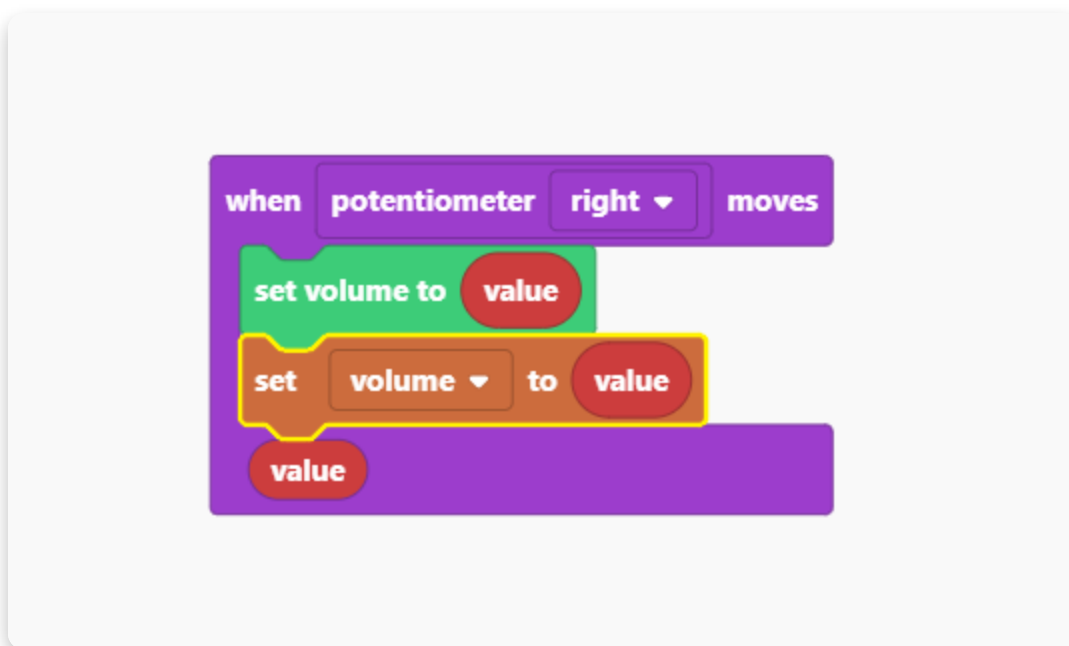
Öffne die Kategorie „DJ“ und wähle den Block „set volume to“ aus und lass ihn hier fallen:



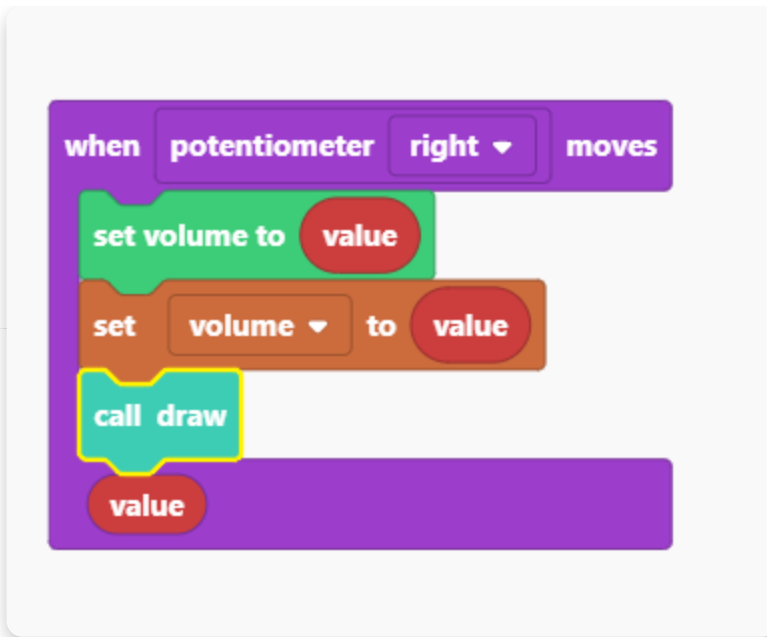
Zieh den kleinen, roten „value“-Block wieder vom unteren Ende in das Feld mit der Zahl:



Öffne nun die Kategorie „Variables“ und füge den Block „set“ hier ein:



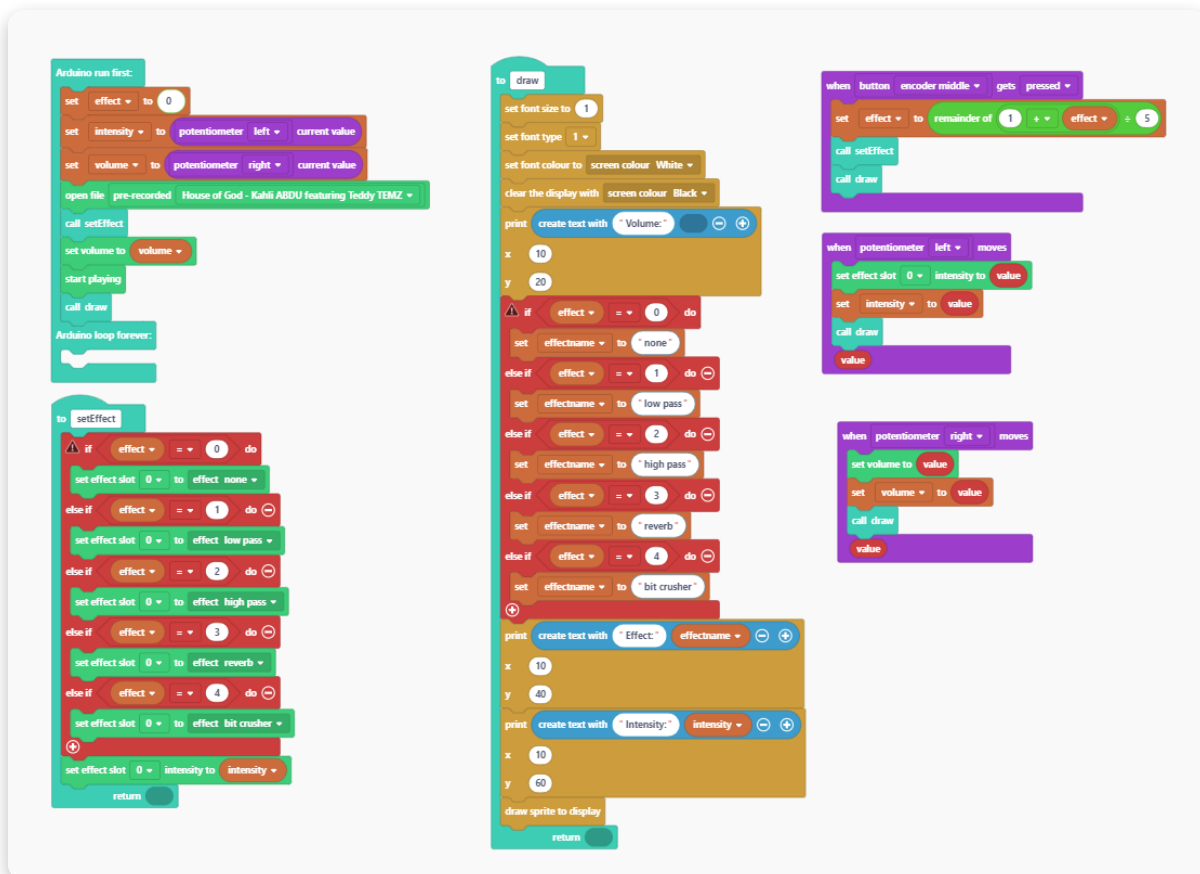
Nun fehlt noch die „call draw“ Funktion, die wieder ans Ende des Blocks kommt.



Das wars!

Jetzt sollte Dein Sketch fertig zum Start sein. Es kann allerdings ein paar Augenblicke dauern bis alles kompiliert ist. Habe also bitte etwas Geduld.

Wenn Dein Projekt nicht startet, überprüfe bitte nochmals Deinen Sketch anhand der Beispielfotos in dieser Anleitung.

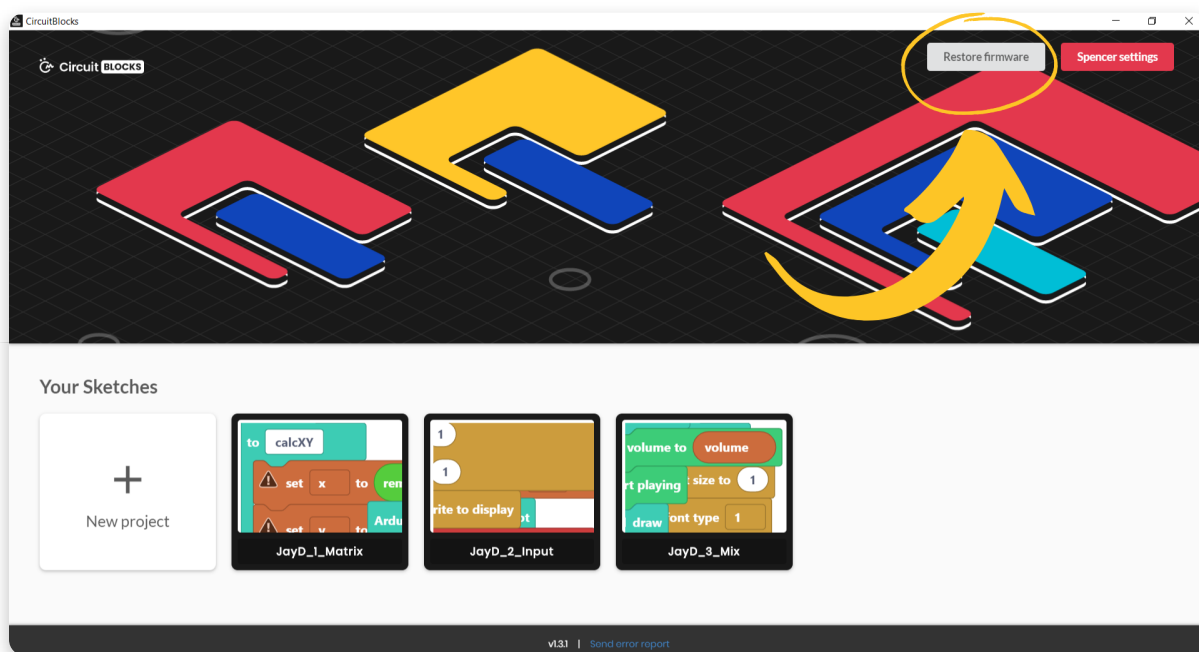


Die Firmware zurücksetzen

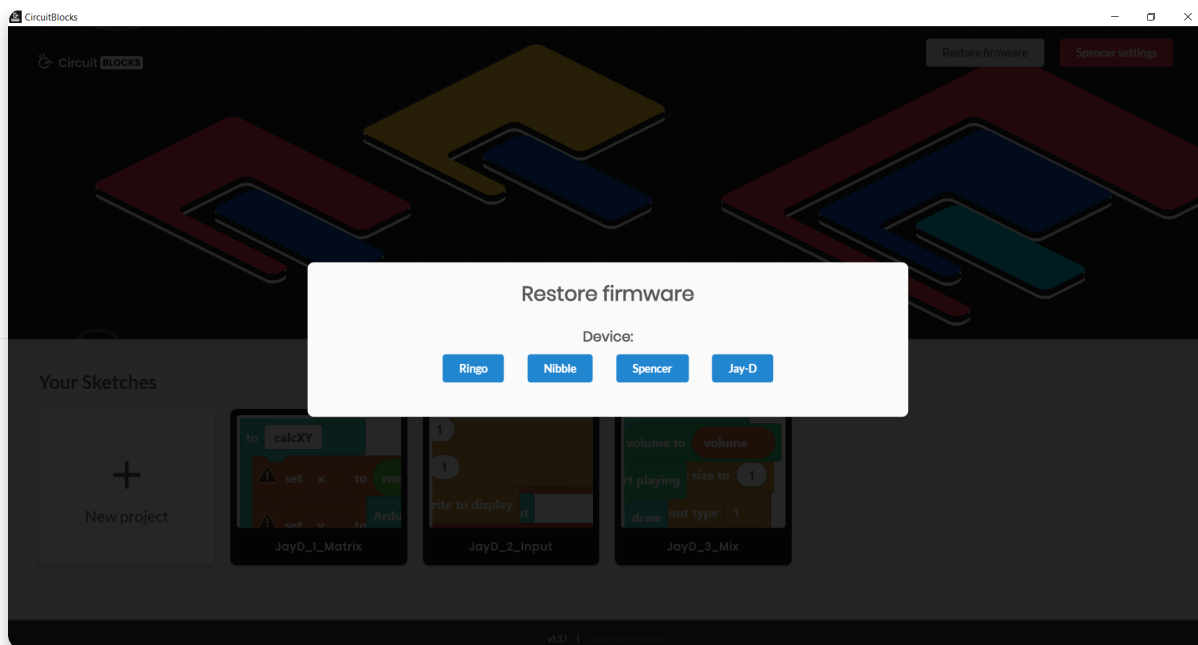
Die Firmware zurücksetzen

Wenn Du mit Deinen Programmierversuchen fertig bist, möchtest Du vielleicht wieder Deinen „normalen“ Jay-D zurück haben. Dafür musst Du wieder seine Basis-Firmware laden.

Das ist schnell und einfach gemacht: Verbinde Jay-D mit einem USB Anschluss an Deinem Computer und drücke in der Software CircuitBlocks oben rechts auf den Knopf „Restore firmware“.



Es öffnet sich ein Fenster in dem Du nach dem Modell Deines Bausatzes gefragt wirst. Hier wählst Du, ganz klar, Jay-D aus.



Nach ein paar Sekunden ist wieder die Basis Firmware in Jay-D geladen und funktioniert wie gewöhnlich.

Das kannst Du immer dann machen, wenn Du mit dem Programmieren fertig bist und wieder den ursprünglichen (Lieferzustand) haben möchtest.

Werde kreativ!

Herzlichen Glückwunsch! Du hast das Ende unser Programmieranleitung für Jay-D erreicht.

Wir möchten mit künftigen Firmware-Updates und neuen Versionen von CircuitBlocks die Funktionen von Jay-D um neue, coole Dinge erweitern und Hoffen Du bist genau so gespannt darauf, wie wir.

Bis dahin experimentiere doch mal auf eigene Faust mit Jay-Ds verschiedenen Licht-, Display- und Soundfunktionen. Teile Deine Erfahrungen und finde neue Ideen und Anregungen in unserem CircuitMess Gruppenforum:

<https://community.circuitmess.com/>

Bereit für fortgeschrittene Dinge? Sobald Sie CircuitBlocks beherrschen und bereit für eine erweiterte Programmierung sind, versuchen Sie es mit **PlatformIO** zum Programmieren neuer Funktionen:

Schauen Sie sich hier unsere Anleitung dazu an:

- [PlatformIO Jay-D programming guide](#)

Wenn Du Hilfe mit Deinem Bausatz benötigst oder Fragen hast, schreibe uns doch einfach eine Email an: contact@circuitmess.com und wir werden Dir so schnell wie möglich antworten.

Vielen Dank & mach weiter so!

