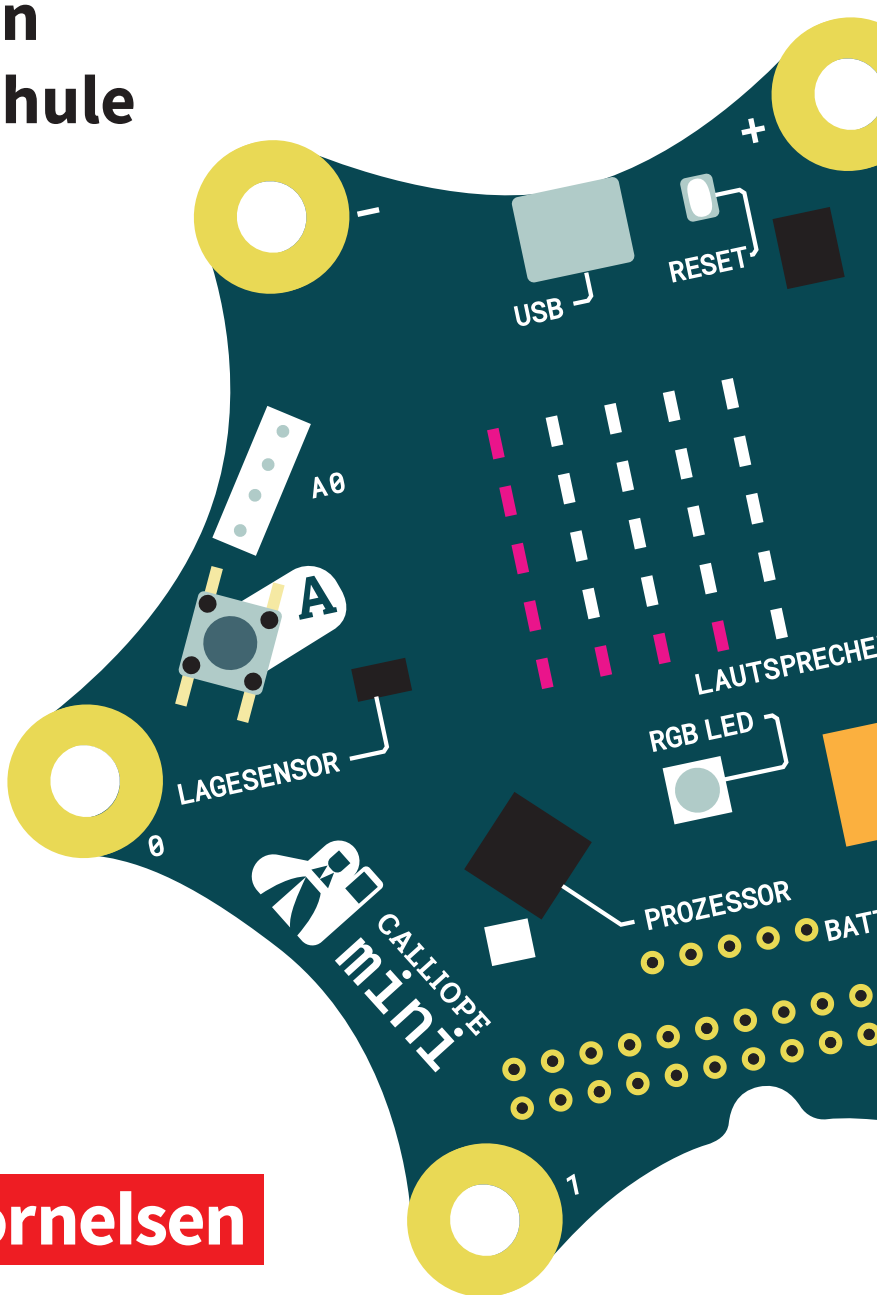
Für Lehrer

Coden mit dem Calliope mini

Programmieren in der Grundschule

Lehrermaterial für den Einsatz ab Klasse 3



Cornelsen

Coden mit dem Calliope mini

Programmieren in der Grundschule

Lehrermaterial
für den Einsatz ab Klasse 3

Autoren: Michael Abend (Glossar, Der *Calliope mini* und das Nim-Spiel, Der *Calliope mini* als Rechtschreibtrainer, Der *Calliope mini* als Stoppuhr und Countdown-Zähler),
Kirstin Gramowski (Morsen mit dem *Calliope mini*, Der *Calliope mini* als 1x1-Kopfrechentrainer, Der *Calliope mini* als Zufallsgenerator, Nachbarzahlen mit dem *Calliope mini* bestimmen),
Lars Pelz (Der *Calliope mini* als Taktgeber, Der *Calliope mini* als Minipiano, Der *Calliope mini* als automatisches Fahrradrücklicht, Bildimpulse und Reizwörter mit dem *Calliope mini* erzeugen),
Bernd Poloczek (Lernlandkarte, Nachwort (Erfahrungen & Gelingensbedingungen))

Berater: Michael Abend, Kirstin Gramowski, Lars Pelz, Bernd Poloczek

Redaktion: Patrizia Schwarzer

Illustration: Calliope gGmbH, Berlin: Cover, S. 3, S. 5, S. 62/63, Rückseite
zweiband.media GmbH, Berlin: alle verbleibenden Illustrationen im Innenteil

Foto: Sibylle Baier, Berlin: S. 64

Umschlaggestaltung: COSAKitchen, Corinna Babylon, Berlin

Layout, Grafik und technische Umsetzung: zweiband.media GmbH, Berlin

www.cornelsen.de
www.cornelsen.de/calliope

1. Auflage, 1. Druck 2017

Alle Drucke dieser Auflage sind inhaltlich unverändert und können im Unterricht nebeneinander verwendet werden.

ISBN: 9783066000122

Druck: AZ Druck und Datentechnik GmbH, Kempten

PEFC zertifiziert
Dieses Produkt stammt aus nachhaltig bewirtschafteten Wäldern und kontrollierten Quellen.
www.pefc.de
PEFC/04-31-2260

Inhaltsverzeichnis

Liebe Lehrerinnen und Lehrer, liebe pädagogische Fachkräfte,

das Schreiben von Computerprogrammen, das sogenannte Coding ist in der Schule ein noch recht junger Trend, der den heranwachsenden Generationen die Möglichkeit bietet, sich technologisches Wissen zu erschließen und für die handlungsorientierte Entdeckung der Welt zu nutzen.

Der Cornelsen Verlag unterstützt die Initiative der *Calliope gGmbH*, mithilfe derer es jedem Schulkind in Deutschland bereits ab der 3. Klasse möglich sein soll über einen kleinen programmierbaren Mikrocontroller, den *Calliope mini*, einen spielerischen Zugang zur digitalen Welt zu erhalten.

*„Mit dem **Calliope mini** können schon Grundschulkinder kreativ und spielerisch lernen, wie die digitale Welt funktioniert." (Gesche Joost, Professorin für Designforschung an der Universität der Künste Berlin und Mitbegründerin Calliope gGmbH)*

Auf den ersten Blick erscheint das Programmieren, zudem noch in der Grundschule als nicht triviale Notwendigkeit. Weder die Lehrpläne der Grundschule noch andere (bildungspolitische) Verordnungen verleihen dem Thema Verbindlichkeit. Wird das Programmieren jedoch nicht als Selbstzweck betrachtet, sondern in einen allgemeineren Kontext gerückt, öffnet sich ein breites Feld an Anknüpfungspunkten, insbesondere auch für grundschulrelevante Belange.

Ein Aspekt ist die Bedeutsamkeit von Medienkompetenz. Diese umfasst weitaus mehr als die reine Bedienkompetenz digitaler Geräte. So nimmt nicht nur das Arbeiten mit Medien, sondern auch das Lernen über Medien eine große Rolle ein. Ein Kompetenzaufbau im Bereich Coding trägt hierüber hinausgehend zu einem Grundverständnis der uns umgebenden digitalen Welt bei, das uns erlaubt, diese technologische „black box" zu entmystifizieren. Statt in einem rezeptiven Stadium zu verharren, befähigt ein erstes Verständnis des Programmierens zum kritisch-reflektierten Urteil, wie bspw. dazu, sich Technologie zunutze zu machen (z. B. Routinen abarbeiten lassen) und einen souveränen Umgang mit technologischen Prozessen anzubahnen. Darüber hinaus wird eine Teilhabe an der digitalen Gesellschaft ermöglicht.

Innerhalb des Unterrichts eröffnet sich durch das Programmieren eine Bandbreite an Kompetenzen, die gezielt gefördert werden: Coden ...

- fördert das forschend-entdeckende Lernen durch eine spielerische Herangehensweise und fehlertolerantes Arbeiten.
- fördert das Abstraktionsvermögen, die Modellierungs- und Problemlösekompetenz sowie das analytische Denken und klare Strukturieren.
- fördert übergreifende Kompetenzen: Selbstständigkeit, Eigeninitiative und das Bilden eigener Hypothesen.
- fördert konstruktive Vorgehensweisen und kreativ-produktive Lösungsprozesse.
- trägt zu einer hohen Schüleraktivierung bei: Coden schafft Handlungs- und Gesprächsanlässe.
- begünstigt eine natürliche Differenzierung durch viele richtige Wege zum (selbst gesetzten) Ziel und bietet vielfältige Gestaltungsspielräume.

Als didaktischer Kooperationspartner der *Calliope gGmbH* bündeln wir als Cornelsen Verlag die oben genannten Punkte und überführen sie im vorliegenden Lehrermaterial in konkrete Beispiele mit Lehrplanbezug.

Das erwartet Sie:

- 11 Coding-Beispiele mit dem *Calliope mini* zu Inhalten der Lehrplänen der Fächer Sachkunde, Deutsch und Mathematik der Grundschule ab Klasse 3
- Schritt-für-Schritt-Anleitungen für Programmierneulinge – Ganz ohne Programmiervorkenntnisse erstellen Sie eigene Programme und bauen systematisch Ihre Coding-Kompetenz auf.
- Coden jederzeit als Werkzeug zur Problemlösung zu verstehen und zu erkennen, dass es nie nur einen Selbstzweck erfüllt
- ausgewählte Beispiele, die einen anderen Zugang zu Lehrplanthemen bieten (u. a. das Beispiel Morsen mit dem *Calliope mini*) – die mit herkömmlichen Medien (Buch, Heft, ...) nicht oder nicht so schnell zu erstellen sind (z. B. der *zufallsgesteuerte 1x1-Kopfrechentrainer*) – oder konkrete Anwendungsbezüge aufzeigen (z. B. Der *Calliope mini als automatisches Fahrradrücklicht*)

Probieren Sie es einmal für sich selbst und später in Ihrem Unterricht aus!

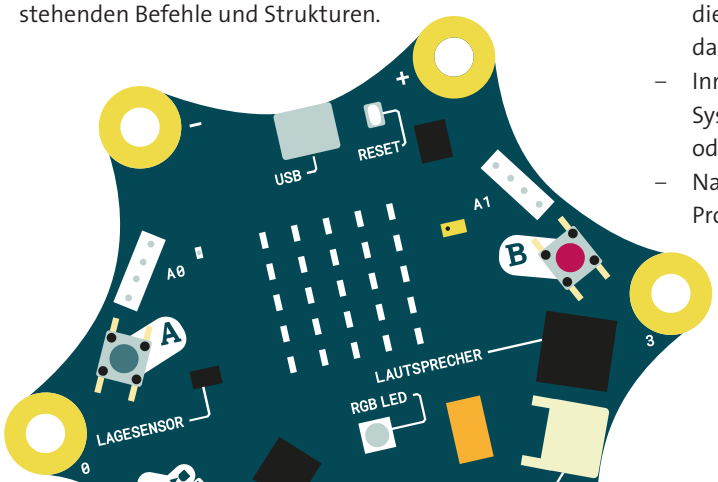
Wie „denkt“ ein Computer?

Ein Computer kann glücklicherweise (noch) nicht selbstständig denken. Er besteht aus einem elektronischen Schaltkreis (Prozessor), der nur mit Einsen und Nullen rechnen kann, allerdings in einer kaum vorstellbaren Geschwindigkeit. Was und wie der Computer rechnen soll, muss ihm vorher „beigebracht“ werden. Diese Tätigkeit nennt man Programmieren oder Coden.
Wenn Menschen Aufgaben zu bewerkstelligen haben (wir sprechen in diesem Zusammenhang von Problemen), deren Lösung zeitaufwändig und arbeitsintensiv ist, können sie die Bearbeitung dieser Probleme auf einen Computer übertragen. Dazu muss die Vorgehensweise zur Lösung der Probleme als eine Folge eindeutiger Anweisungen formuliert werden. Eine solche Folge nennt man in der Informatik „Algorithmus“.

Ein Algorithmus kann in einer Programmiersprache aufgeschrieben werden. Das ist eine Art Zwischenlösung, eine Sprache, die für den Menschen noch lesbar ist, jedoch auch vom Computer verarbeitet werden kann. Das daraus entstandene Programm stellt die elektronischen Schaltkreise des Computers so ein, dass er genau die Arbeitsschritte durchführt, die zuvor im Algorithmus beschrieben wurden. Der Aufbau eines Algorithmus folgt zumeist dem grundlegenden Prinzip 1) Eingabe (Informationen, die den Computer aus seiner Umgebung erreichen), 2) Verarbeitung (nach den Vorschriften des Algorithmus, wobei neue Informationen entstehen) und 3) Ausgabe (der neu entstandenen Informationen über verschiedene Wege).

Coden mit einer graphischen Programmiersprache am Beispiel von *NEPO®*

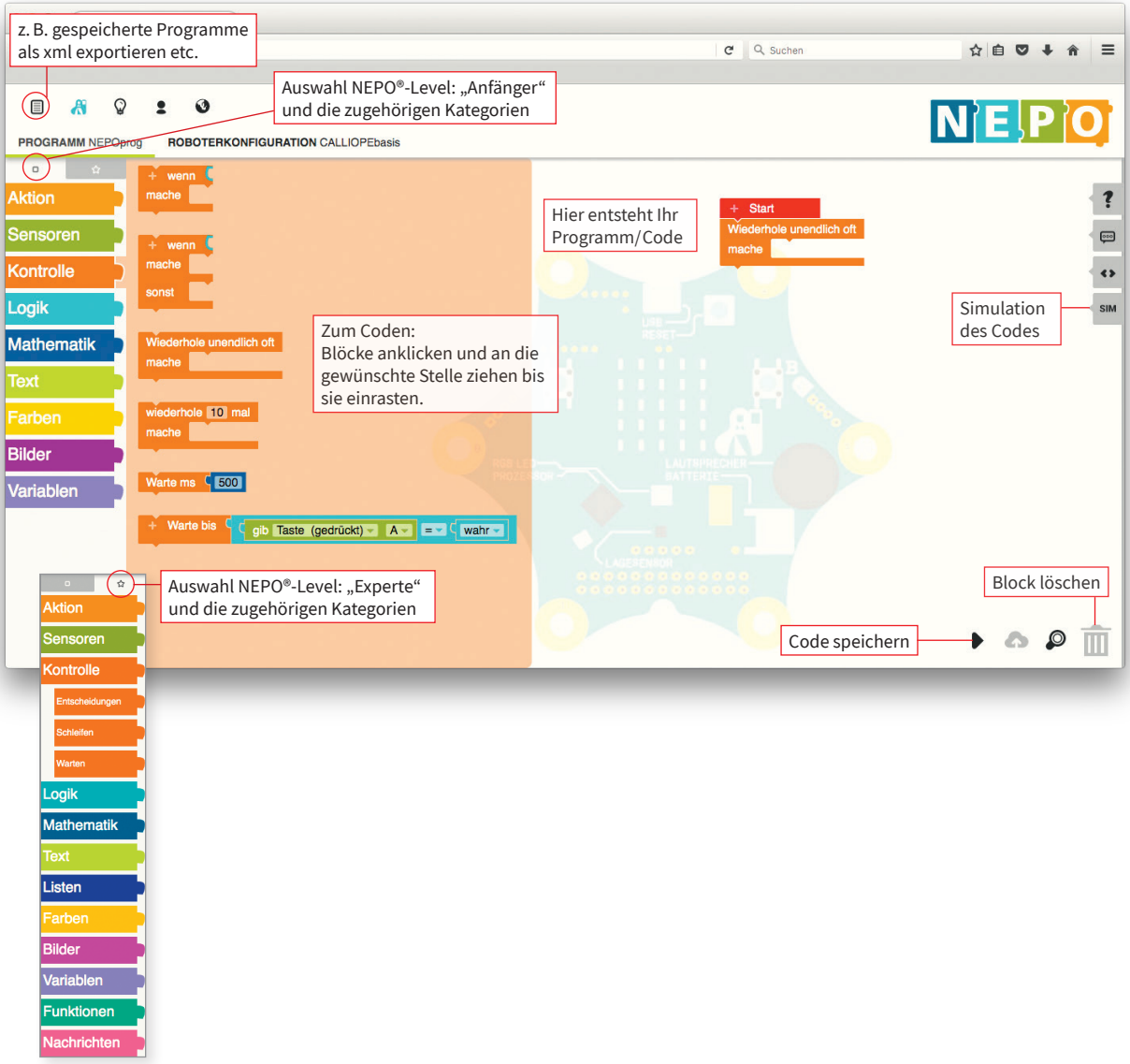
Programmiersprachen bestehen im Grundsatz aus Befehlen, Entscheidungen und Wiederholungen. Eine im Computer gespeicherte Abfolge dieser Anweisungen nennt man Programm. Programme beschreiben eindeutig und genau, was der Computer, der ein Programm ausführt, tun soll, um ein damit verbundenes Problem zu lösen. Es gibt verschiedene Programmiersprachen (textuelle und graphische), mit denen Programme geschrieben werden. Der Begriff Coden bezeichnet dabei die Tätigkeit der Erstellung eines Programms – also das kodieren einer Anweisungsablauf zur Lösung eines Problems für den Computer.
Im Gegensatz zu textuellen Programmiersprachen, die ein hohes Abstraktionsvermögen erfordern, bieten graphische Programmiersprachen einen einfachen Einstieg ins Coden. Mit ihnen können die Grundkonzepte der Computerprogrammierung spielerisch und explorativ erlernt werden. Dabei werden vorgefertigte Code-Bausteine (Blöcke) wie Puzzleteile ineinandergeschachtelt, um Programme zu erstellen. Sogenannte Editoren (Oberflächen, innerhalb derer programmiert wird) ermöglichen einen schnellen und ganzheitlichen Überblick über die zur Verfügung stehenden Befehle und Strukturen.



Ein weiterer Vorteil der graphischen Programmierung besteht darin, dass Programmelemente (Blöcke) nur so zusammengefügt werden können, wo sie laut zugrundeliegendem Regelsystem erlaubt sind. So werden Syntaxfehler vermieden, die beim Coden in textuellen Programmiersprachen ein großes Frustrationspotenzial in sich bergen. Graphische Programmiersprachen sind heute zahlreich vertreten. Im Kern funktionieren sie nach stets gleichen Prinzipien. Wenn Sie die Grundidee des Codens, also die Formulierung einer konkreten Anweisungsfolge als Problemlöserezept, verstanden haben, können Sie sofort in jede graphische Programmiersprache einsteigen.
In den nachfolgenden Beispielen kommt die graphische Programmiersprache *NEPO®* im *Open Roberta Lab®* vom Fraunhofer-Institut zum Einsatz. Der internetbasierte Editor dieser Open-Source-Plattform erfordert keine Installation, nur eine Internetverbindung und einen aktuellen Browser.
Im Folgenden lernen Sie den Editor *Open Roberta Lab®* und die Programmiersprache *NEPO®* kennen:

- Rufen Sie zunächst in einem aktuellen Browser die Website calliope.cc/editor auf. Klicken Sie dann auf den Editor *Open Roberta Lab®*.
- Innerhalb des *Open Roberta Lab®* wählen Sie das System *Calliope* und Ihre Version (*Calliope 2016* oder *Calliope 2017*) aus.
- Nach der Auswahl des Systems können Sie das Programmieren mit *NEPO®* starten.

Das *Open Roberta Lab*® und die Programmiersprache *NEPO*®



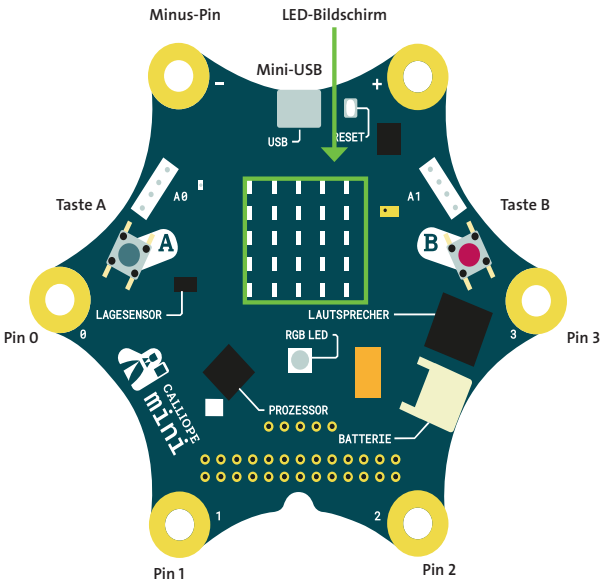
Vorgehen bei den nachfolgenden Beispielen sowie allgemeine Tipps beim Coden:

1. Anfänger- oder Expertenlevel wie beim Beispiel angegeben einstellen. Die beiden Modi sagen nichts über die Schwierigkeit aus, sondern beinhalten mehr oder weniger Blöcke.
2. Bei einem Klick auf die *NEPO*®-Kategorien erscheinen die zugehörigen Blöcke. Eine Erklärung zum jeweiligen Block erscheint, wenn der Mauszeiger über diesem verweilt.
3. Wählen Sie den benötigten Block aus und ziehen Sie ihn mit gedrückter linker Maustaste unter den roten „Start“-Block. Alle weiteren Blöcke platzieren Sie an den entsprechenden Stellen innerhalb des Codes, so dass der neue Block in die schon bestehende Struktur einrastet.
4. Der Klick auf die rechte Maustaste bietet Ihnen eine Reihe weiterer Optionen (z. B. Kopieren, Löschen eines Blocks)
5. Eine umfassende Erläuterung der Blöcke und ihrer Einsatzmöglichkeiten finden Sie unter: https://mp-devel.iais.fraunhofer.de/wiki/pages/viewpage.action?pageId=10847065
6. Bereits am Bildschirm kann das Programm innerhalb des Simulationsmodus ausgeführt werden. Sie können am Bildschirm prüfen, ob Ihr Programm nach Ihren Vorstellungen abläuft. Um diese Simulation auszuführen, klicken Sie auf „SIM“ in der rechten Randspalte.
7. Speichern des Codes: Zum Übertragen des erstellten Programms auf den *Calliope mini* und zum anschließenden Testen klicken Sie auf den „Play“-Knopf unten rechts und folgen der Anleitung auf dem Bildschirm.
8. Legen Sie sich ein Profil an, falls Sie Ihre Programme von überall zugänglich machen, an Programmen weiterarbeiten oder sie mit anderen teilen wollen.

Den *Calliope mini* mit *NEPO*® programmieren

Für den Einstieg in die Programmierung eignen sich sogenannte Mikrocontroller oder Kleinstcomputer. Diese verfügen über sehr einfache Ein- und Ausgabemöglichkeiten (z. B. wenige Knöpfe statt einer Tastatur, einige LEDs statt eines hochauflösenden Displays). Der *Calliope mini* ist ein solcher Mikrocontroller, der sich u. a. über eine graphische Programmiersprache steuern lässt und nach den Prinzipien eines Algorithmus arbeitet. Für die Eingabe hat der *Calliope mini* z. B. Sensoren, die u. a. die Lichtstärke, seine Lage im Raum oder die Stärke des Erdmagnetfeldes messen können. Auch können Tasten gedrückt werden, um Eingaben zu erzeugen. Zur Ausgabe stehen beim *Calliope mini* z. B. ein. LED-Bildschirm, eine RGB-LED und ein Lautsprecher zur Verfügung. Des Weiteren können Motoren angeschlossen werden, deren Drehzahl und Drehrichtung der Prozessor im *Calliope mini* steuern kann.
Wenn Sie demnächst Ihr erstes eigenes Programm mit *NEPO*® gecodet haben, können Sie dieses auf Ihren *Calliope mini* überspielen und dort ausführen lassen.
Alle Beispiele im vorliegenden Titel verbinden beide Bestandteile (*NEPO*® und *Calliope mini*) und zeigen verschiedene Möglichkeiten, wie Sie den *Calliope mini* programmieren können, um ihn sinnvoll in Ihrem Fachunterricht einsetzen zu können.

Hier die wichtigsten Funktionen des Calliope mini



Weitere Infos über Funktionalitäten und Sensoren finden Sie unter: calliope.cc/ueber-mini

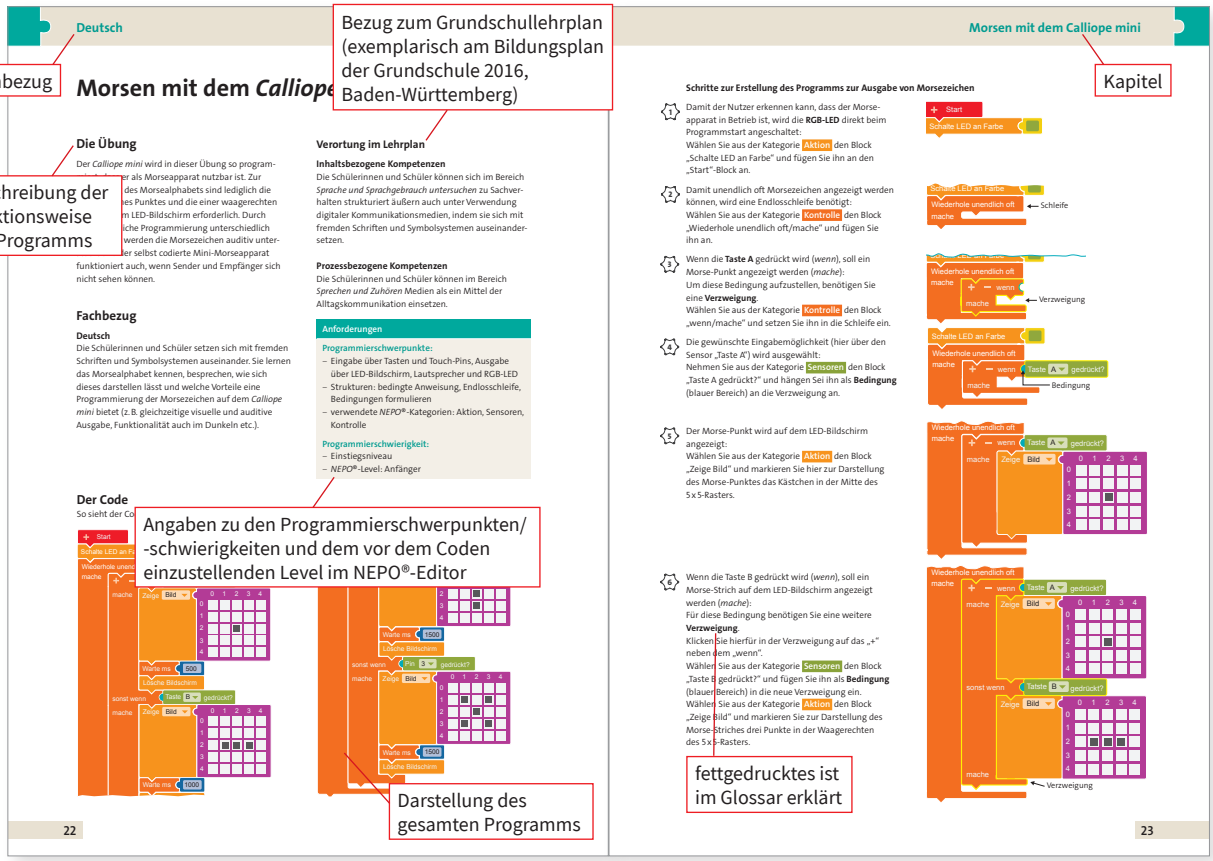
Das benötigen Sie, um die Beispiele auszuprobieren

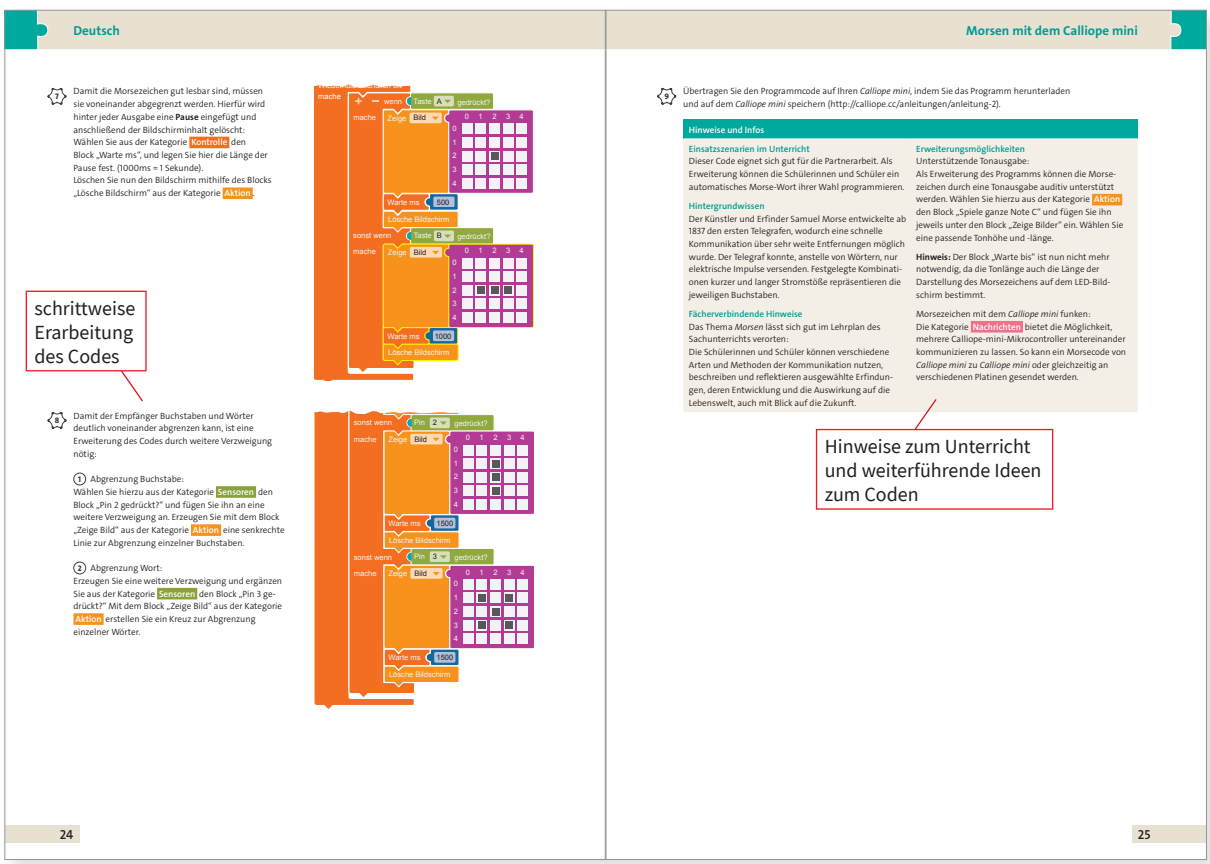
- Ein internetfähiges Endgerät (z. B. Laptop, PC)
- Den webbasierten Editor *NEPO*® vom *Open Roberta Lab*® für die Programmierung, zu finden auf calliope.cc/editor
- Ein Akku-Pack, um den *Calliope mini* mit Strom zu versorgen
- Ein Verbindungskabel (USB-mini zu USB), um den Code auf den *Calliope mini* zu übertragen
- Weitere Hilfestellungen finden Sie unter calliope.cc/anleitungen

So übertragen Sie Ihr Programm auf den *Calliope mini*:

- Verbinden Sie den *Calliope mini* per USB-auf USB-mini-Kabel mit dem Rechner.
- Der *Calliope mini* erscheint als Speichermedium (USB-Laufwerk).
- Speichern Sie die aus *NEPO*® heruntergeladene Programm-Datei entweder direkt auf dem *Calliope mini* oder verschieben die Datei vom lokalen Speicherort auf das „MINI"-Laufwerk.
- Während die Programmübertragung läuft, blinkt das Lämpchen links des USB-Anschlusses.
- Nach Abschluss der Übertragung leuchtet das Lämpchen dauerhaft solange die Platine mit Strom versorgt wird.
- Jetzt ist der *Calliope mini* bereit, das übertragene Programm auszuführen. Bei *Calliope 2016* müssen Sie zuvor die Reset-Taste drücken.
- Hinweis: Der *Calliope mini* speichert stets nur ein Programm. Wird ein neues Programm übertragen, so überschreibt dieses das bereits aufgespielte Programm.
- Starten Sie das Programm auf dem *Calliope mini*, indem Sie die in Ihrer Programmierung festgelegte Eingabe auslösen, z. B. indem Sie Taste A drücken.
- Anschließend können Sie das Ergebnis Ihres Programms auf dem *Calliope mini* sehen.

Aufbau der Beispiele





Der *Calliope mini* als automatisches Fahrradrücklicht

Die Übung

Der *Calliope mini* wird als automatisches Fahrradrücklicht programmiert. Die Beleuchtung soll sich einschalten, wenn es dunkel wird und sich wieder ausschalten, sobald es hell genug ist. Dazu wird der im *Calliope mini* eingebaute Lichtsensor abgefragt.

Der Lichtsensor des *Calliope mini* wandelt das bei ihm einfallende Licht in einen Zahlenwert um, damit die jeweiligen Lichtstärken vom Prozessor des *Calliope mini* unterschieden werden können. Je geringer der Lichteinfall, desto geringer ist der damit verknüpfte Zahlenwert. Der Programmierer muss festlegen, bei welchen vom Sensor ermittelten Zahlenwerten die Grenze zwischen „dunkel“ und „hell“ gezogen wird.

Fachbezug

Sachkunde
Bei der Programmierung der automatischen Fahrradbeleuchtung lernen die Schülerinnen und Schüler, dass der *Calliope mini* die Stärke des bei ihm einfallenden Lichtes messen kann. Sie stellen fest, dass sie die Lichtstärke variieren können, indem sie den *Calliope mini* abschatten.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Raum und Mobilität* Voraussetzungen für die sichere Teilnahme am Verkehr beschreiben und umsetzen sowie ihr Fahrrad hinsichtlich seiner Verkehrssicherheit überprüfen und warten.

Prozessbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Kommunizieren und sich verständigen* Ideen, Lern- und Lösungswege, gewonnene Erkenntnisse ausdrücken (z. B. bei der Planung und dem Bau technischer Produkte oder beim Vergleich von Verkehrsmitteln).

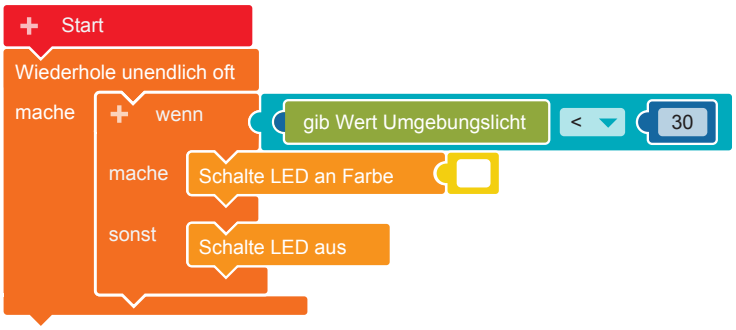
Anforderungen

Programmierschwerpunkte:
- Lichtsensor abfragen
- verwendete Programm-Strukturen: Endlosschleife, Verzweigung, Bedingung, Befehl
- verwendete *NEPO*®-Kategorien: Aktion, Kontrolle, Sensoren, Mathematik, Logik

Programmierschwierigkeit:
- Einstiegsniveau
- *NEPO*®-Level: Anfänger

Der Code

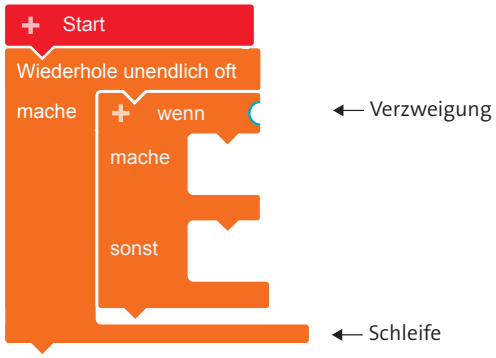
So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.



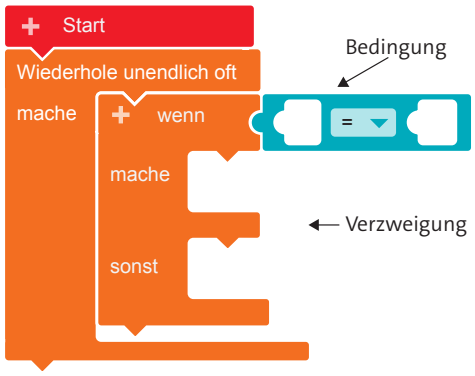
Schritte zur Erstellung des Programms für ein automatisches Fahrradrücklicht

1 Damit der *Calliope mini* jederzeit auf eine Änderung des Lichteinfalls reagieren kann, muss die Lichtstärke ununterbrochen gemessen werden: Dafür benötigen Sie eine Endlos**schleife.** Alle **Blöcke**, die Sie in die Endlosschleife einsetzen, werden ständig wiederholt. Wählen Sie aus der Kategorie **Kontrolle** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.

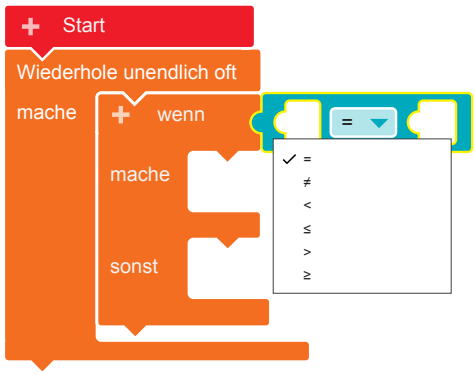
2 Die gemessene Lichtstärke soll in zwei Bereiche eingeteilt werden: „dunkel“ und „hell“:
Dafür benötigen Sie eine **Verzweigung**.
Wählen Sie aus der Kategorie Kontrolle den Block „wenn/mache/sonst“ und fügen Sie ihn in die **Schleife** ein.
Diese Verzweigung dient dazu, anhand einer **Bedingung** zwischen zwei alternativen **Programm**abläufen zu unterscheiden.



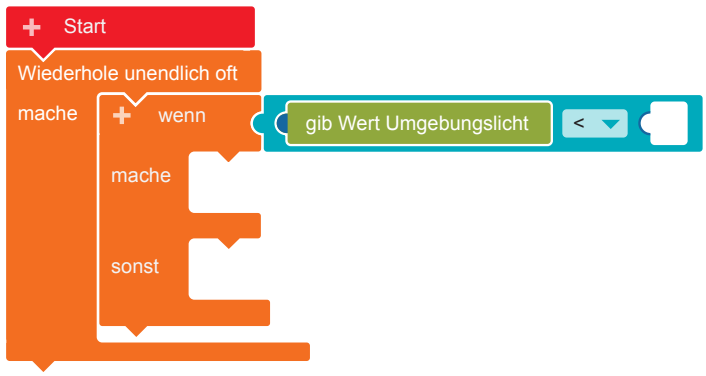
3 Der Schwellenwert muss geprüft werden, um zwischen „hell“ und „dunkel“ zu unterscheiden:
Dazu benötigen Sie einen **Vergleich** zwischen zwei Zahlen.
Wählen Sie aus der Kategorie Logik den Block  .
Fügen Sie ihn als **Bedingung** an die **Verzweigung** an.



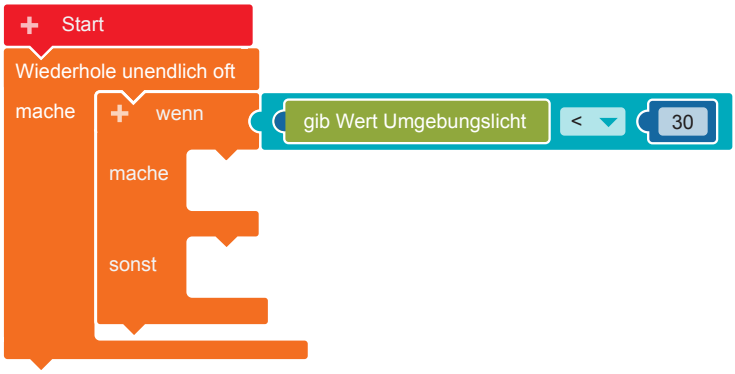
4 Das Licht soll eingeschaltet werden, sobald es dunkel genug ist:
Dazu muss folgende **Bedingung** erfüllt sein:
Der Messwert des Umgebungslichts unterschreitet einen vorgegebenen Schwellenwert. Dafür ist das Zeichen „=“ (ist gleich) ungeeignet und das Zeichen „<“ (ist kleiner als) muss gewählt werden.
Klicken Sie dazu auf das Zeichen „=“ und wählen Sie „<“ aus.



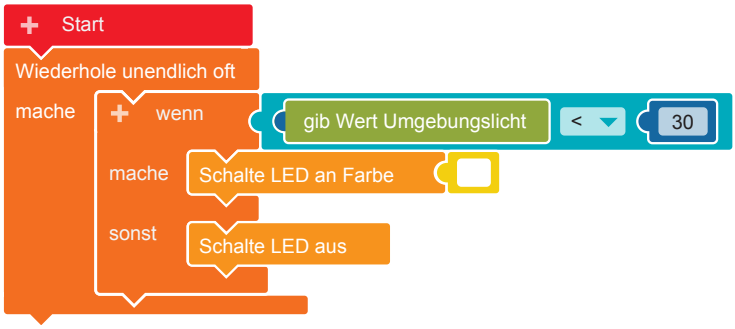
5 Die Stärke des Lichteinfalls muss vom *Calliope mini* gemessen und als Zahlenwert zum **Vergleichen** bereitgestellt werden:
Wählen Sie aus der Kategorie Sensoren den Block „gib Wert Umgebungslicht“ und fügen Sie ihn in die linke Lücke des „ist kleiner als“-Blocks ein.
Die Lichtstärke wird vom *Calliope mini* als Zahl zwischen 0 (kein Licht) und 255 (sehr hell) ermittelt.



6 Der Vergleichswert zur ermittelten Lichtstärke muss festgelegt werden:
Wählen Sie aus der Kategorie **Mathematik** den Block „0“ (null) und fügen Sie ihn in die rechte Lücke des „ist kleiner als“-Blocks ein. Klicken Sie dann auf das Feld mit der Null und tragen Sie eine Zahl zwischen 0 und 255 ein. Den gewünschten Wert können Sie durch Ausprobieren ermitteln. Ein guter Startwert ist 30. Damit ist der Schwellenwert festgelegt, der angibt, ab welcher Lichtstärke die Umgebung des *Calliope mini* für „dunkel“ gehalten wird.
Nun ist die **Bedingung** vollständig.
Sie trifft zu, wenn die Stärke des Umgebungslichts kleiner als der eingestellte Wert ist.



7 Die **RGB-LED** soll eingeschaltet werden, wenn die **Bedingung** zutrifft, d. h. es „dunkel“ ist:
Wählen Sie aus der Kategorie **Aktion** den Block „Schalte LED an Farbe“ und fügen Sie ihn an der mit „mache“ bezeichneten Lücke in die **Verzweigung** ein. Für den Fall, dass die Bedingung nicht zutrifft, es also „hell“ ist, soll die RGB-LED ausgeschaltet werden.
Wählen Sie aus der Kategorie **Aktion** den Block „Schalte LED aus“ und fügen Sie ihn an der mit „sonst“ bezeichneten Lücke in die Verzweigung ein.



8 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

Bei Tages- oder künstlichem Licht sollte die RGB-LED nach dem Start des Programms ausgeschaltet sein. Platzieren Sie den *Calliope mini* in den Schatten, etwa durch Abschatten mit den Händen, Ablegen in einem Schuhkarton mit Deckel, oder halten Sie ihn unter Ihren Arbeitstisch. Die RGB-LED sollte sich nun einschalten. Falls sie das nicht tut, erhöhen Sie den Schwellenwert ein wenig (siehe Schritt 6), damit die Umgebung früher als „dunkel“ eingestuft wird.

Anwendungsbezug

Lichtsensoren werden in Umgebungen genutzt, in denen es erforderlich ist, bei einer zu geringen natürlichen Beleuchtung zusätzliche Helligkeit zu schaffen. Dazu gehören bspw. das automatische Abblendlicht bei Fahrzeugen, ein automatischer Blitz in einer Kamera, die Regelung der Display-Helligkeit bei Smartphones oder sich abends ein- und morgens ausschaltende Straßenlaternen.

Erweiterungsmöglichkeiten

Statt nur die RGB-LED einzuschalten, können die Schülerinnen und Schüler auch selbst erstellte Bildfolgen auf dem LED-Bildschirm ablaufen lassen, wenn sich der *Calliope mini* im Dunklen befindet. Dazu wird zwischen den Bildern jeweils eine kurze Pause („Warte ms“-Block) eingefügt.

Der *Calliope mini* als Minipiano

Die Übung

Der *Calliope mini* wird zunächst für die Ausgabe einzelner Töne programmiert, die über den Tongenerator ausgegeben werden. Der über die Berührung der Hände mit den Touch-Pins hergestellte Kontakt zwischen den Schaltflächen ist der Auslöser der Tonausgabe.
Das erste Programm enthält bereits alle Grundstrukturen, die für die Vervollständigung zum Minipiano notwendig sind.

Fachbezug

Sachunterricht

Mit dem Minipiano können Schülerinnen und Schüler im Bereich *Technik* lernen, dass ihr Körper den elektrischen Strom leitet. Sie stellen mit ihren Fingern eine Verbindung zwischen zwei Kontaktflächen auf dem *Calliope mini* her. Leitende Verbindungen werden mit verschiedenen Tönen angezeigt. So erhält man ein Miniinstrument mit vier verschiedenen Tönen.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler erforschen im Bereich *Natur und Leben* ausgewählte Körperteile. Indem sie mit ihren Fingern eine Verbindung zwischen zwei *Calliope mini*-Kontaktflächen herstellen erfahren sie, dass ihr Körper den elektrischen Strom leitet.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler erkunden und verstehen im Bereich *Welt* Methoden der Welterkundung und Erkenntnisgewinnung (z. B. beobachten, experimentieren, planvoll umgehen) und wenden diese an.

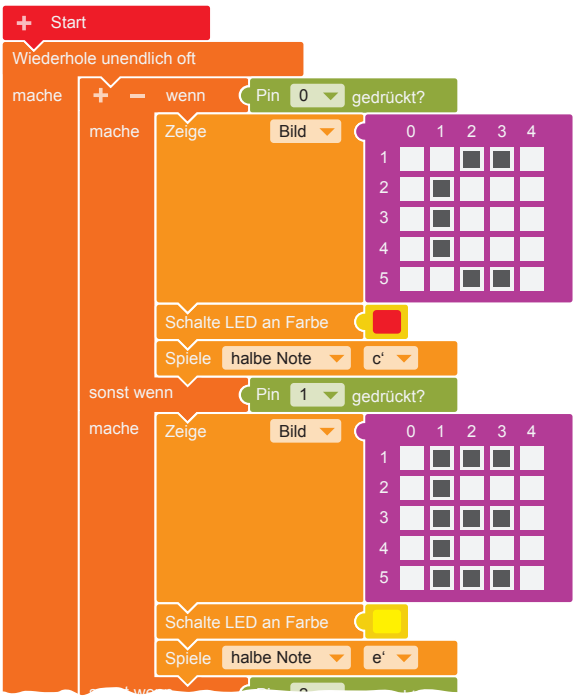
Anforderungen

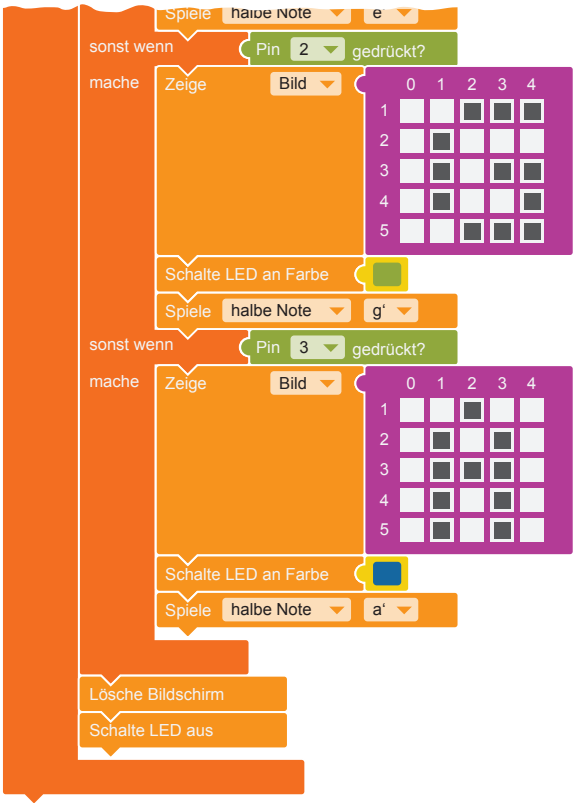
Programmierschwerpunkte:
- Eingabe über Touch-Pins, Ausgabe über Lautsprecher und LED-Bildschirm
- Strukturen: Endlosschleife, Verzweigung, Befehl
- verwendete *NEPO*®-Kategorien: Kontrolle, Aktion und Sensoren

Programmierschwierigkeit:
- Einstiegsniveau
- *NEPO*®-Level: Anfänger

Der Code

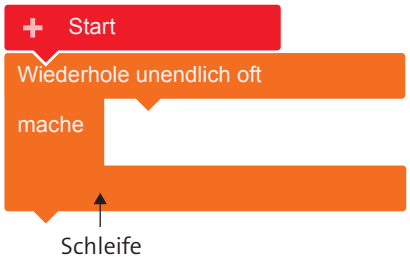
So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.



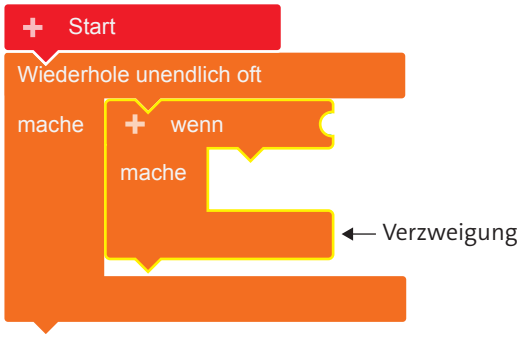


Schritte zur Erstellung des Programms für einen einzelnen Ton

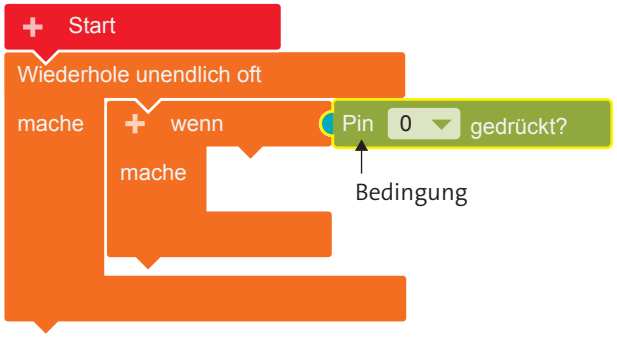
1 Damit der *Calliope mini* auf beliebig viele aufeinanderfolgende Touch-Pin-Berührungen reagieren kann, benötigen Sie eine Endlosschleife:
Wählen Sie aus der Kategorie **Kontrolle** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



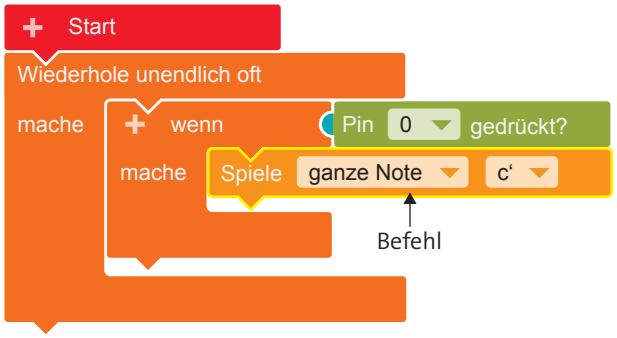
2 Der Calliope soll, falls ein Touch-Pin berührt wird, einen Ton erzeugen. Dazu benötigen Sie eine **Verzweigung**.
Wählen Sie der Kategorie **Kontrolle** den Block „wenn/mache“ und fügen Sie ihn in die Schleife ein.



3 Um zu entscheiden, ob ein Touch-Pin berührt wurde oder nicht, muss der *Calliope mini* dessen Zustand ermitteln und als **Wahrheitswert** bereitstellen:
Wählen Sie aus der Kategorie **Sensoren** den Block „Pin 0 gedrückt?“ und fügen ihn als **Bedingung** (blauer Bereich) an die Verzweigung an.



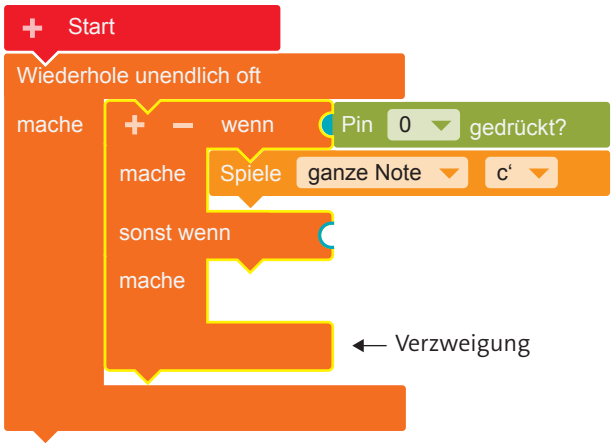
4 Ein Ton soll abgespielt werden, wenn die Bedingung zutrifft, d. h. der Touch-Pin berührt wird:
Wählen Sie aus der Kategorie **Aktion** den Block „Spiele ganze Note c‘“ und fügen Sie ihn in die mit „mache“ bezeichnete Lücke der Verzweigung ein.
Dieser **Befehl** lässt den *Calliope mini* den angegebenen Ton erzeugen.



5 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

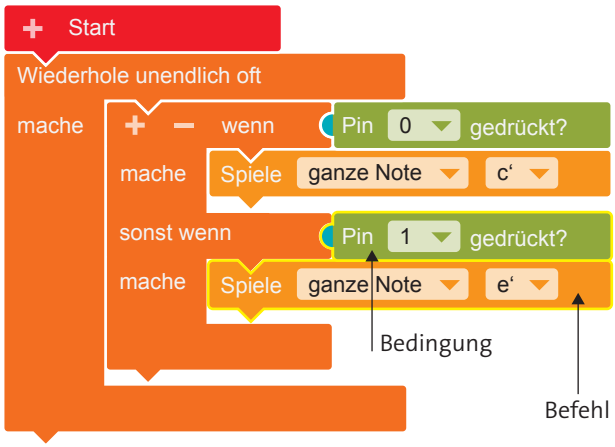
Schritte zur Erstellung des Programms für mehrere Töne

1 Erweitern Sie das Programm um die Abfrage der anderen **Touch-Pins**, damit weitere Töne gespielt werden können:
Klicken Sie dazu auf das „+“ neben dem „wenn“.
Eine weitere **Verzweigung** erscheint. Die neue Verzweigung wird ausgeführt, falls Pin 0 *nicht* mit dem Minus-Pin verbunden ist. Damit bekommt der *Calliope mini* die Möglichkeit, einen weiteren Touch-Pin zu überprüfen.



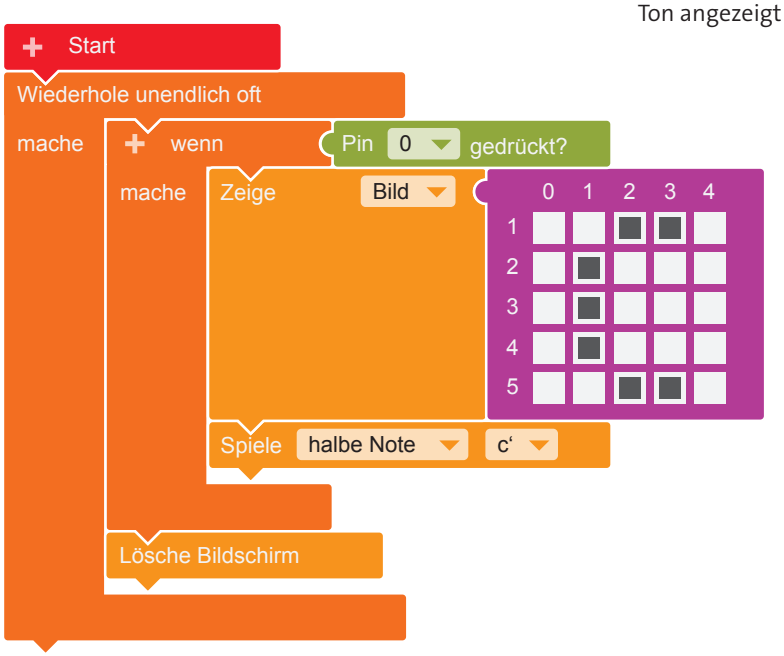
2 Stellen Sie ein, welchen Ton der *Calliope mini* spielen soll, wenn der Touch-Pin 1 berührt wird:
Fügen Sie aus der Kategorie Sensoren die **Bedingung** zur Überprüfung des Touch-Pin 1 ein. Fügen Sie aus der Kategorie Aktion dann einen weiteren **Befehl** zur Tonausgabe ein und wählen Sie den gewünschten Ton. In der Abbildung rechts wurde ein Beispiel gewählt.

3 Erweitern Sie das Programm so, dass alle vier Touch-Pins überprüft und vier verschiedene Töne ausgegeben werden können.



4 Erweitern Sie das Programm so, dass der Name des gerade gespielten Tons auf dem **LED-Bildschirm** angezeigt wird. In der Abbildung unten sehen Sie ein Beispiel für einen einzelnen Ton.

Wählen Sie aus der Kategorie Aktion den Block „Zeige Bild“ und fügen Sie ihn in die mit „mache“ bezeichnete Lücke der **Verzweigung** ein. Wählen Sie im 5 x 5-Raster die Felder so per Klick aus, dass der gerade gespielte Ton angezeigt wird.



5 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

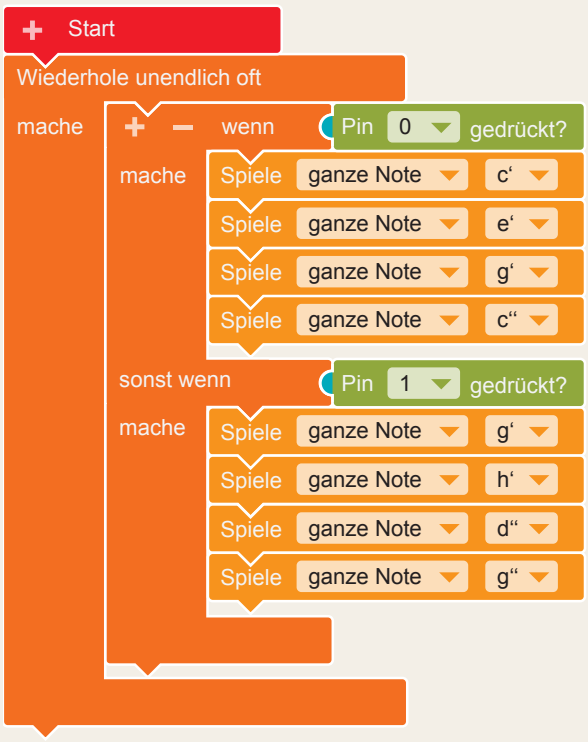
Beim Verbinden der Touch-Pins muss ein Finger auf dem Minus-Pin ruhen. Halten Sie dazu den *Calliope mini* in der linken Hand und umschließen Sie mit dem Daumen und Zeigefinger den Minus-Pin. Berühren Sie mit einem Finger der rechten Hand die Pins 0 bis 3, um die Tonausgabe zu starten.

Anwendungsbezug

Das Thema verbindet den Sachunterricht u. a. auf sehr anschauliche Weise mit dem Fach Musik, indem es eine grundlegende Funktionsweise elektrischer Instrumente aufzeigt. Keyboards und E-Pianos arbeiten nach demselben Grundprinzip wie das Programm: Ein Ton wird ausgegeben, wenn eine **Taste** gedrückt wird, d. h. der Stromkreis geschlossen ist.

Erweiterungsmöglichkeiten

Als Erweiterung des Programms können verschiedene Akkorde für jeden Pin programmiert werden. Dabei wird bei einem Kontakt nicht ein einzelner Ton, sondern eine kurze Tonfolge abgespielt. Beispielsweise kann ein C-Dur-Akkord bei Berührung des Pin 0 und ein G-Dur-Akkord bei Berührung des Pin 1 gespielt werden. So lässt sich ein Lied begleiten.



Der *Calliope mini* als Taktgeber

Die Übung

Der *Calliope mini* bietet die Möglichkeit, den Ablauf eines Programms zeitlich zu steuern. Die Genauigkeit liegt im Millisekundenbereich. Damit wird ein Taktgeber realisiert, der bspw. zur Darstellung unterschiedlicher Herzschlagfrequenzen genutzt werden kann.

Das Programm lässt den *Calliope mini* periodisch Ausgaben erzeugen, z. B. Lichtblitze oder Töne.

Fachbezug

Sachunterricht
Die Schülerinnen und Schüler lernen, die Zeitmessung im *Calliope mini* zu nutzen, um einen elektronischen Taktgeber selbst zu bauen. Dafür wird ein strikter zeitlicher Ablauf benötigt, der mittels der eingebauten Wartezeit-Blöcke realisiert wird.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Natur und Leben* ausgewählte Körperteile beschreiben, indem sie z. B. den Taktgeber mit ihrem Herzschlag abgleichen.

Prozessbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Welt erkunden und verstehen* Erfahrungen vergleichen, ordnen und auf unterschiedliche Kontexte beziehen (z. B. in Hinsicht auf Zeitgefühl und Zeitbewusstsein).

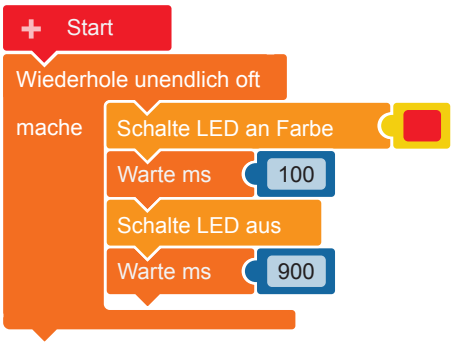
Anforderungen

Programmierschwerpunkte:
- Steuerung des Programmablaufs durch Wartezeiten
- Strukturen: Endlosschleife, Befehl
- verwendete *NEPO*®-Kategorien: Kontrolle und Aktion

Programmierschwierigkeit:
- Einstiegsniveau
- *NEPO*®-Level: Anfänger

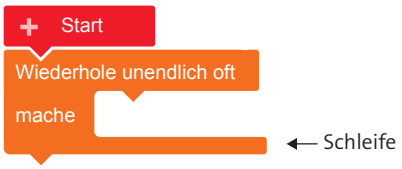
Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

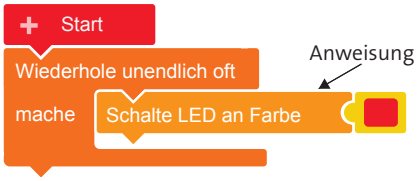


Schritte zur Erstellung des Programms Taktgeber

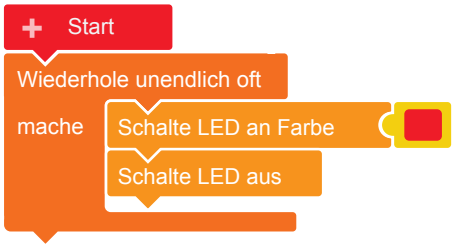
1. Die Taktimpulse sollen ununterbrochen ausgegeben werden:
Dazu benötigen Sie eine Endlos**schleife**.
Wählen Sie aus der Kategorie **Kontrolle** den **Block** „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



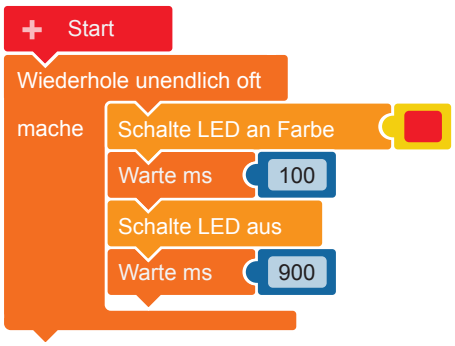
2. Die Lichtblitze entstehen durch das zeitgesteuerte Ein- und Ausschalten der **RGB-LED**:
Dazu benötigen Sie eine **Anweisung** zum Einschalten der RGB-LED.
Wählen Sie dazu aus der Kategorie **Aktion** den Block „Schalte LED an Farbe“ aus und fügen Sie ihn in die Schleife ein.



3. Die Anweisung zum Ausschalten der RGB-LED finden Sie ebenfalls in der Kategorie **Aktion**:
Fügen Sie den Block „Schalte LED aus“ in das Programm ein.



4. Damit das An- und Ausschalten sichtbar wird, müssen **Warte**zeiten bestimmt werden, die festlegen, wie lange die RGB-LED leuchtet bzw. ausgeschaltet ist:
Wählen Sie dazu aus der Kategorie **Kontrolle** den Block „Warte ms“, der das Programm für die angegebene Anzahl an Millisekunden anhält und anschließend wieder fortsetzt.
Die RGB-LED wird eingeschaltet und nach einer Zehntelsekunde (100 Millisekunden) wieder ausgeschaltet. Für den Rest (900 Millisekunden) bis zur vollen Sekunde (1000 Millisekunden) ist sie dunkel, d. h. sie blitzt einmal pro Sekunde auf, was einem Takt von 60 Schlägen pro Minute entspricht.
Ist ein schnellerer Takt gewünscht, so muss die Ausschaltzeit reduziert werden. Entsprechend verlangsamt sich der Takt, wenn die Ausschaltzeit verlängert wird.



5. Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Erweiterungsmöglichkeiten

Das Tempo des Taktgebers kann durch Tastendruck verändert werden.

(1) Mit jedem Betätigen der **Taste** A soll der Taktgeber 5 (ms) Millisekunden schneller werden: Die Ausschaltzeit der RGB-LED wird um 5 ms verkürzt. Mit jedem Betätigen der Taste B wird der Taktgeber hingegen um 5 ms langsamer. Für diese Änderungen der Taktfrequenz, deren aktueller Wert in der **Variablen** *wartezeit* abgelegt ist, wird die **Wartezeit** jeweils entsprechend eingegeben.

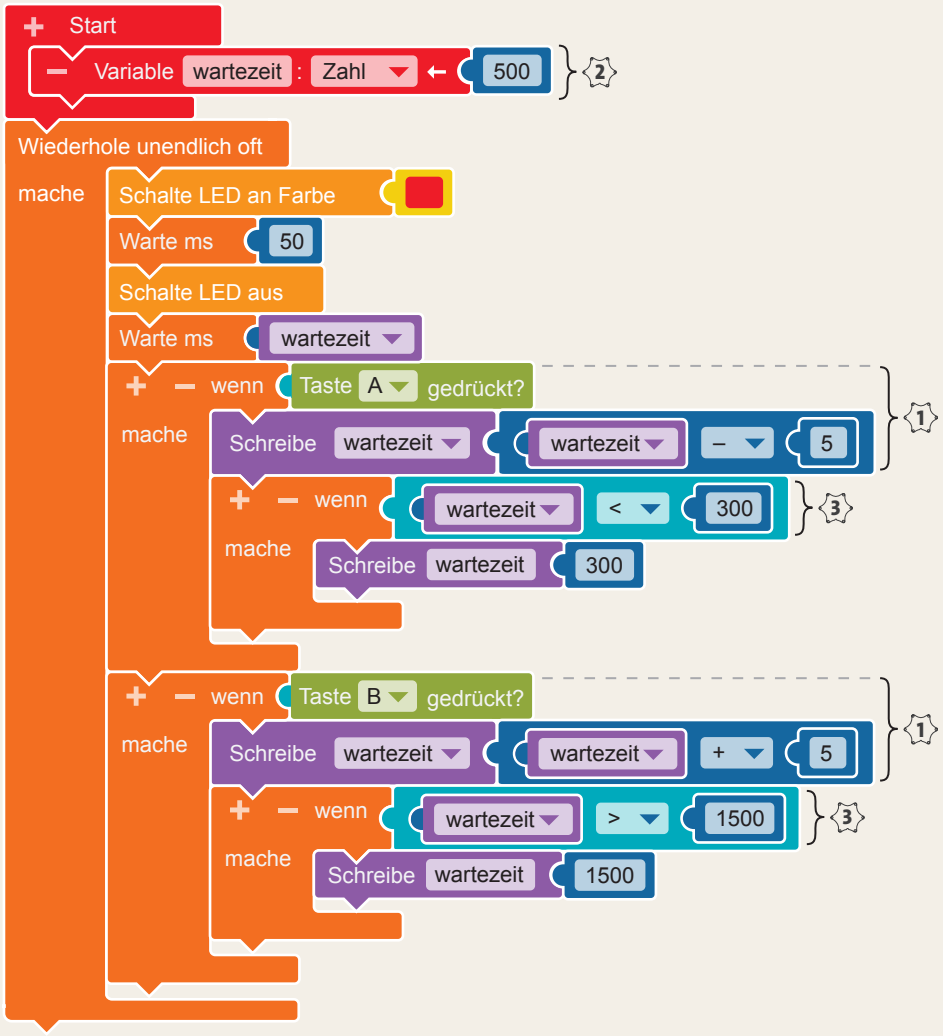
(2) Variable anlegen: Klicken Sie auf das „+“-Symbol neben dem „Start“-Block.

(3) Damit der Taktgeber nicht allzu schnell oder allzu langsam blitzen kann, wird die Wartezeit nach unten bei 300 ms und nach oben bei 1500 ms begrenzt.

Einsatzszenarien im Unterricht

Die Schülerinnen und Schüler vergleichen die Taktrate des *Calliope mini* mit ihrer Herzfrequenz, indem sie ihren Puls fühlen und gleichzeitig den *Calliope mini* beobachten. Sie versuchen durch Anpassen der Ausschaltzeit (untere Wartezeit im Programm) den vom *Calliope mini* erzeugten Takt ihrem Puls anzupassen.

Anwendungsbereiche: Taktgeber können als Zeitmessinstrumente verwendet werden, indem sie Zählwerke steuern. Außerdem nutzen Musiker diese Geräte zur Orientierung beim gemeinsamen Musizieren oder bei Studioaufnahmen. Blinklampen zur Warnung an Fahrzeugen oder Baustellen verwenden ebenfalls Taktgeber.



Der *Calliope mini* als Stoppuhr und Countdown-Zähler

Die Übung

Mit diesem Programm kann der *Calliope mini* als Stoppuhr verwendet werden. Zum Starten wird die Taste A einmal gedrückt. Zeitgleich mit dem Starten der Stoppuhr leuchtet die RGB-LED grün auf. Ein Druck auf die Taste B stoppt die Uhr und die gestoppte Zeit wird in Sekunden auf dem LED-Bildschirm angezeigt. Gleichzeitig wechselt die Farbe der RGB-LED von Grün auf Rot. Der Tastendruck kann jeweils durch einen kurzen Ton akustisch unterstützt werden.

Eine Variante der Stoppuhr ist das Rückwärtszählen einer Zeitspanne mit einem Countdown-Zähler (Wecker). Der Countdown startet durch einen Tastendruck auf „A". Zu jeder Sekunde wird auf dem LED-Bildschirm eine Animation gezeigt und die RGB-LED leuchtet grün. Nach einer zuvor im Programm festgelegten Zeit, z. B. einer Minute, schaltet die RGB-LED auf Rot um, ein Schlussbild erscheint und ein Weckton ertönt zehnmal hintereinander.

Fachbezug

Sachkunde

Die Schülerinnen und Schüler begegnen dem Phänomen Zeit in vielfältiger Weise. Sie können sich in überschaubaren Zeiträumen orientieren, unterscheiden Zeitbegriffe sowie zeitliche Strukturen und wenden sie an. Mit dem *Calliope mini* steht ein Instrument zur Zeitmessung zur Verfügung, das variabel eingesetzt werden kann.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Zeit und Wandel* lineare Zeitbegriffe und Instrumente (Uhr, ...) anwenden sowie Zeit als endliches und unendliches Phänomen erfassen und erlebte und gemessene Zeit in Bezug zueinander setzen, indem sie die Stoppuhr und den Countdown-Zähler als Zeitmesser und zur zeitlichen Kontrolle anwenden.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Welt erkunden und verstehen* die Zeit als endliches und unendliches Phänomen erfassen, erlebte und gemessene Zeit in Bezug zueinander setzen sowie entsprechende Erfahrungen vergleichen, ordnen und auf unterschiedliche Kontexte beziehen (z. B. Zeitgefühl und Zeitbewusstsein).

Anforderungen

Programmierschwerpunkte:

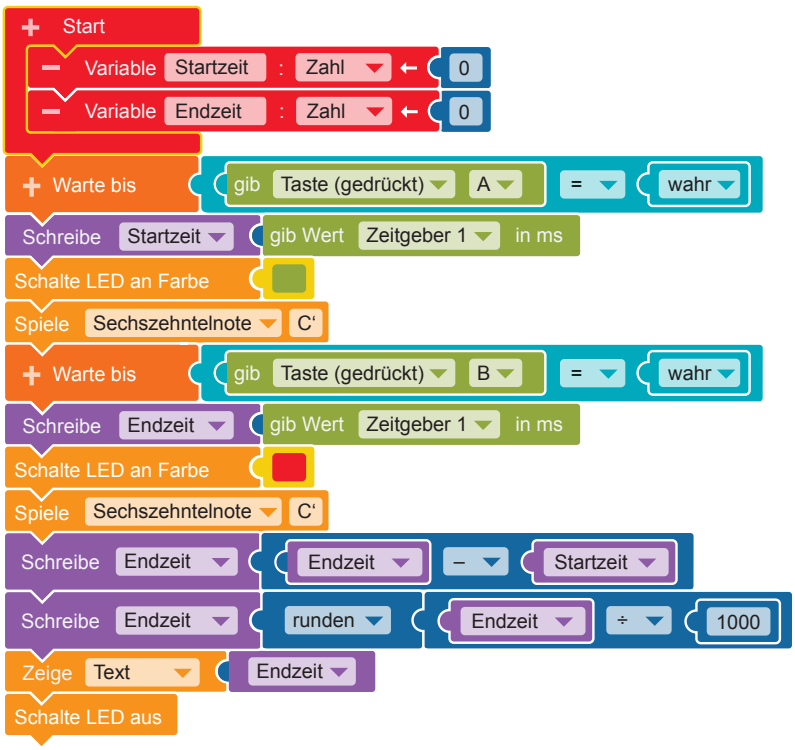
- Eingabe über Tasten, Ausgabe über LED-Bildschirm, RGB-LED und Lautsprecher
- Strukturen: bedingtes Warten, Zählschleife
- mit Variablen rechnen
- Zeitgeber abfragen
- Funktionen verwenden (Countdown-Zähler)
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle, Logik, Mathematik, Variablen, Funktionen

Programmierschwierigkeit:

- Einstiegsniveau
- *NEPO*®-Level: Anfänger/Experte

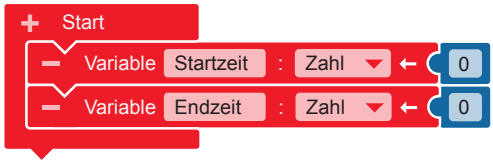
Der Code für die Stoppuhr

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

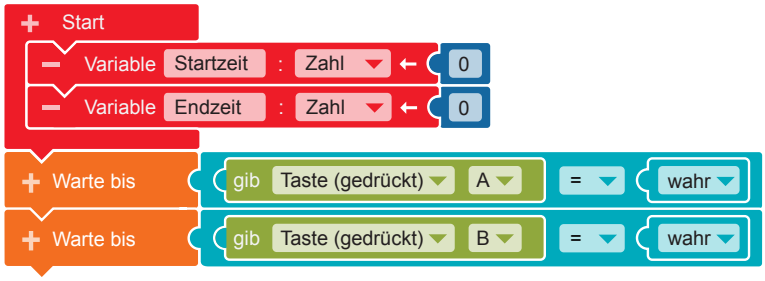


Schritte zur Erstellung des Programms für eine Stoppuhr

1. Damit die Startzeit und die Endzeit der Stoppuhr gespeichert werden können, müssen dafür zwei **Variablen** angelegt werden:
Um eine neue Variable anzulegen, klicken Sie auf das „+“ links neben „Start“.
Ein neuer Block erscheint. Legen Sie darin die Namen (*Startzeit, Endzeit*) und den **Datentyp** (Zahl) der Variablen fest. Beide Variablen haben zunächst den Wert Null.



2. Es werden zwei **Anweisungen** zur Tastensteuerung der Stoppuhr benötigt:
Wählen Sie aus der Kategorie Kontrolle den Block „Warte bis“ aus und fügen Sie ihn zweimal an.
Dieser Block beinhaltet bereits die Abfrage, ob die Taste A gedrückt wurde.
Ändern Sie im zweiten Block Taste „A“ auf Taste „B“.

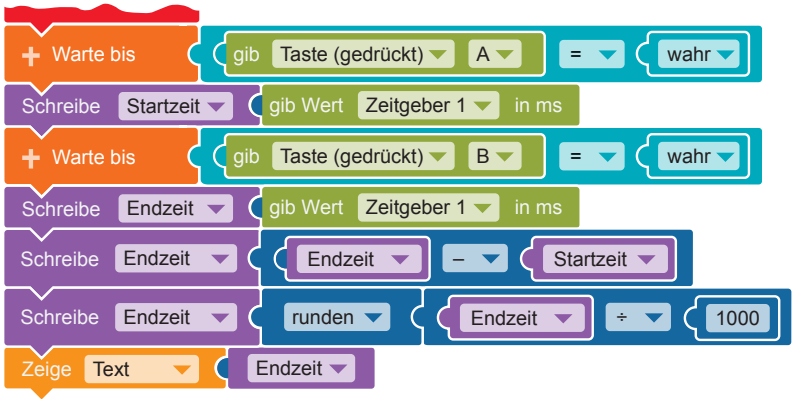


3 Im Anschluss an die Tastenabfragen werden zwei **Anweisungen** zum Speichern der Zeitwerte angefügt:
Wählen Sie aus der Kategorie Variablen den Block „Schreibe“ und fügen Sie ihn jeweils unter den **Warte**blöcken ein.

Wählen Sie dabei im ersten Block die Variable *Startzeit* und im zweiten die Variable *Endzeit* aus.
Aus der Kategorie Sensoren docken Sie den Block „gib Wert „Zeitgeber 1“ in ms“ jeweils an die neuen Blöcke an.

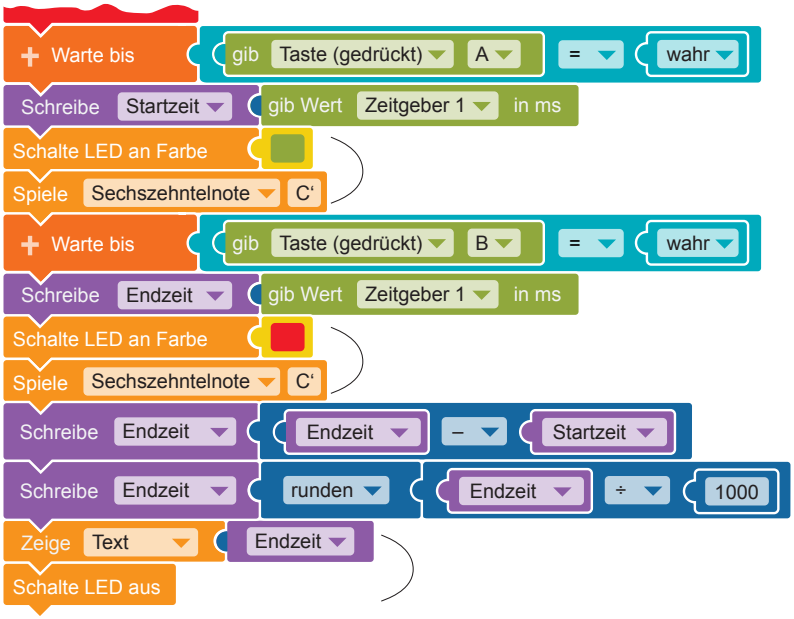
4 Jetzt kann die verstrichene Zeit in der Variablen „Schreibe *Endzeit*“ berechnet werden:
Wählen Sie aus der Kategorie Mathematik den arithmetischen Baustein, ändern diesen auf Subtraktion und setzen Sie die Variablen *Endzeit* und *Startzeit* ein.
Im zweiten Schritt werden durch Division durch 1000 und Rundung aus den Tausendstelsekunden ganze Sekunden berechnet (Schalten Sie für diesen Block in den Expertenmodus).

Die gestoppte Zeit wird auf dem LED-Bildschirm angezeigt:
Wählen Sie dazu aus der Kategorie Aktion den Block „Zeige Text“ und ersetzen Sie den Text „Hallo“ durch die Variable *Endzeit*.
Wählen Sie dazu aus der Kategorie Variablen den Block *Endzeit*.



5 Zusätzliche **Ausgaben** des *Calliope mini* ergänzen das Stoppuhr-Programm:
Wählen Sie aus der Kategorie Aktion die Blöcke „Schalte LED an Farbe“ und „Spiele Note“.
Am Ende wird die RGB-LED wieder ausgeschaltet.

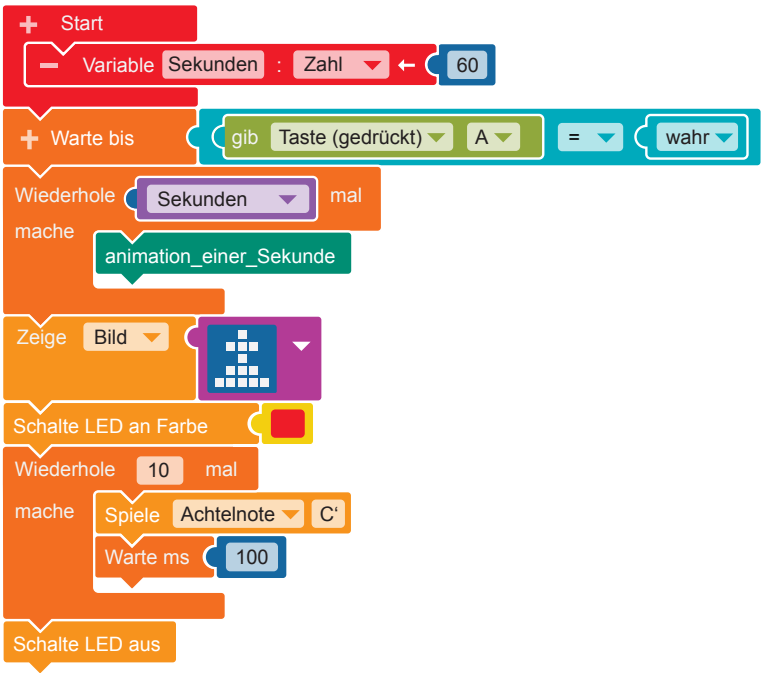
6 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).



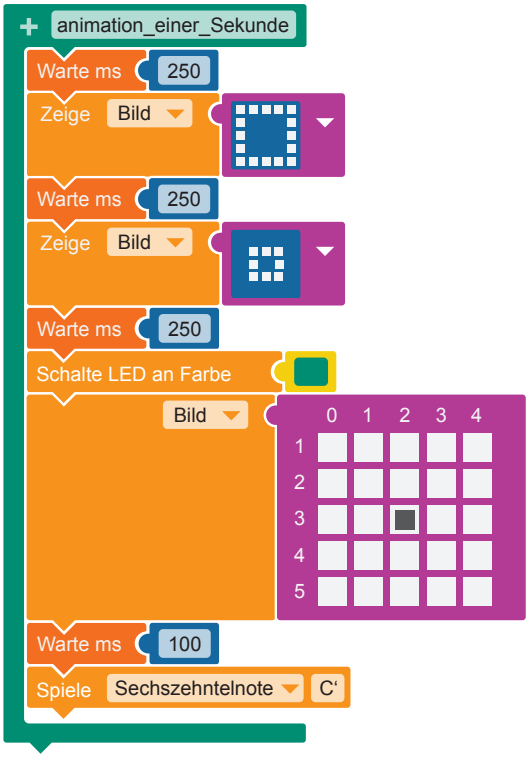
Der Code für den Countdown-Zähler:

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

Hauptprogramm



Funktion

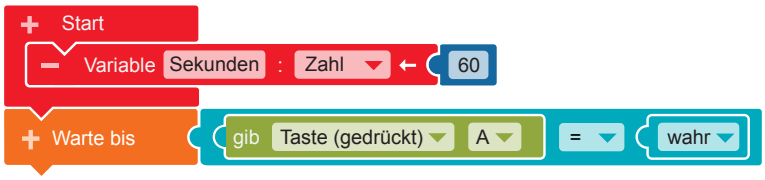


Schritte zur Erstellung des Programms für einen Countdown-Zähler (Wecker)

1. Als Erstes wird eine **Variable** für die zu speichernde Zeitspanne des Countdowns benötigt:

   Um eine neue Variable anzulegen, klicken Sie auf das „+“ links neben „Start“-Block. Ein neuer Block erscheint. Legen Sie darin den Namen (*Sekunden*) und den **Datentyp** (Zahl) der Variablen fest. Der gewählte Startwert „60“ gibt die Sekunden des Countdowns an.

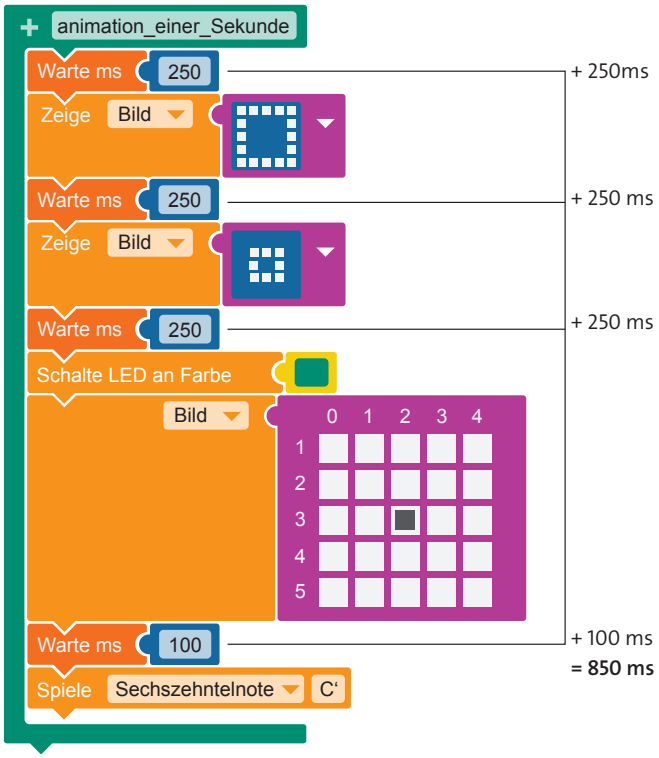
2. Für den Start des Countdowns während der Programmausführung muss der Druck auf die Taste A registriert werden:
   Wählen Sie aus der Kategorie **Kontrolle** den Block „Warte bis“ und fügen Sie ihn an den „Start“ Block. Dieser Block beinhaltet bereits die Abfrage, ob die Taste A gedrückt wurde.



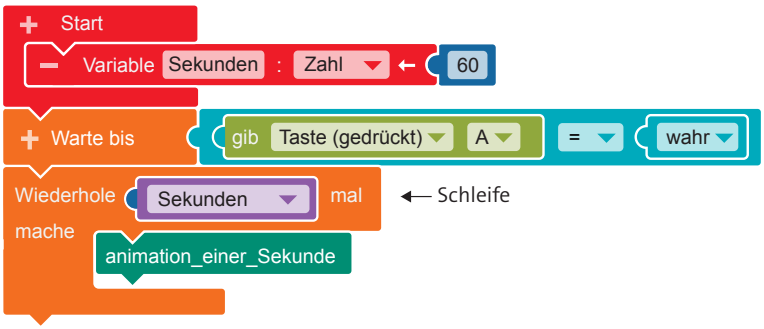
3 Die Animation, die innerhalb einer Sekunde abläuft, wird außerhalb des Hauptprogramms in einer **Funktion** programmiert:
Wählen Sie aus der Kategorie **Funktionen** den Block „mache Etwas“ und legen Sie diesen in einen freien Fensterbereich neben das Hauptprogramm und ändern den Namen in *animation_einer_Sekunde*.

Innerhalb der **Funktion** fügen Sie die **Aktions**blöcke für die Anzeige der Bilder auf dem LED-Bildschirm, sowie die Warteblöcke aus der Kategorie **Kontrolle** wie rechts abgebildet ein.

Die Summe der Wartezeit-Blöcke beträgt nur 850 Millisekunden. Die restliche Zeit bis zur vollen Sekunde wird durch die Animation verbraucht.

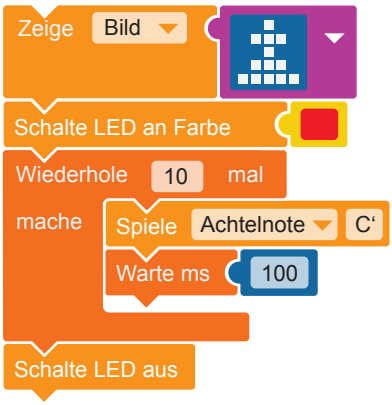


4 Die innerhalb der Funktion programmierte Sekundenanimation muss laut Startwert 60mal wiederholt werden:
Wähle Sie aus der Kategorie **Kontrolle** ▶ **Schleifen** den Block „Wiederhole 10 mal“ und ersetzen Sie den Wert 10 durch die **Variable** *Sekunden*. In diese Zähl**schleife** fügen Sie den vorher erstellten Block *animation_einer_Sekunde* aus der Kategorie **Funktionen** ein.



5 Nach Beendigung der Schleife ist der Countdown abgelaufen. Dies soll über folgende Ausgaben angezeigt werden:
Wähle Sie aus der Kategorie **Aktion** den Block „Zeige Bild“ mit einer geeigneten Darstellung und z. B. weitere **Befehle** für die RGB-LED oder eine Tonausgabe.

6 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).



Hinweise und Infos

Einsatzszenarien im Unterricht

- Zeitschätzungen, z. B. für die Dauer der Bearbeitung einer Aufgabe
- Zeitmessungen bei Experimenten
- Zeitmessung bei Wettbewerben
- Zusammenhang von Weg pro Zeit

Erweiterungsmöglichkeiten

- Die Stoppuhr kann so umprogrammiert werden, dass in kleineren Zeiteinheiten gemessen werden kann. Für Zehntelsekunden wird der Teiler von 1000 auf 100 geändert.
- Zusätzliche Ausgabe von Tönen oder Lichtsignalen für jede Minute / Sekunde (z. B. für Zeitmessungen bei Spielen / im Sport).

Morsen mit dem *Calliope mini*

Die Übung

Der *Calliope mini* wird in dieser Übung so programmiert, dass er als Morseapparat nutzbar ist. Zur Darstellung des Morsealphabets sind lediglich die Ausgabe eines Punktes und die einer waagerechten Linie auf dem LED-Bildschirm erforderlich. Durch eine zusätzliche Programmierung unterschiedlich langer Töne werden die Morsezeichen auditiv unterstützt und der selbst codierte Mini-Morseapparat funktioniert auch, wenn Sender und Empfänger sich nicht sehen können.

Fachbezug

Deutsch

Die Schülerinnen und Schüler setzen sich mit fremden Schriften und Symbolsystemen auseinander. Sie lernen das Morsealphabet kennen, besprechen, wie sich dieses darstellen lässt und welche Vorteile eine Programmierung der Morsezeichen auf dem *Calliope mini* bietet (z. B. gleichzeitige visuelle und auditive Ausgabe, Funktionalität auch im Dunkeln etc.).

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler können sich im Bereich *Sprache und Sprachgebrauch untersuchen* zu Sachverhalten strukturiert äußern auch unter Verwendung digitaler Kommunikationsmedien, indem sie sich mit fremden Schriften und Symbolsystemen auseinandersetzen.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Sprechen und Zuhören* Medien als ein Mittel der Alltagskommunikation einsetzen.

Anforderungen

Programmierschwerpunkte:

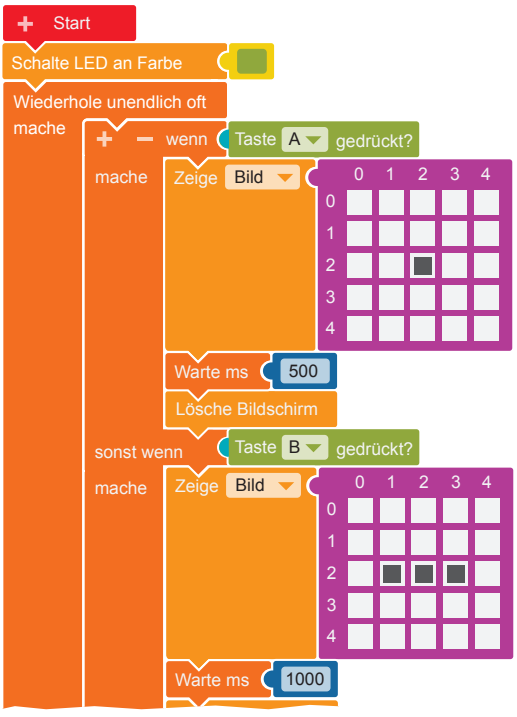
- Eingabe über Tasten und Touch-Pins, Ausgabe über LED-Bildschirm, Lautsprecher und RGB-LED
- Strukturen: bedingte Anweisung, Endlosschleife, Bedingungen formulieren
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle

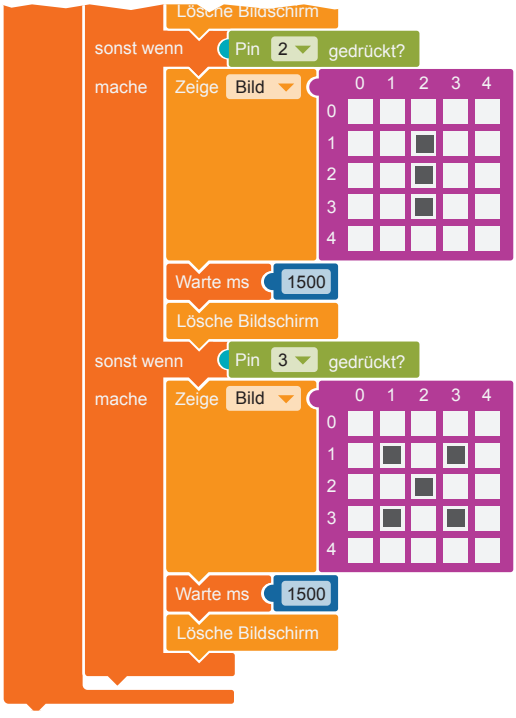
Programmierschwierigkeit:

- Einstiegsniveau
- *NEPO*®-Level: Anfänger

Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

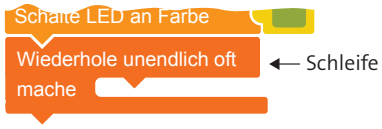




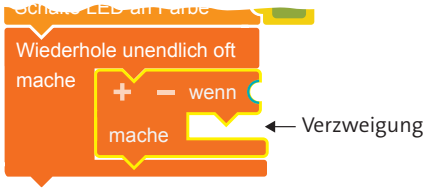
Schritte zur Erstellung des Programms zur Ausgabe von Morsezeichen

1. Damit der Nutzer erkennen kann, dass der Morseapparat in Betrieb ist, wird die **RGB-LED** direkt beim Programmstart angeschaltet:
Wählen Sie aus der Kategorie **Aktion** den Block „Schalte LED an Farbe“ und fügen Sie ihn an den „Start“-Block an.

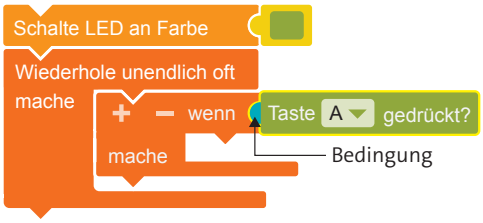
2. Damit unendlich oft Morsezeichen angezeigt werden können, wird eine Endlosschleife benötigt:
Wählen Sie aus der Kategorie **Kontrolle** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an.



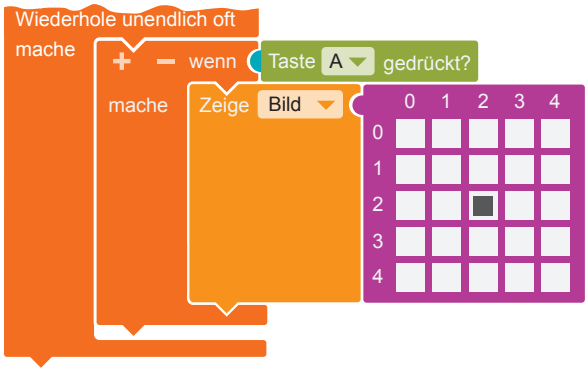
3. Wenn die **Taste A** gedrückt wird (*wenn*), soll ein Morse-Punkt angezeigt werden (*mache*):
Um diese Bedingung aufzustellen, benötigen Sie eine **Verzweigung**.
Wählen Sie aus der Kategorie **Kontrolle** den Block „wenn/mache“ und setzen Sie ihn in die Schleife ein.



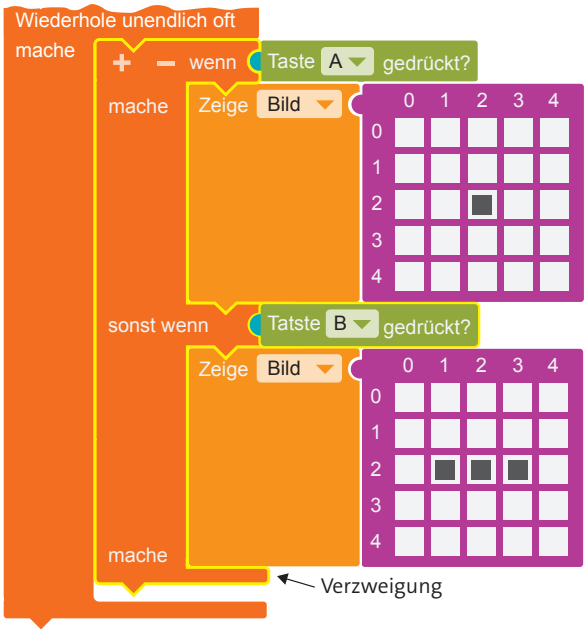
4. Die gewünschte Eingabemöglichkeit (hier über den Sensor „Taste A“) wird ausgewählt:
Nehmen Sie aus der Kategorie **Sensoren** den Block „Taste A gedrückt?“ und hängen Sei ihn als **Bedingung** (blauer Bereich) an die Verzweigung an.



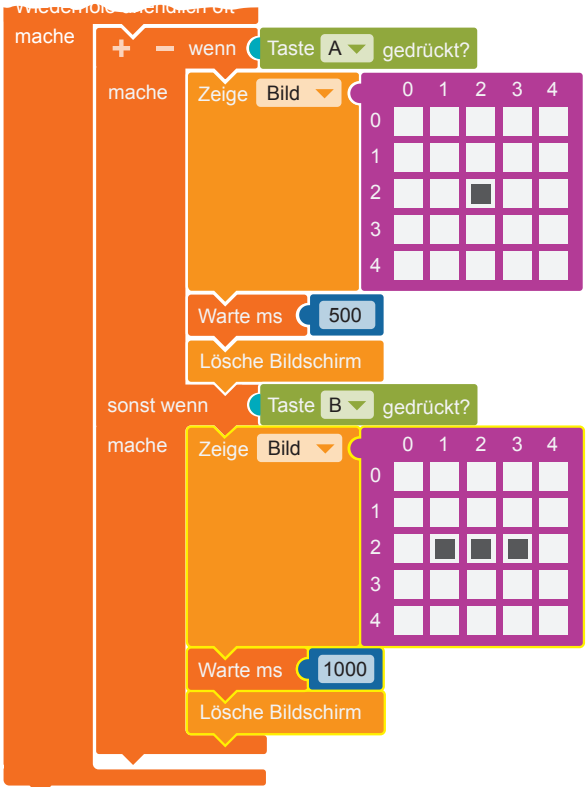
5. Der Morse-Punkt wird auf dem LED-Bildschirm angezeigt:
Wählen Sie aus der Kategorie **Aktion** den Block „Zeige Bild“ und markieren Sie hier zur Darstellung des Morse-Punktes das Kästchen in der Mitte des 5 x 5-Rasters.



6. Wenn die Taste B gedrückt wird (*wenn*), soll ein Morse-Strich auf dem LED-Bildschirm angezeigt werden (*mache*):
Für diese Bedingung benötigen Sie eine weitere **Verzweigung**.
Klicken Sie hierfür in der Verzweigung auf das „+“ neben dem „wenn“.
Wählen Sie aus der Kategorie **Sensoren** den Block „Taste B gedrückt?“ und fügen Sie ihn als **Bedingung** (blauer Bereich) in die neue Verzweigung ein.
Wählen Sie aus der Kategorie **Aktion** den Block „Zeige Bild“ und markieren Sie zur Darstellung des Morse-Striches drei Punkte in der Waagerechten des 5 x 5-Rasters.



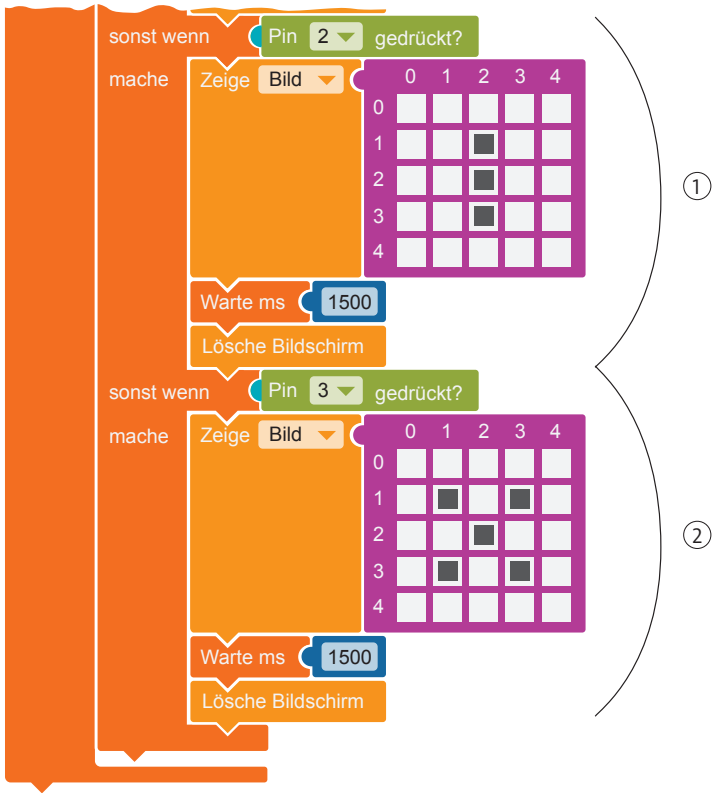
7 Damit die Morsezeichen gut lesbar sind, müssen sie voneinander abgegrenzt werden. Hierfür wird hinter jeder Ausgabe eine **Pause** eingefügt und anschließend der Bildschirminhalt gelöscht:
Wählen Sie aus der Kategorie Kontrolle den Block „Warte ms“, und legen Sie hier die Länge der Pause fest (1000 ms = 1 Sekunde).
Löschen Sie nun den Bildschirm mithilfe des Blocks „Lösche Bildschirm“ aus der Kategorie Aktion.



8 Damit der Empfänger Buchstaben und Wörter deutlich voneinander abgrenzen kann, ist eine Erweiterung des Codes durch weitere Verzweigung nötig:

(1) Abgrenzung Buchstabe:
Wählen Sie hierzu aus der Kategorie Sensoren den Block „Pin 2 gedrückt?“ und fügen Sie ihn an eine weitere Verzweigung an. Erzeugen Sie mit dem Block „Zeige Bild“ aus der Kategorie Aktion eine senkrechte Linie zur Abgrenzung einzelner Buchstaben.

(2) Abgrenzung Wort:
Erzeugen Sie eine weitere Verzweigung und ergänzen Sie aus der Kategorie Sensoren den Block „Pin 3 gedrückt?“ Mit dem Block „Zeige Bild“ aus der Kategorie Aktion erstellen Sie ein Kreuz zur Abgrenzung einzelner Wörter.



9 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

Dieser Code eignet sich gut für die Partnerarbeit. Als Erweiterung können die Schülerinnen und Schüler ein automatisches Morse-Wort ihrer Wahl programmieren.

Hintergrundwissen

Der Künstler und Erfinder Samuel Morse entwickelte ab 1837 den ersten Telegrafen, wodurch eine schnelle Kommunikation über sehr weite Entfernungen möglich wurde. Der Telegraf konnte, anstelle von Wörtern, nur elektrische Impulse versenden. Festgelegte Kombinationen kurzer und langer Stromstöße repräsentieren die jeweiligen Buchstaben.

Fächerverbindende Hinweise

Das Thema *Morsen* lässt sich gut im Lehrplan des Sachunterrichts verorten:
Die Schülerinnen und Schüler können verschiedene Arten und Methoden der Kommunikation nutzen, beschreiben und reflektieren ausgewählte Erfindungen, deren Entwicklung und die Auswirkung auf die Lebenswelt, auch mit Blick auf die Zukunft.

Erweiterungsmöglichkeiten

Unterstützende Tonausgabe:
Als Erweiterung des Programms können die Morsezeichen durch eine Tonausgabe auditiv unterstützt werden. Wählen Sie hierzu aus der Kategorie Aktion den Block „Spiele ganze Note C“ und fügen Sie ihn jeweils unter den Block „Zeige Bilder“ ein. Wählen Sie eine passende Tonhöhe und -länge.

Hinweis: Der Block „Warte bis“ ist nun nicht mehr notwendig, da die Tonlänge auch die Länge der Darstellung des Morsezeichens auf dem LED-Bildschirm bestimmt.

Morsezeichen mit dem *Calliope mini* funken:
Die Kategorie Nachrichten bietet die Möglichkeit, mehrere Calliope-mini-Mikrocontroller untereinander kommunizieren zu lassen. So kann ein Morsecode von *Calliope mini* zu *Calliope mini* oder gleichzeitig an verschiedenen Platinen gesendet werden.

Bildimpulse und Reizwörter mit dem *Calliope mini* erzeugen

Die Übung

Der *Calliope mini* wird als Ideen-Generator programmiert, der automatisch Anregungen für einen Schreibanlass erzeugen soll. Dazu wird auf Knopfdruck eines von drei oder mehr Bildern zufällig ausgewählt und ausgegeben. Außerdem erscheint kurz darauf die Beschreibung des Bildes als einzelnes Wort. Die im *Calliope mini* verfügbaren Bilder und Wörter können selbst gestaltet bzw. frei gewählt werden.

Fachbezug

Deutsch

Die Schülerinnen und Schüler entwickeln auf Grundlage eines auf dem 5x5-LED-Bildschirm dargestellten Bildes Ideen für eine kleine Geschichte und planen diese. Gleichzeitig kann über die Eindeutigkeit und Bedeutung von Piktogrammen in unterschiedlichen Darstellungen diskutiert werden.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schüler können im Bereich *mit Texten und anderen Medien umgehen* nach Anregungen eigene Texte planen und schreiben, indem sie das dargebotene Bild/Wort als Textanlass verwenden und eine Reizwortgeschichte schreiben.

Prozessbezogene Kompetenzen

Die Schüler können im Bereich *Schreiben* eine Schreibidee entwickeln, planen und aufschreiben und auf die logische Reihenfolge achten sowie je nach Schreibanlass adressaten- und funktionsgerecht schreiben.

Anforderungen

Programmierschwerpunkte:

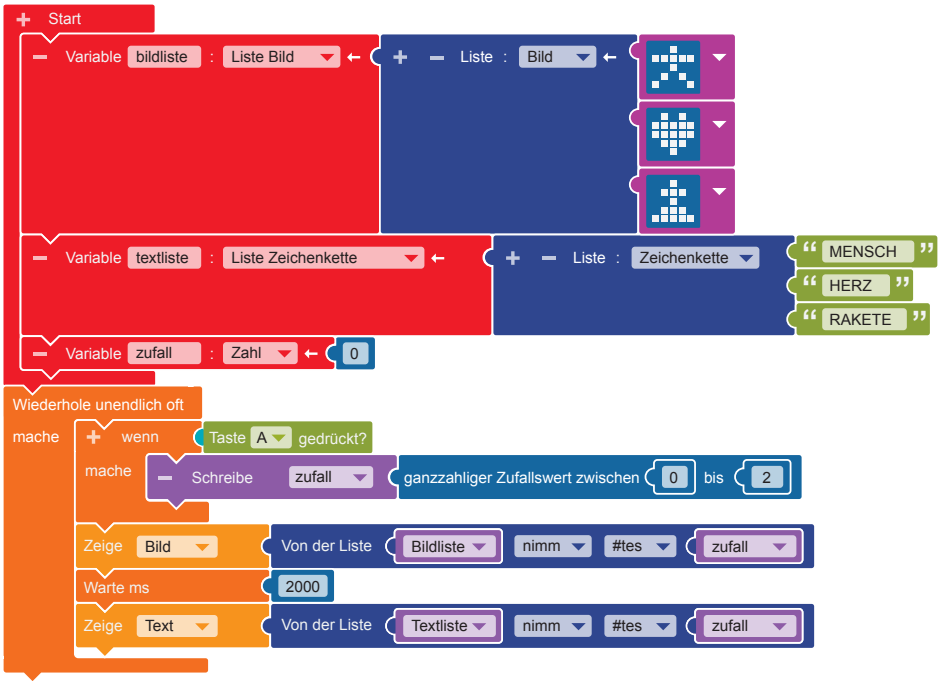
- Eingabe über Tasten, Ausgabe über LED-Bildschirm
- zufällige Auswahl eines Elements aus einer Liste
- Strukturen: Endlosschleife, Variable, Liste, Zufall, Verzweigung, Befehlverwendete *NEPO*®-Kategorien: Kontrolle, Aktion, Liste, Variablen, Mathematik, Bilder, Sensoren

Programmierschwierigkeit:

- fortgeschrittenes Niveau
- *NEPO*®-Level: Experte

Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

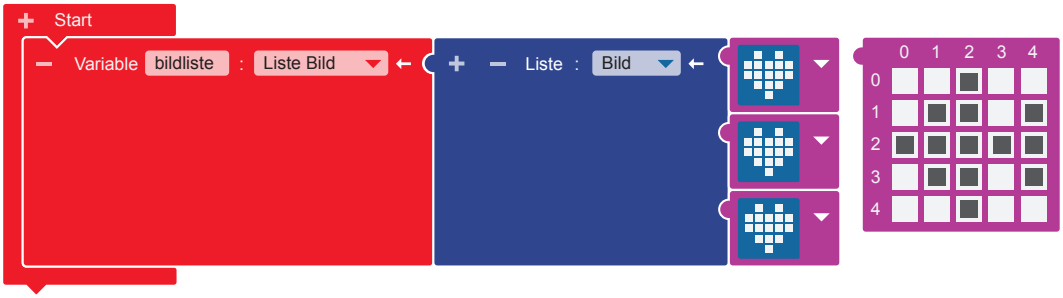


Schritte zur Erstellung des Programms für die Bildimpulse und Reizwörter

1 Die zur Auswahl stehenden Bilder und Wörter sollen in **Listen** gespeichert werden:
Erstellen Sie zunächst eine Liste, in der die verfügbaren Bilder enthalten sind. Erzeugen Sie dazu eine neue **Variable**, indem Sie auf das „+“ neben „Start“ klicken. Ändern Sie den Namen der Variablen in *bildliste* und wählen Sie im **Ausklappmenü** „Liste Bild“ als **Datentyp**.

2 Es wird mit den Bildern begonnen:
Wählen Sie die Bilder aus, die später vom *Calliope mini* auf dem **LED-Bildschirm** angezeigt werden sollen.

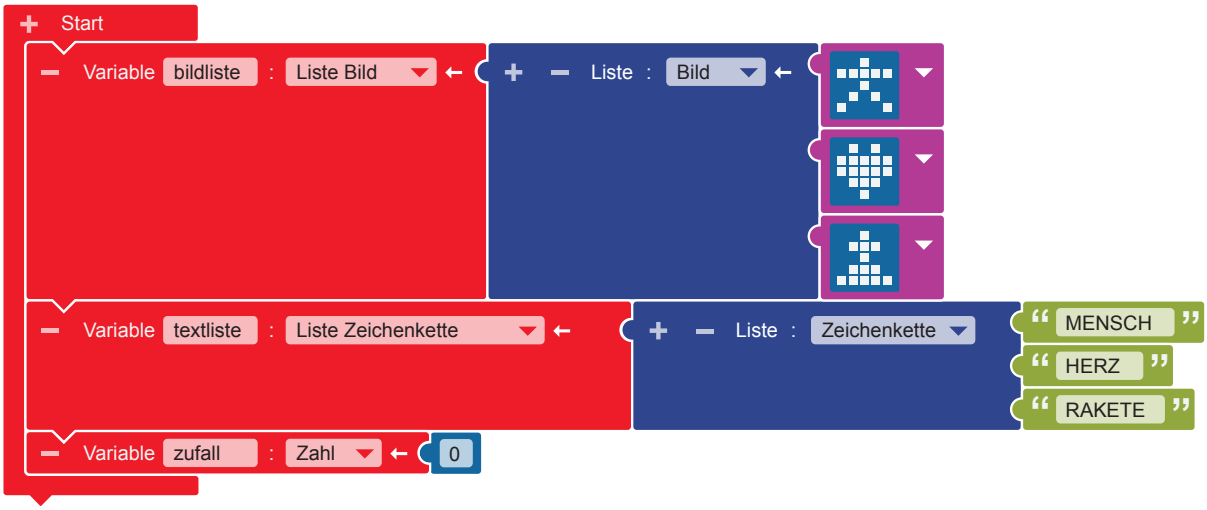
Sie können mit den in *NEPO®* vorgegebenen Bildern beginnen oder eigene Bilder erstellen. Die benötigten Blöcke finden Sie in der Kategorie Bilder.



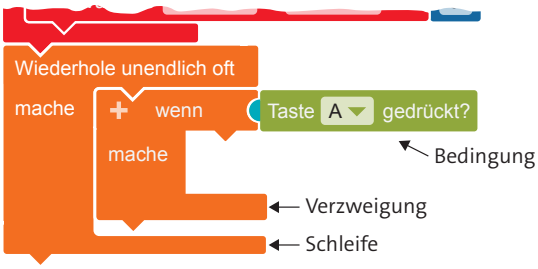
3 Die Wörter, die zu den Bildern angezeigt werden sollen, müssen ebenfalls in einer Liste abgelegt werden:
Erstellen Sie zwei weitere Variablen. Nennen Sie die zweite Variable *textliste*. Wählen Sie „Liste **Zeichenkette**“ als Datentyp. Danach können Sie die zu den Bildern passenden Wörter in die grünen Textblöcke eingeben. Achten Sie darauf, dass die Wörter in derselben Reihenfolge wie die Bilder angeordnet sind. Im Beispiel steht das Strichmännchen an erster **Stelle** in der Bildliste sowie das Wort „MENSCH“ an erster Stelle in der Textliste.

Die Bilder und Texte sollen vom *Calliope mini* zufällig ausgewählt werden:

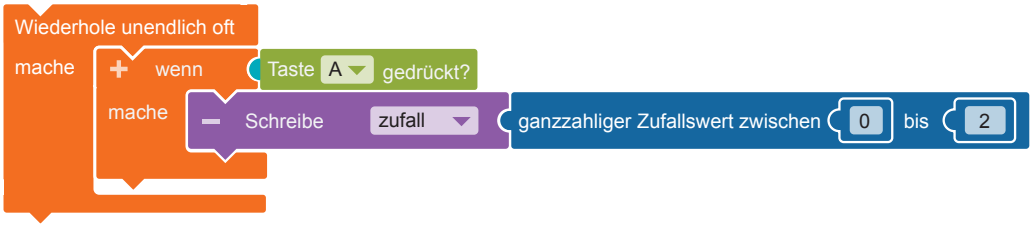
Erstellen Sie eine dritte Variable, die zur Aufbewahrung der erzeugten **Zufall**szahl dient. Nennen Sie sie diese *zufall*.



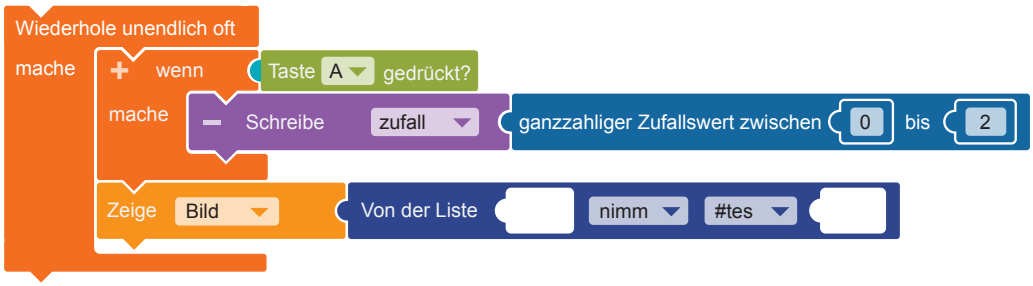
4 Nach dem Drücken der Taste A sollen ein Bild und der dazugehörige Text angezeigt werden:
Dazu wird im Programm zunächst eine Endlos**schleife** eingefügt. Fügen Sie in die Endlosschleife eine **Verzweigung** ein, mit der geprüft wird, ob der Benutzer gerade die Taste A drückt.
Wählen Sie aus den Kategorien Kontrolle und Sensoren die entsprechenden Blöcke aus.



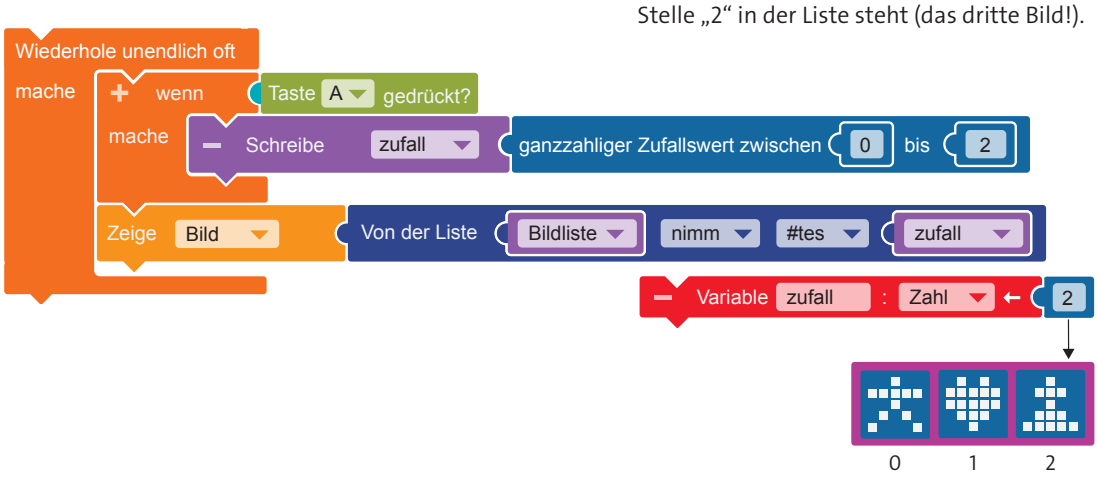
5 Das anzuzeigende Bild und der dazugehörige Text sollen zufällig ausgewählt werden:
Wählen Sie aus der Kategorie Variablen den Block „Schreibe *zufall*“ und fügen Sie ihn in die Verzweigung ein.
Daran hängen Sie den Block „ganzzahliger Zufallswert zwischen“ aus der Kategorie Mathematik. Ändern Sie die Grenzen für den Zufallswert auf 0 für die untere und auf 2 für die obere Grenze.
Durch diese **Befehle** erzeugt der *Calliope mini* eine zufällige Zahl (0, 1, 2).
Diese erzeugte Zahl wird in die Variable *zufall* geschrieben. Das bedeutet, dass die Variable *zufall* den Wert aufbewahrt, der in sie hineingeschrieben wurde.



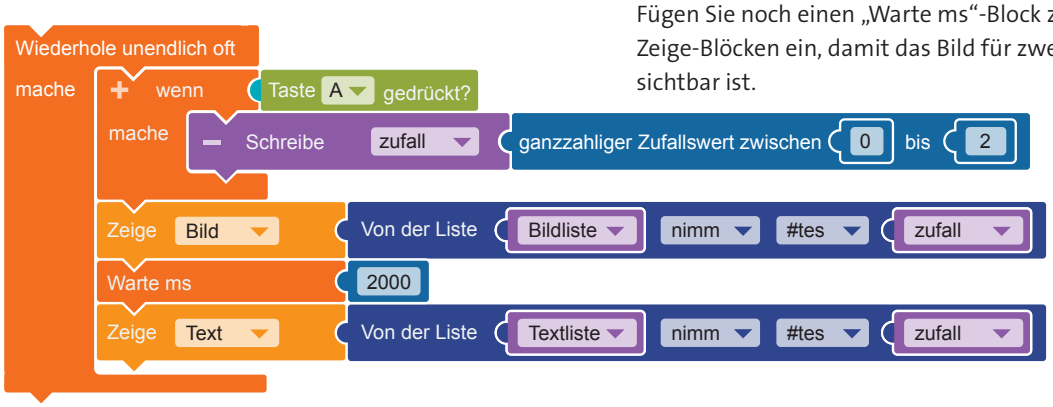
6 Es soll ein Bild angezeigt werden, das an der Stelle in der Liste steht, die durch den Wert der Variablen *zufall* bezeichnet wird:
Zum Anzeigen eines Bildes dient der Block „Zeige Bild“ aus der Kategorie Aktion.
Fügen Sie ihn unter der Verzweigung ein.
Wählen Sie anschließend aus der Kategorie Liste den Block „von der Liste nimm #tes“. Er dient dazu, ein Element einer Liste über dessen Nummer (**Index**) auszuwählen.
Das erste Bild in der Liste hat die Nummer (den Index) 0, das zweite die 1 etc .



7 Bei der Auswahl der Listenelemente muss festgelegt werden, aus welcher Liste sie stammen:
Fügen Sie in die Lücke nach „von der Liste“ die Variable namens *bildliste* aus der Kategorie **Variablen** ein. Ebenfalls aus der Kategorie **Variablen** nehmen Sie die Variable *zufall* und fügen diese in die Lücke nach „#tes“ ein. Hierdurch wird das Bild ausgewählt, das an der Stelle in der Liste steht, deren Nummer dem Wert der Variablen *zufall* entspricht. Hat die Variable bspw. den Wert „2“, so wird das Bild ausgewählt, das an der Stelle „2“ in der Liste steht (das dritte Bild!).



8 Ein zum angezeigten Bild passendes Wort soll ebenfalls ausgegeben werden:
Wählen Sie aus der Kategorie **Aktion** den Block „Zeige Text“ und fügen Sie ihn unter dem Block „Zeige Bild“ ein. Erstellen Sie eine Kopie der Listenauswahl. Ändern Sie die kopierte Zeile so, dass *textliste* im kopierten Block steht. Klicken Sie dazu auf das Wort *bildliste* und wählen Sie im Ausklappmenü das Wort *textliste*. Fügen Sie den Block „von der Liste“ an den „Zeige Text“-Block an. Beachten Sie, dass dies erst funktioniert, wenn Sie das Wort *textliste* ausgewählt haben. Die neue Zeile zeigt einen Text aus der Textliste an, der durch den Wert der Variablen *zufall* ausgewählt wird. Um denselben Wert hier erneut abrufen zu können, muss er vorher in der Variablen zwischengespeichert werden. Wäre der bei der Textauswahl verwendete Auswahlwert ein anderer, würden Text und Bild nicht mehr zusammenpassen.
Fügen Sie noch einen „Warte ms“-Block zwischen den Zeige-Blöcken ein, damit das Bild für zwei Sekunden sichtbar ist.



9 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

Die Schülerinnen und Schüler wählen zunächst Bilder aus *NEPO*® aus und versehen sie mit entsprechenden Wörtern. Dann begeben sie sich in eine Partnerarbeit, tauschen ihre Platinen aus und beginnen, zu den Begriffen ihres Partners eine Geschichte zu schreiben. Durch die zufällige Auswahl ist nicht vorhersehbar, in welcher Reihenfolge die Begriffe erscheinen. Im weiteren Unterrichtsverlauf können sie eigene Bilder entwerfen und sie mit Begriffen versehen.

Erweiterungsmöglichkeiten

Das Programm kann so erweitert werden, dass außer den Bildern und Texten auch zufällige Farben angezeigt werden, die in die zu schreibende Geschichte übernommen werden müssen.

Der *Calliope mini* als Rechtschreibtrainer

Die Übung

In dieser Übung wird der *Calliope mini* als Rechtschreibtrainer programmiert. Wird er kopfüber gedreht und anschließend wieder in seine Ausgangslage gebracht, erscheint auf dem LED-Bildschirm ein Wort in Laufschrift. Ein Buchstabe dieses Wortes ist durch einen Unterstrich ersetzt. Dem Wort folgen zwei Buchstaben, von denen einer der richtige Ergänzungsbuchstabe ist. Wenn der erste Buchstabe richtig ist, muss die Taste A gedrückt werden, wenn der zweite richtig ist, die Taste B. Ob die Eingabe richtig oder falsch ist, zeigt anschließend ein Haken (richtig) oder ein Kreuz (falsch) an. Die Eingabe der richtigen Lösung wird durch einen höheren Ton und die grün leuchtende RGB-LED verstärkt, eine Falscheingabe wird von einem tieferen Ton und der rot leuchtenden RGB-LED (falsch) begleitet. Ein erneutes Drehen des *Calliope mini* auf den Kopf und zurück in die aufrechte Lage lässt das nächste Wort erscheinen.
Die angezeigten Wörter lassen sich beliebig ändern und ergänzen.

Fachbezug

Deutsch

Die Schülerinnen und Schüler üben mit dem *Calliope mini* Wörter und überprüfen diese auf ihre orthografische Richtigkeit. Dabei können sie sich während des Programmierprozesses individuell Wörter zusammenstellen, die noch nicht vollständig gesichert sind und diese jeweils dem aktuellen Wissensstand anpassen.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Texte verfassen* richtig schreiben. Sie nutzen Regelmäßigkeiten der normgerechten Schreibung, überprüfen ihre Texte auf orthografische Richtigkeit und wenden Rechtschreibmuster und -strategien an.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Schreiben* Rechtschreibstrategien anwenden, sie verfügen über Fehlersensibilität und Rechtschreibgespür und nutzen Rechtschreibprogramme elektronischer Medien als Korrekturhilfe.

Anforderungen

Programmierschwerpunkte:

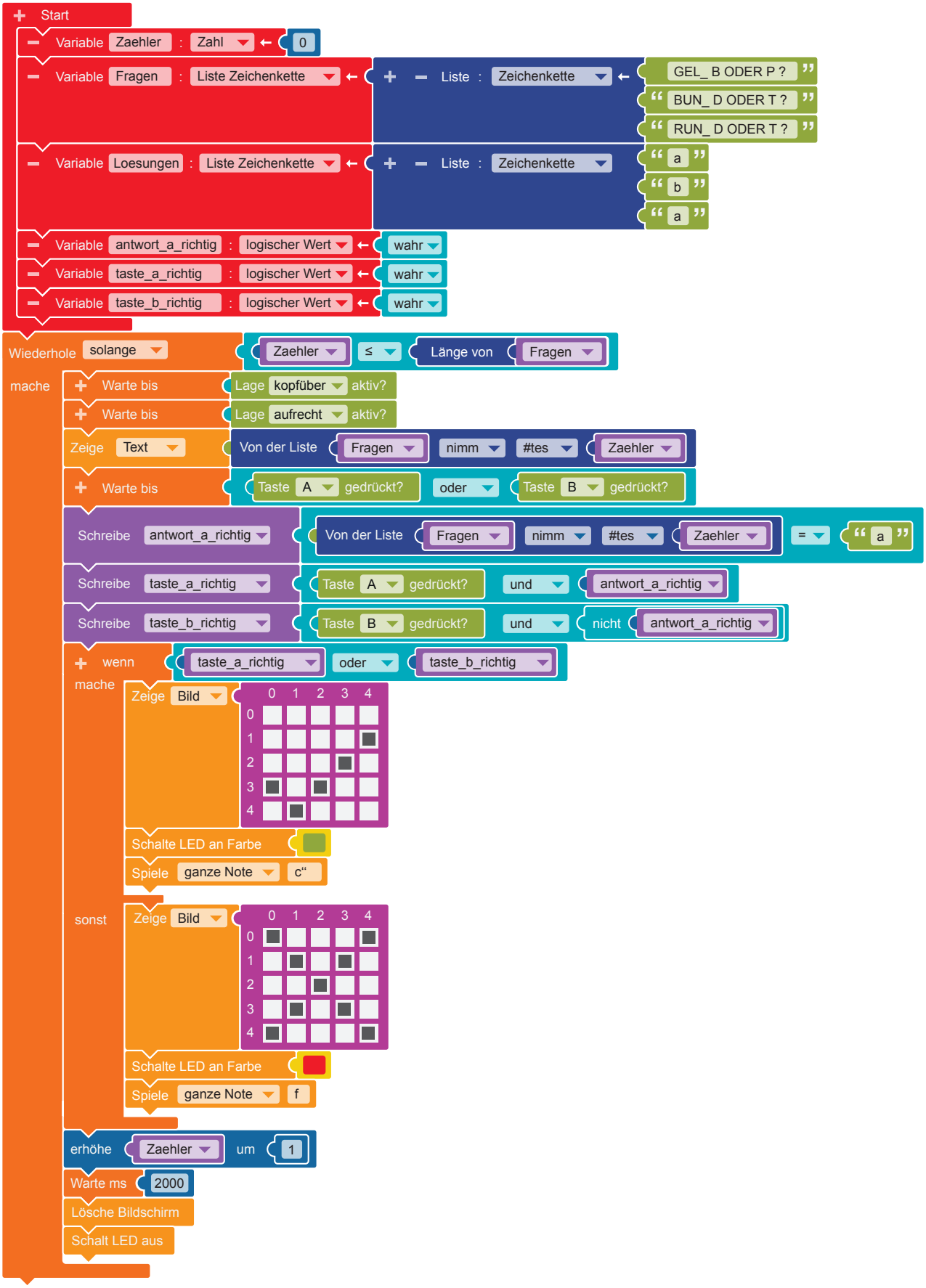
- Eingabe über Tasten, Ausgabe über LED-Bildschirm, RGB-LED und Lautsprecher
- Strukturen: bedingte Schleife, bedingtes Warten, Verzweigung
- Bedingungen formulieren
- mit Listen arbeiten (Index ▶ Position eines Listenelements)
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle, Logik, Mathematik, Listen, Variablen

Programmierschwierigkeit:

- fortgeschrittenes Niveau
- *NEPO*®-Level: Experte

Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.



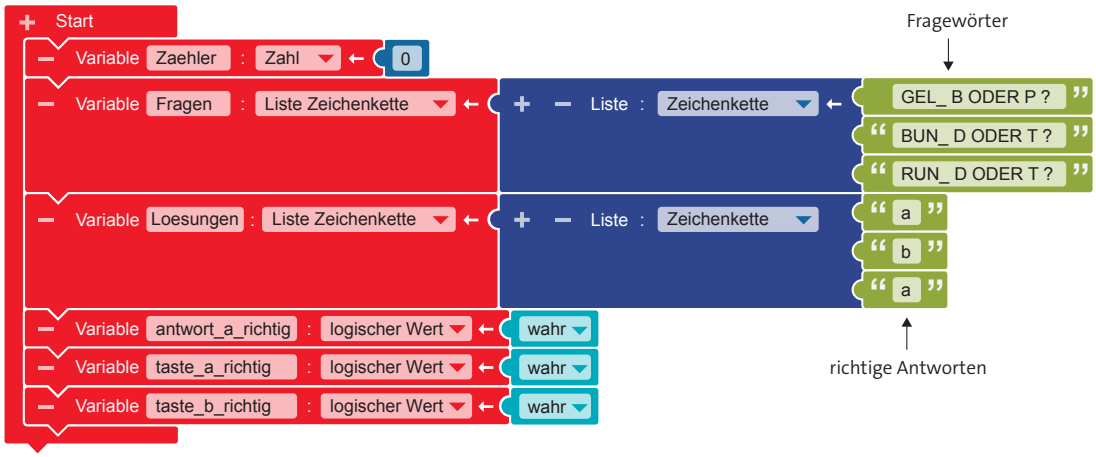
Schritte zur Erstellung des Programms für den Rechtschreibtrainer

1 Für den Rechtschreibtrainer sollen Übungswörter und deren richtige Schreibung festgelegt werden:
Um eine neue Variable anzulegen, klicken Sie auf das „+“-Symbol links neben „Start“. Ein neuer Block erscheint. Ebenso wird durch einen Klick auf das „+“ im blauen Listenblock eine weitere **Zeichenkette** hinzugefügt.

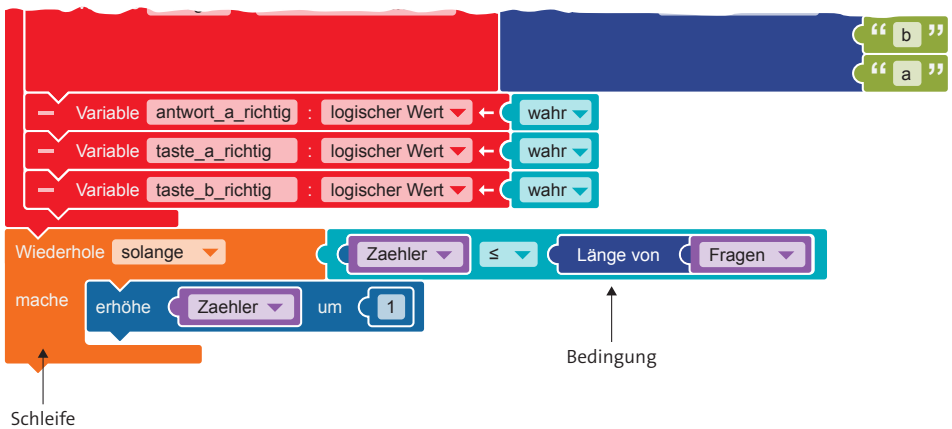
Es werden insgesamt sechs Variablen erstellt:
- eine mit dem Namen *Zaehler* vom **Datentyp** „Zahl“.
- zwei weitere **Variablen** mit den Namen *Fragen* und *Loesungen* vom Datentyp „**Liste** Zeichenkette“.

Die Fragewörter (frei wählbar) und die richtige Antwort („a“ oder „b“) tragen Sie in die hellgrünen Textfelder ein.
Hinweis: Der Unterstrich bei den Fragewörtern steht für den einzusetzenden Buchstaben.

Schließlich benötigen Sie noch drei Variablen vom Datentyp „logischer Wert“ mit den Namen *antwort_a_richtig*, *taste_a_richtig* und *taste_b_richtig*.
Alle drei erhalten zunächst den **Wahrheitswert** „wahr“ aus der Kategorie Logik.



2 Es wird eine **Schleife** (wiederholte **Anweisung**) erstellt, wobei die Bearbeitung jeder Frage einen Schleifendurchlauf darstellt:
Wählen Sie aus der Kategorie Kontrolle ▶ Schleifen den Block „Wiederhole solange/mache“ und fügen Sie ihn an die roten Variablenblöcke an.
Diese Schleife soll solange wiederholt werden, wie es Fragen in der Liste gibt.
Die Variable *Zaehler* hat anfänglich den Wert „0“ und im letzten Schleifendurchlauf bei drei Fragen den Wert „2“. Dies liegt daran, dass Listenindizes immer mit dem **Index** „0“ beginnen.
Für die Schleifen**bedingung** benötigen Sie aus der Kategorie Logik den **Vergleich**sblock „≤“, aus der Kategorie Variablen die Blöcke „Zaehler“ und „Fragen“ und aus der Kategorie Listen den Block „Länge von“. In die Schleife fügen Sie aus der Kategorie Mathematik den Block „erhöhe um“ ein, in den Sie den Zahlenwert „1“ eingeben.



3 Der Spieler übernimmt über folgende Anweisungen die Steuerung des Rechtschreibtrainers:
Wählen Sie aus der Kategorie **Kontrolle** den Block „Warte bis“ und fügen ihn dreimal in die Schleife vor dem Block „erhöhe Zähler um 1“ ein.
An die ersten beiden „Warte bis“-Blöcke docken Sie aus der Kategorie **Sensoren** die Blöcke „Lage kopfüber“ und „Lage aufrecht“ an.

An den dritten „Warte bis“-Block fügen Sie aus der Kategorie **Logik** den Block  an. Wählen Sie im **Ausklappmenü** „oder“ aus. Die beiden Lücken füllen Sie mit den Blöcken „Taste A gedrückt?“ und „Taste B gedrückt?“ aus der Kategorie **Sensoren**.

Jetzt ist der *Calliope mini* bereit für die nächste Frage: Um diese zufällig anzuzeigen, wählen Sie den Block „Zeige Text“ aus der Kategorie **Aktion** aus und fügen aus der Kategorie **Listen** den Block „von der Liste „Fragen“ nimm #tes“ ein. Die beiden Lücken füllen Sie mit den Blöcken „Fragen“ und „Zaehler“ aus der Kategorie **Variablen**, da diese als Listenindex verwendet werden.

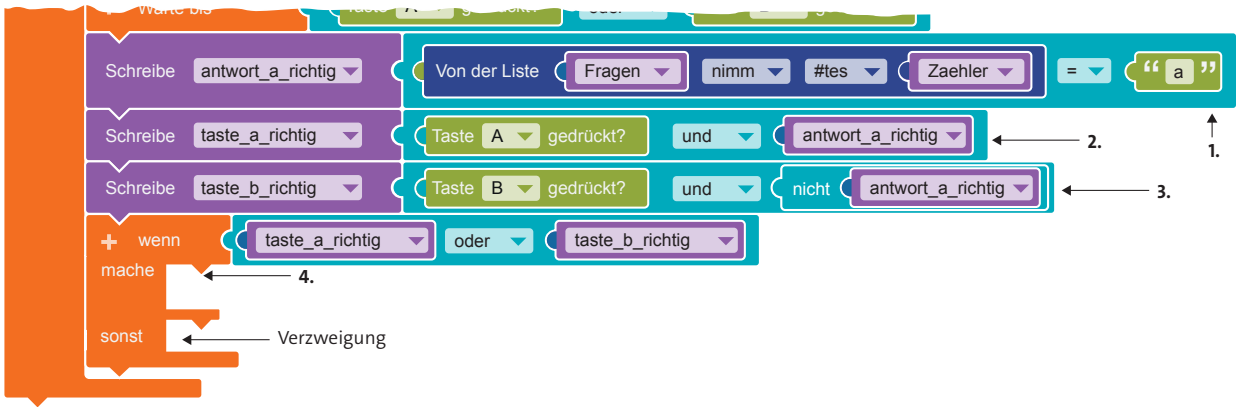
4 Die nächsten vier Schritte überprüfen, ob die gedrückte Taste zur richtigen Lösung passt:
Als Vorbereitung wählen Sie aus der Kategorie **Variablen** drei „Schreibe“-Blöcke aus, innerhalb derer Sie die Variablen, *antwort_a_richtig*, *taste_a_richtig* und *taste_b_richtig* auswählen. An den ersten „Schreibe“-Block fügen Sie aus der Kategorie **Logik** den Block an, an die beiden anderen den Block  .

1. Ist der Eintrag in der Liste „Loesungen“ gleich „a“? Fügen Sie in die erste Bedingung aus der Kategorie **Listen** den Block „von der Liste _ nimm #tes“ ein. Die linke Lücke füllen Sie mit dem Block „Loesungen“, die rechte mit dem Block „Zaehler“ jeweils aus der Kategorie **Variablen**. Auf der rechten Seite vom Gleichheitszeichen setzten Sie den Buchstaben „a“ aus der Kategorie **Text** ein.
2. Wenn die Antwort „a“ richtig ist, ist auch die Taste „A“ gedrückt worden?
3. Wenn die Antwort „a“ nicht richtig ist, muss Antwort „b“ richtig sein. Ist dann die Taste „B“ gedrückt?

Schritt **2** und **3**:
In die beiden folgenden **Bedingungen** fügen Sie aus der Kategorie **Sensoren** „Taste A gedrückt?“ und „Taste B gedrückt?“ auf der linken Seite ein und auf der rechten Seite wird die **Variable** *antwort_a_richtig* zweimal eingesetzt. Dabei wird bei der unteren Bedingung **3** die **Variable** *antwort_a_richtig* vom Negationsblock „nicht“ aus der Kategorie **Logik** umschlossen.

4. Ist das Ergebnis aus einer der vorigen zwei Fragen (Schritt 2 oder 3) wahr, hat der Spieler eine richtige Antwort gegeben.

Es folgt eine **Verzweigung** aus der Kategorie **Kontrolle ▶ Entscheidungen** mit dem Block „wenn/mache/sonst“. An diese docken Sie den Bedingungsblock  aus der Kategorie **Logik** an. Das „und“ ändern Sie im **Ausklappmenü** in „oder“ und ergänzen links und rechts vom „oder“ die Blöcke *taste_a_richtig* sowie *taste_b_richtig* aus der Kategorie **Variablen**.

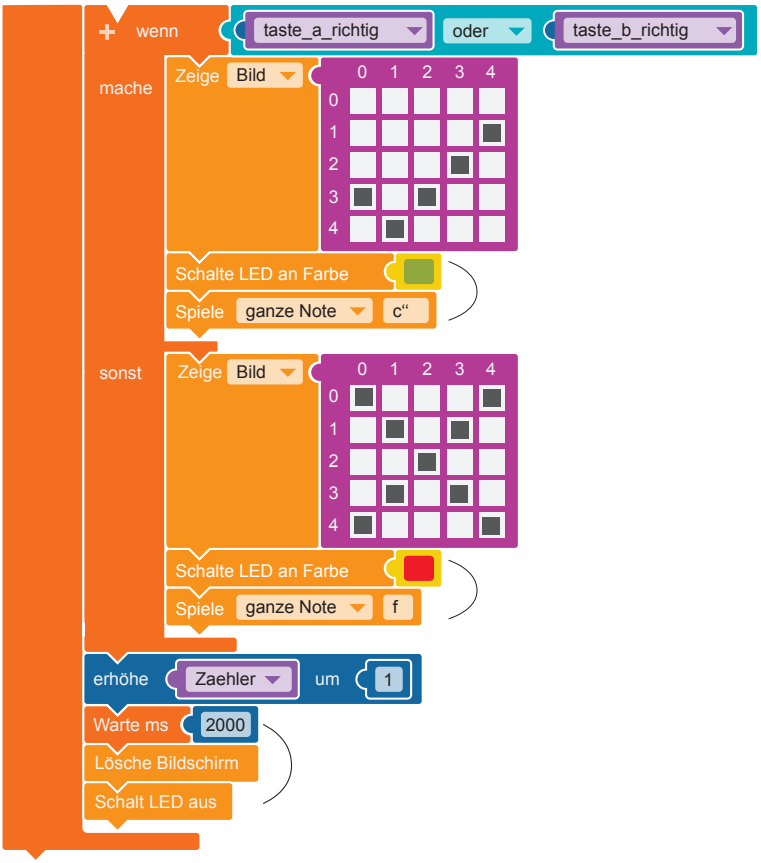


5 Es erfolgt nun eine zur richtigen und zur falschen Antwort passende Ausgabe in Form eines Bildes:
Wenn der Vergleich der Tasteneingabe mit der richtigen Antwort übereinstimmt, kann nun innerhalb der Verzweigung hinter „mache“ über die Kategorie **Aktion ▶ Anzeige** und den Block „Zeige Bild“ z. B. ein Haken eingefügt werden.
Hinter „sonst“, wenn eine nicht zur richtigen Antwort passende Taste gedrückt wurde, wird entsprechend ein Bild für den **LED-Bildschirm** mit z. B. einem „X“ eingefügt.

Als zusätzliche Rückmeldung an den Anwender soll die **RGB-LED** leuchten und ein Ton zu hören sein.

Zwischen jeder Frage ist der LED-Bildschirm zu löschen und die RGB-LED auszuschalten:
An diesen Stellen können Sie aus der Kategorie **Aktion ▶ Statusleuchte** den Block „Schalte LED an Farbe“ und aus **Aktion ▶ Klang** den Block „Spiele ganze Note c“ für die zusätzlichen Ausgaben für die RGB-LED und die Tonausgabe ergänzen.

Zwischen den Fragen, also hinter den Block „wenn/mache/sonst“ wird der LED-Bildschirm nach einer Wartezeit (**Kontrolle ▶ Warten**) von zwei Sekunden (= 2000 Millisekunden) gelöscht und die RGB-LED wieder ausschalten (**Aktion ▶ Anzeige** und **Aktion ▶ Statusleuchte**).



6 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

- beliebige Rechtschreibübungen, bei denen einzelne Buchstaben die Schreibschwierigkeit des Wortes ausmachen, z. B. Rechtschreibphänomene wie die Auslautverhärtung
- das Finden solcher Wörter in Texten oder Übungen z. B. aus dem Schulbuch und das Hinzufügen dieser Wörter zu der Frageliste im Rechtschreibtrainer

Erweiterungsmöglichkeiten

- eine Erweiterung wäre das Zählen der richtigen Antworten als weitere **Variable** *Punkte*
- ebenso könnte das jeweilige Wort in richtiger Schreibung nach jeder Frage angezeigt werden. Möglicherweise auch im z. B. Plural oder im Infinitiv, um die richtige Schreibung zu begründen
- ähnlich wie im Programmbeispiel „Morsen mit dem *Calliope mini*“ können Fragen und Antworten per Funk zwischen zwei *Calliope mini* übertragen werden (Kategorie **Nachrichten** ▶ Block „Sende Nachricht“)

Der *Calliope mini* als Zufallsgenerator

Die Übung

In der nachfolgenden Übung wird der *Calliope mini* zu einem Zufallsgenerator (Ziffern-Würfel) programmiert. Durch das Drücken einer Taste wird eine zufällige Zahl zwischen 1 und 6 ausgegeben. In einem weiteren Schritt kann der Würfel so programmiert werden, dass anstelle der Ziffer die jeweilige Augenzahl des Würfels angezeigt wird.

Fachbezug

Mathematik
Die Schülerinnen und Schüler programmieren den *Calliope mini* zum Zufallsgeber (sechsflächiger Würfel). Sie können mit ihm einfache Zufallsexperimente durchführen und die Wahrscheinlichkeit von Ereignissen einschätzen, beschreiben und vergleichen.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Daten, Häufigkeit und Wahrscheinlichkeit* einfache Zufallsexperimente durchführen sowie die Wahrscheinlichkeit von Ereignissen bei einfachen Zufallsexperimenten einschätzen, beschreiben (sicher, möglich, unmöglich) und vergleichen, indem sie den programmierten *Calliope mini*-Würfel spielerisch-praktisch für Zufallsexperimente einsetzen.

Prozessbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Argumentieren* Fragen stellen und Vermutungen äußern sowie Begründungen suchen (auch von Gesetzmäßigkeiten).

Anforderungen

Programmierschwerpunkte:
- Eingabe über Tasten, Ausgabe über LED-Bildschirm
- Strukturen: bedingte Anweisung, Schleife
- Bedingungen formulieren
- mit Variablen arbeiten (Code 2)
- Eingabewerte vergleichen; Logik (Code 2)
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle, Logik, Mathematik, Variablen

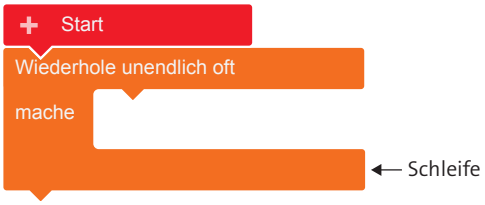
Programmierschwierigkeit:
- Einstiegsniveau
- *NEPO*®-Level: Experte

Der Code 1: Programmierung eines Ziffern-Würfels

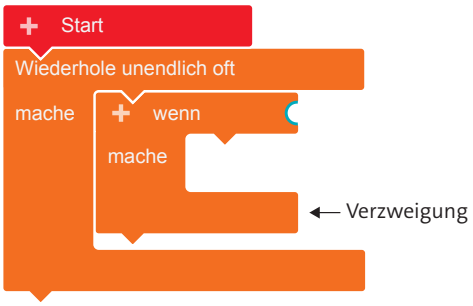
So sieht der Code des gesamten Programms aus. Nachfolgend wird dieser Schritt für Schritt erarbeitet.

Schritte zur Erstellung des Programms für einen Ziffern-Würfel

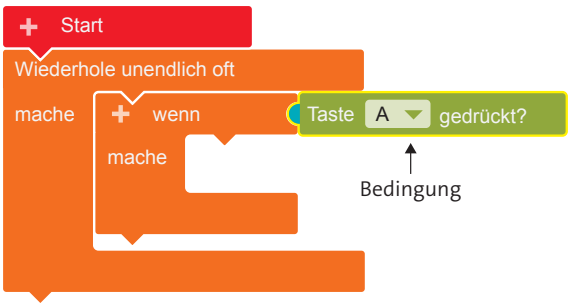
1 Damit unendlich viele Zufallszahlen ausgegeben werden können, wird eine Endlos**schleife** benötigt:
Wählen Sie aus der Kategorie **Kontrolle ▶ Schleifen** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



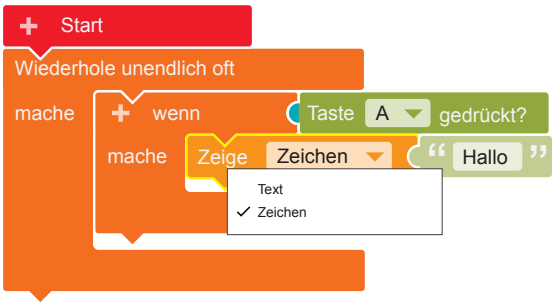
2 Wenn die Taste A gedrückt wird (*wenn*), soll eine Zufallszahl ausgegeben werden (*mache*):
Dafür benötigen Sie eine **Verzweigung**. Wählen Sie aus der Kategorie **Kontrolle ▶ Entscheidungen** den Block „wenn/mache“ und fügen Sie ihn in die Schleife ein.



3 Die gewünschte **Eingabe**möglichkeit (hier über den **Sensor** „Taste A“) wird ausgewählt:
Wählen Sie aus der Kategorie **Sensoren** den Block „Taste A gedrückt?“ und fügen Sie ihn als **Bedingung** (blauer Bereich) an die Verzweigung an.



4 Die Zufallszahl soll auf dem **LED-Bildschirm** angezeigt werden:
Wählen Sie den Block **Aktion ▶ Anzeige** den Block „Zeige Text“ und klicken Sie im **Ausklappmenü** auf „Zeichen“.

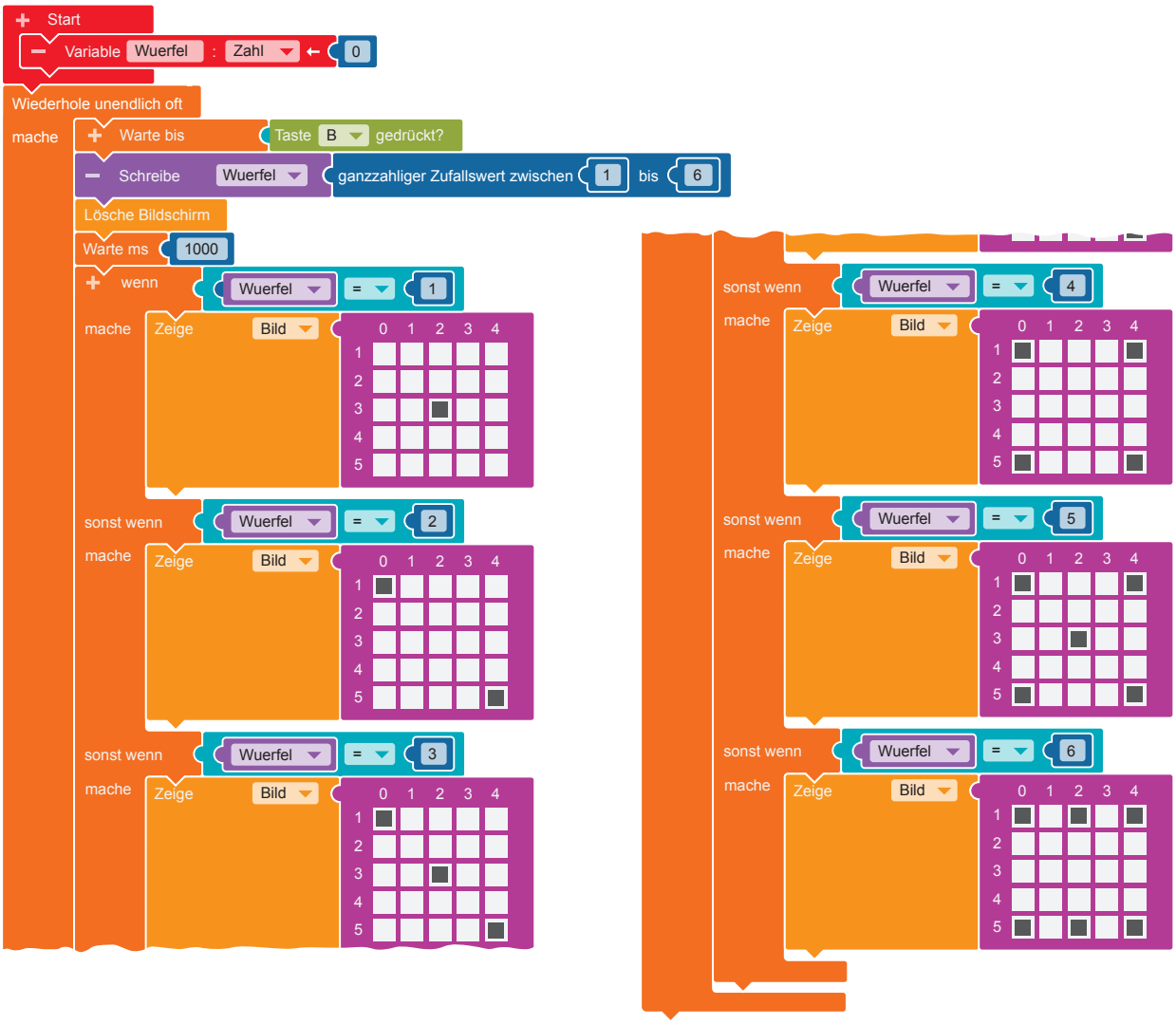


5 Im letzten Schritt muss der Wert der **Zufall**szahl festgelegt werden:
Ersetzen Sie den am Block „Zeige Zeichen“ hängenden Text „Hallo“ durch den Block „ganzzahliger Zufallswert zwischen 1 bis 100“ aus der Kategorie **Mathematik**.
Ersetzen Sie den Wert 100 durch die Zahl 6 (sechsflächiger Würfel), indem Sie auf die 100 klicken und über die Tastatur die Zahl 6 eingeben.

6 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Der Code 2: Programmierung eines Punkte-Würfels

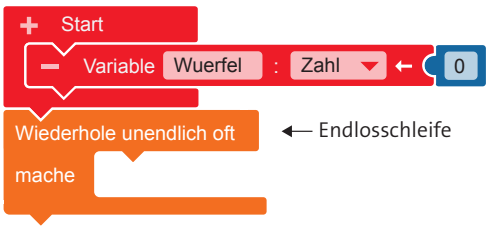
So sieht der Code des gesamten Programms aus. Nachfolgend wird dieser Schritt für Schritt erarbeitet.



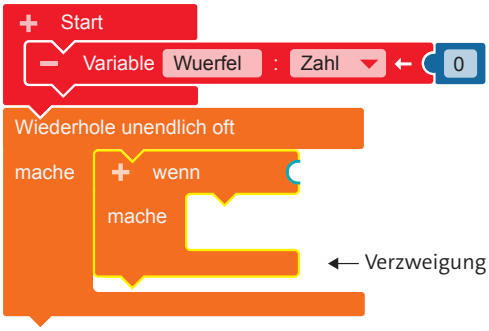
Schritte zur Erstellung des Programms für einen Punkte-Würfel

1 Damit immer wieder neue Zahlen ausgegeben werden können, muss eine **Variable** angelegt und definiert werden:
Um eine neue Variable anzulegen, klicken Sie auf das „+“ links neben „Start“. Ein neuer Block erscheint. Legen Sie darin den Namen (*Wuerfel*) und **Datentyp** (hier: Zahl) der Variablen fest. Klicken Sie dafür auf das jeweilige Feld.

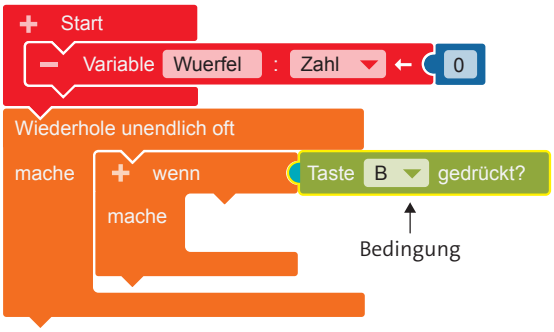
2 Es sollen unbegrenzt Zufallszahlen ausgegeben werden können:
Hierfür benötigen Sie eine Endlos**schleife**.
Wählen Sie aus der Kategorie **Kontrolle** ▶ **Entscheidungen** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



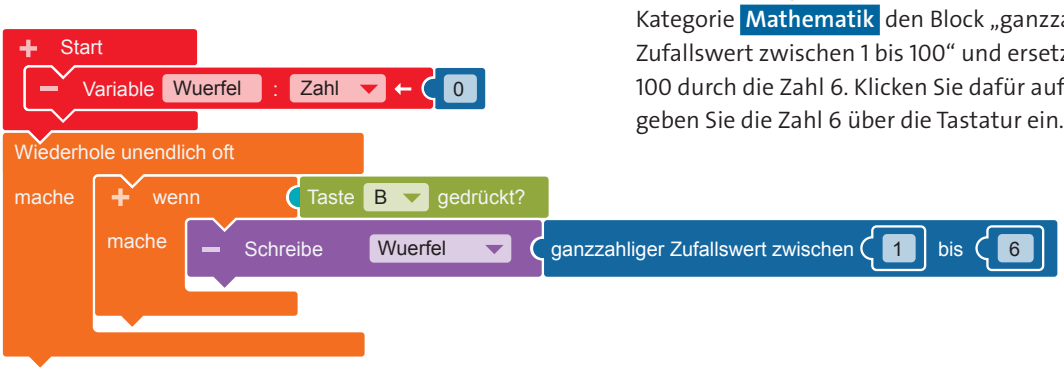
3 Wenn die Taste B gedrückt wird, soll eine Zufallszahl ausgegeben werden:
Um diese **Bedingung** aufzustellen, benötigen Sie eine **Verzweigung**.
Wählen Sie den Block **Kontrolle ▶ Entscheidungen** den Block „wenn/mache“ und fügen Sie ihn in die Schleife ein.



4 Die gewünschte Eingabemöglichkeit (hier über den Sensor „Taste B“) wird ausgewählt:
Wählen Sie aus der Kategorie **Sensoren** den Block „Taste B gedrückt?“ und fügen Sie ihn als **Bedingung** (blauer Bereich) an die Verzweigung an.

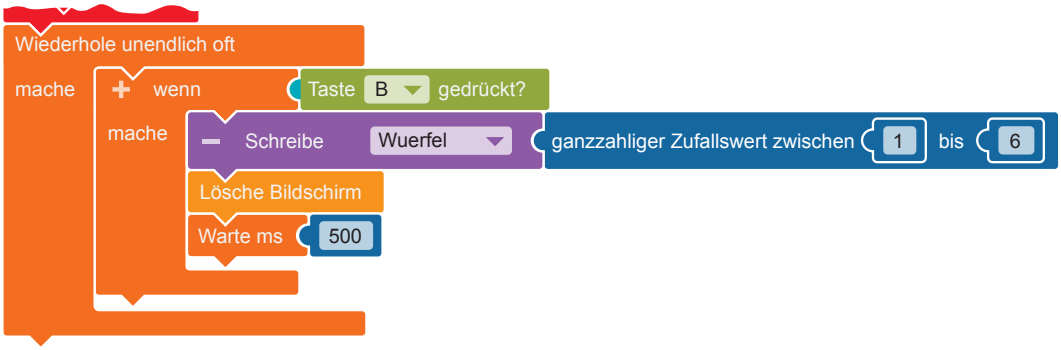


5 Der Wert der Variablen ***Zufallszahl*** muss definiert werden:



Wählen Sie aus der Kategorie **Variablen** den Block „Schreibe *Wuerfel*“. Daran platzieren Sie aus der Kategorie **Mathematik** den Block „ganzzahliger Zufallswert zwischen 1 bis 100“ und ersetzen den Wert 100 durch die Zahl 6. Klicken Sie dafür auf 100 und geben Sie die Zahl 6 über die Tastatur ein.

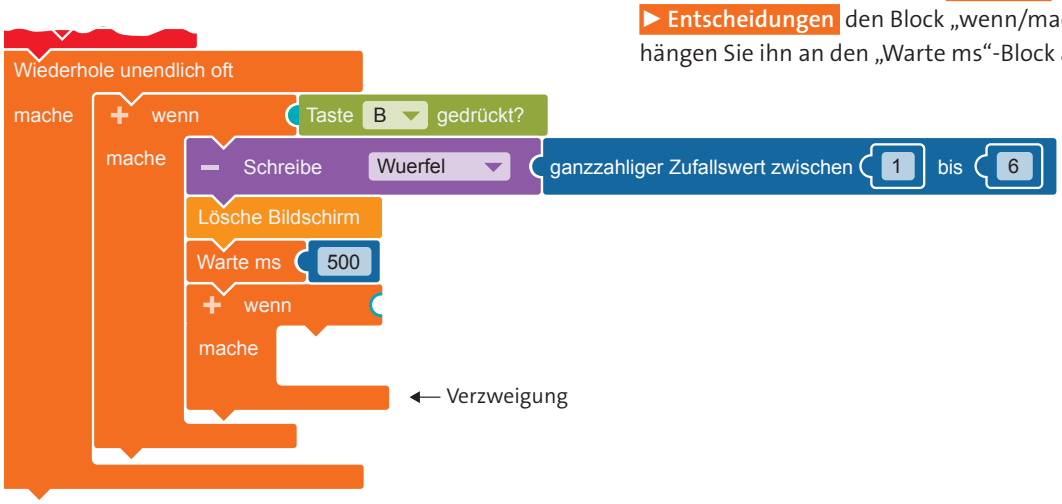
6 Eine bessere Lesbarkeit der Zufallszahlen soll gewährleistet werden:
Löschen Sie den Bildschirminhalt, indem Sie aus der Kategorie **Aktion ▶ Anzeige** den Block „Lösche Bildschirm“ einfügen.



Fügen Sie im Anschluss eine Pause ein. Wählen Sie aus der Kategorie **Kontrolle ▶ Warten** den Block „Warte ms“ und legen Sie die Länge der Pause fest (1000 ms = 1 Sekunde).

7 Die Zufallswerte müssen mit den Punktebildern des Würfels verknüpft werden:

Hierfür benötigen Sie eine weitere Verzweigung. Wählen Sie aus der Kategorie **Kontrolle** ▶ **Entscheidungen** den Block „wenn/mache“ und hängen Sie ihn an den „Warte ms“-Block an.



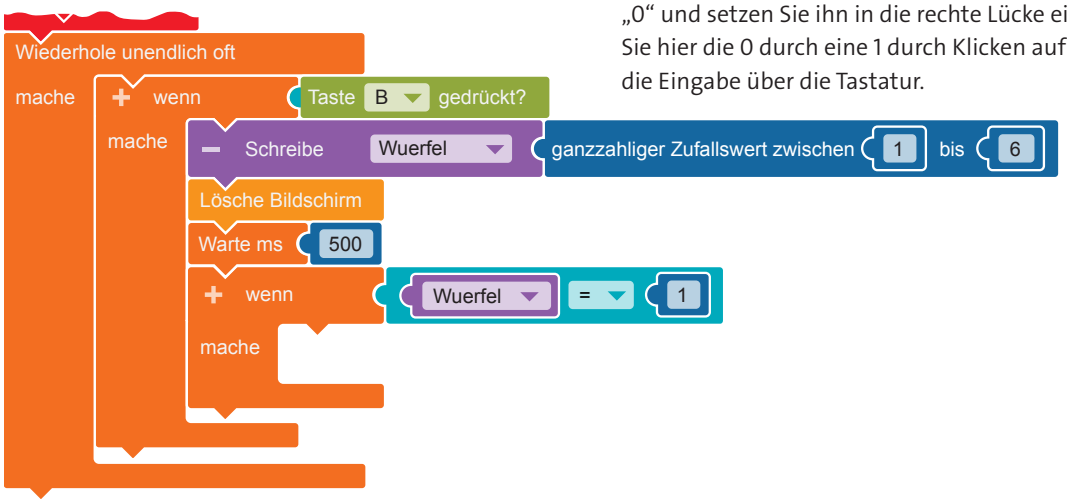
8 Der Zufallswert muss abgefragt werden, damit das entsprechende Punktebild auf dem LED-Bildschirm ausgegeben werden kann:

Dazu benötigen Sie einen **Vergleich** zwischen zwei Zahlen.

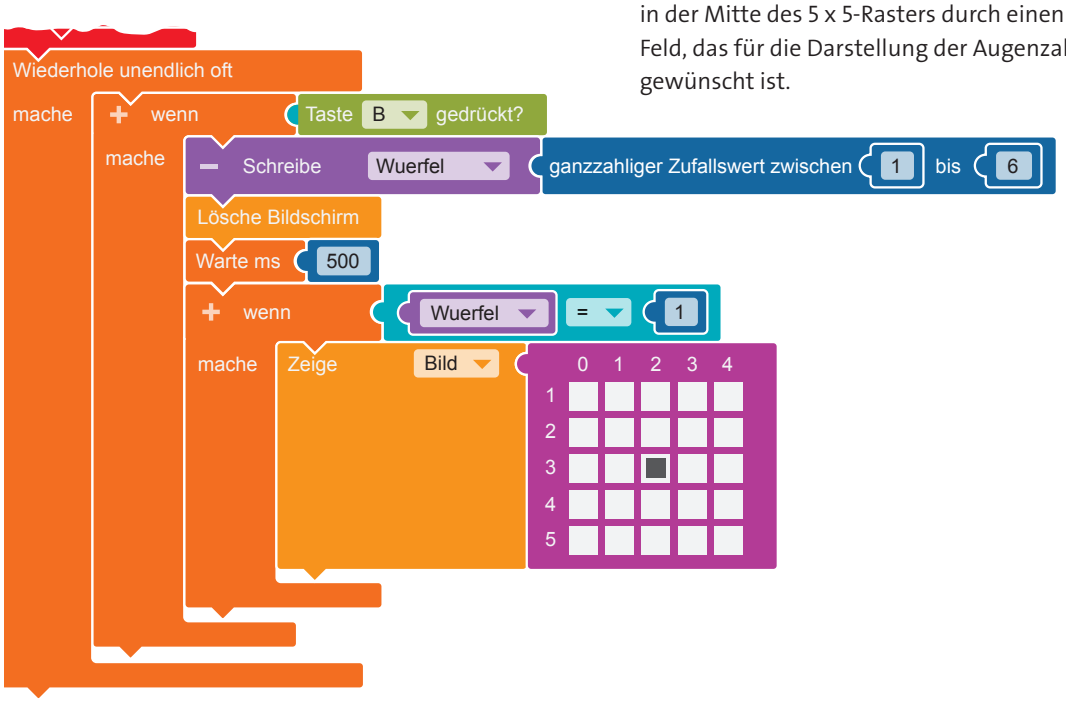
Wählen Sie aus der Kategorie **Logik** den Block  und fügen Sie ihn als Bedingung an die Verzweigung an.

9 Die Zufallszahl aus der Variablen *Wuerfel* wird nun mit der Zahl 1 verglichen:

Wählen Sie aus der Kategorie **Variablen** den Block „*Wuerfel*“ und ziehen die Variable in die linke Lücke des Logik-Blocks.
Nehmen Sie aus der Kategorie **Mathematik** den Block „0“ und setzen Sie ihn in die rechte Lücke ein. Ersetzen Sie hier die 0 durch eine 1 durch Klicken auf die Null und die Eingabe über die Tastatur.

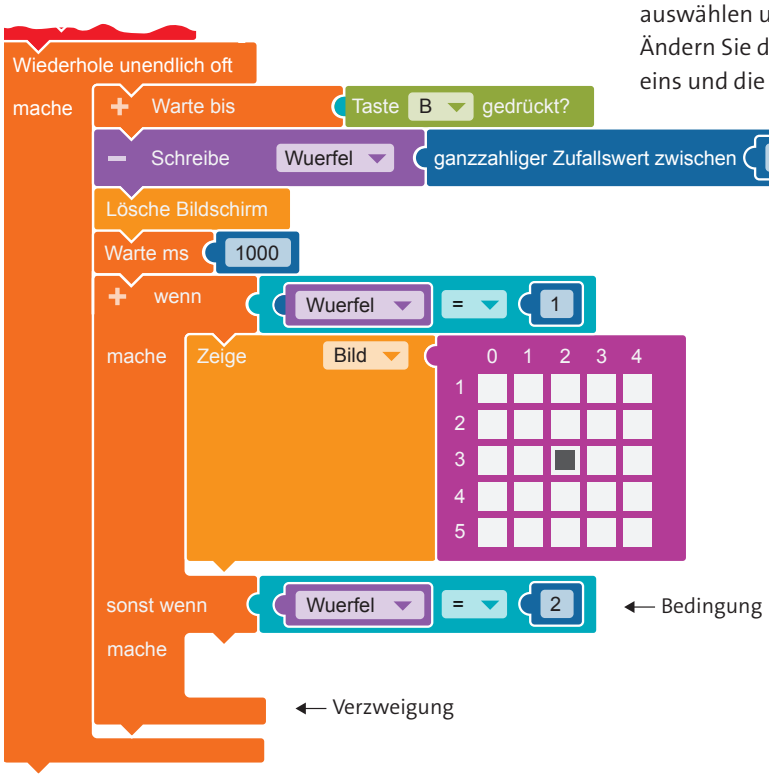


10 Das Punktebild für die Augenzahl 1 wird über den LED-Bildschirm angezeigt:



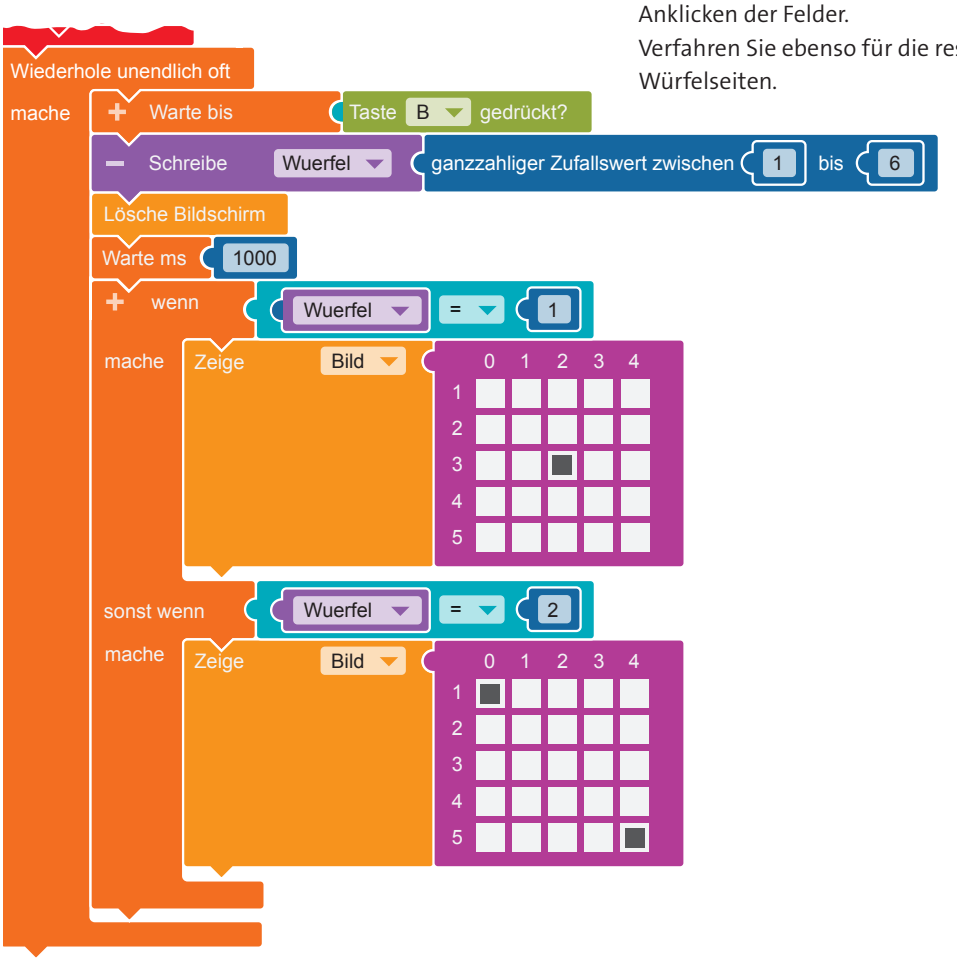
Wählen Sie aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Bild“ und markieren Sie hier den Punkt in der Mitte des 5 x 5-Rasters durch einen Klick auf das Feld, das für die Darstellung der Augenzahl 1 gewünscht ist.

11 Das Punktebild für die Augenzahl 2 soll verknüpft und über den LED-Bildschirm ausgegeben werden:
Klicken Sie nun in der Verzweigung auf das „+“ neben dem „wenn“.



Eine weitere Verzweigung erscheint.
Kopieren Sie nun den kompletten **Logik**-Block aus Schritt 10, indem sie mit der rechten Maustaste auf den zu kopierenden Block klicken und „kopieren“ auswählen und hängen Sie ihn an die Verzweigung an. Ändern Sie den Zahlenwert auf 2 durch Klicken auf die eins und die Eingabe über die Tastatur.

12 Das Punktebild für die Augenzahl 2 wird angezeigt: Kopieren Sie den Block **Aktion ▶ Anzeige** „Zeige Bild“ und markieren Sie die entsprechenden Kästchen für die Darstellung der Augenzahl 2 durch entsprechendes Anklicken der Felder.
Verfahren Sie ebenso für die restlichen vier Würfelseiten.



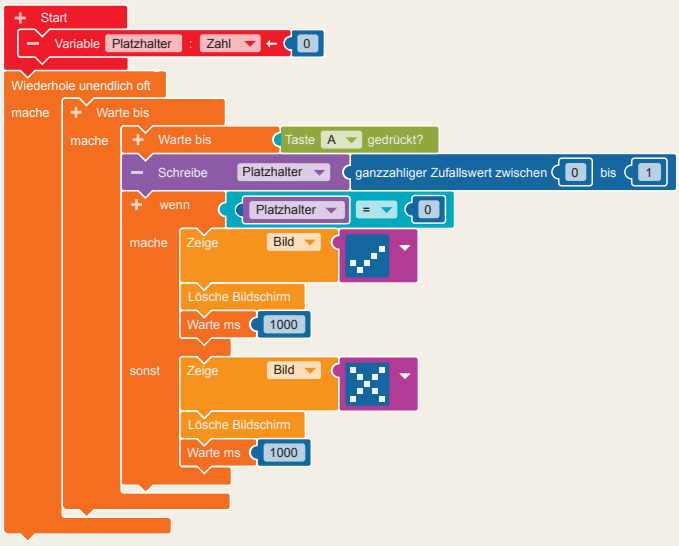
Hinweise und Infos

Einsatzszenarien im Unterricht

- Eigene Erfahrungen beim Würfeln (Glück, Pech) und Erkenntnisse aus Versuchsreihen mit sehr vielen Würfelvorgängen vergleichen.
- Vermutungen zu Zufallsexperimenten und deren Ausgang anstellen.
- Häufigkeitslisten in Partnerarbeit erstellen (zunächst mit einem, dann mit zwei *Calliope mini*-Würfeln); Ergebnisse beschreiben und versuchen zu begründen
- den digitalen Würfel mit dem echtem Würfel vergleichen und diese manipulieren
- weitere platonische Körper (sechs- oder mehrflächige Körper) im Programmcode hinterlegen und untersuchen

Erweiterungsmöglichkeiten

Der *Calliope mini* kann auf ähnliche Weise auch zu einem Mini-Orakel programmiert werden. Bei einem Klick auf die Taste A erscheint per Zufall entweder ein Haken oder ein Kreuz, „ja“ oder „nein“, ein trauriges oder ein fröhliches Gesicht, etc. .



Der *Calliope mini* als 1x1-Kopfrechentrainer

Die Übung

Der *Calliope mini* wird in dieser Übung zum 1x1-Kopfrechentrainer programmiert. Bei einem Klick z. B. auf die Taste A wird eine Multiplikationsaufgabe angezeigt. Wird z. B. Taste B gedrückt, erscheint das Ergebnis dieser Aufgabe auf dem LED-Bildschirm. Beim Programmieren kann in zwei Schritten vorgegangen werden:

Code 1: Programmierung einzelner Multiplikationsaufgaben:
In einem ersten Schritt wird eine bzw. werden mehrere selbst definierte Multiplikationsaufgaben programmiert, wobei verschiedene Sensoren (z. B. Taste A oder B, Touch-Pin gedrückt, Schütteln, Lage kopfüber etc.) verwendet werden können.

Grenzen des Codes
Bei dem Code 1 werden vom Programm keine Rechenoperationen durchgeführt. Stattdessen muss das korrekte Ergebnis zuvor selbst ausgerechnet und hinterlegt worden sein. Dadurch ist die Arbeit mit dem 1x1-Kopfrechentrainer sehr bald erschöpft und der Anwender merkt schnell, dass eine Programmerweiterung notwendig ist.

Code 2: Programmierung des per Zufall generierten 1x1-Rechentrainers:
In einem nächsten Schritt muss folglich ein Programmiercode erstellt werden, bei dem der *Calliope-mini* selbst aktiv rechnet. Dabei sollen immer neue Multiplikationsaufgaben per Zufall ausgegeben und die zugehörigen richtigen Lösungen angezeigt werden.

Fachbezug

Mathematik
Die Schülerinnen und Schüler programmieren einen 1x1-Rechentrainer, der die beiden Faktoren einer Multiplikationsaufgabe zufällig ausgibt und auf Knopfdruck die richtige Lösung anzeigt.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Zahlen und Operationen* die Grundaufgaben des Kopfrechnens aus dem Gedächtnis abrufen, indem sie zunächst einen *Calliope mini*-Rechentrainer programmieren und anschließend als Nutzer die Rechenaufgaben trainieren.

Prozessbezogene Kompetenzen
Die Schülerinnen und Schüler können im Bereich *Darstellen* eine Darstellung in eine andere übertragen (in die Programmiersprache des Editors).

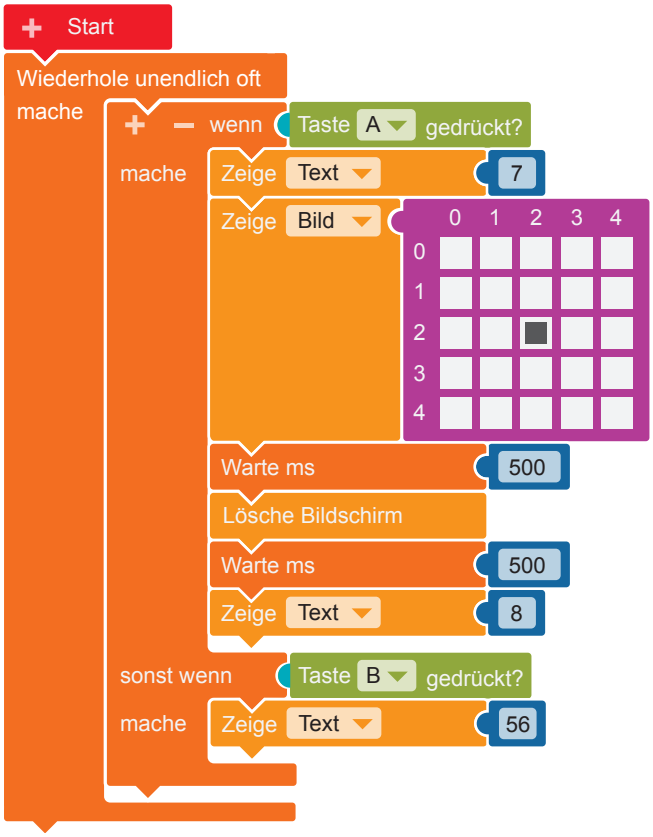
Anforderungen

Programmierschwerpunkte:
- verschiedene Eingabemöglichkeiten anwenden
- Ausgabe von Text (Zahl) und Bild
- Strukturen: Schleife, bedingtes Warten
- Bedingungen formulieren
- mit Variablen arbeiten (Code 2)
- verwendete *NEPO*®-Kategorien: Kontrolle, Aktion, Mathematik, Sensoren, Variablen (nur Code 2)

Programmierschwierigkeit:
- mittleres Niveau
- *NEPO*®-Level: Anfänger (Code 1) – Experte (Code 2)

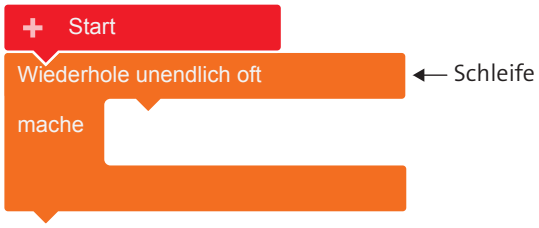
Der Code 1: Programmierung einzelner Multiplikationsaufgaben

So sieht der Code des gesamten Programms aus. Nachfolgend wird es Schritt für Schritt erarbeitet.

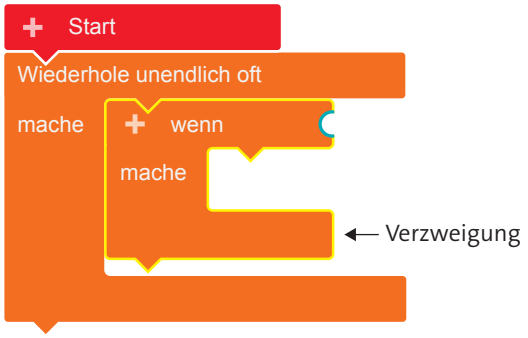


Schritte zur Erstellung des Programms für einzelne Multiplikationsaufgaben

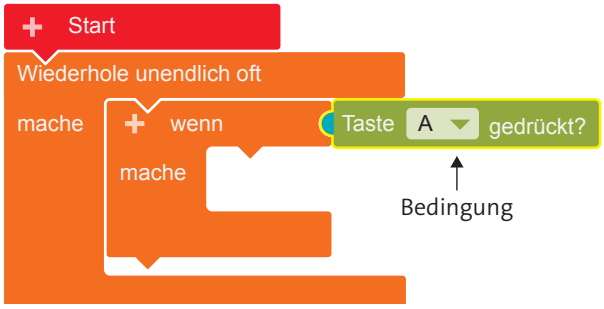
1. Damit die Multiplikationsaufgabe unendlich oft auf dem *Calliope mini* angezeigt werden kann, wird zu Beginn eine Endlos**schleife** angelegt:
Wählen Sie aus der Kategorie **Kontrolle** den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



2. Wenn die Taste A gedrückt wird (*wenn*), soll die Multiplikationsaufgabe angezeigt werden (*mache*):
Um diese **Bedingung** aufzustellen, benötigen Sie eine **Verzweigung**.
Wähle Sie aus der Kategorie **Kontrolle** den Block „wenn/mache“ und fügen Sie diesen in die Schleife ein.



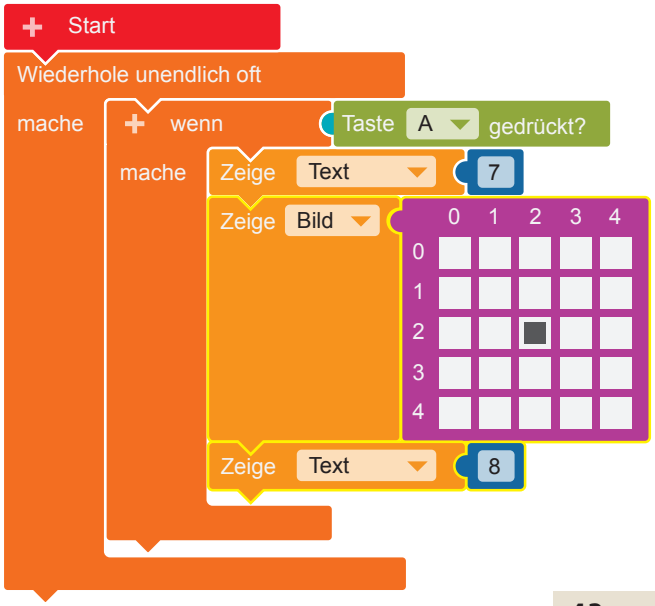
3. Die gewünschte Eingabemöglichkeit (hier über den Sensor „Taste A“) wird ausgewählt:
Wählen Sie aus der Kategorie **Sensoren** den Block „Taste A gedrückt?“ und fügen Sie ihn als Bedingung (blauer Bereich) an die Verzweigung an.



4. Der 1. Faktor der Multiplikationsaufgabe soll auf dem **LED-Bildschirm** angezeigt werden:
Wählen Sie aus der Kategorie **Aktion** den Block „Zeige Text “ und fügen Sie ihn in die Verzweigung ein.
Ersetzen Sie den Text „Hallo“ durch den Block „0“ aus der Kategorie **Mathematik**. Ergänzen Sie hier eine beliebige Zahl zwischen 0 und 10 (▶ Faktor 1) durch Klicken auf die Null und Eingabe über die Tastatur.

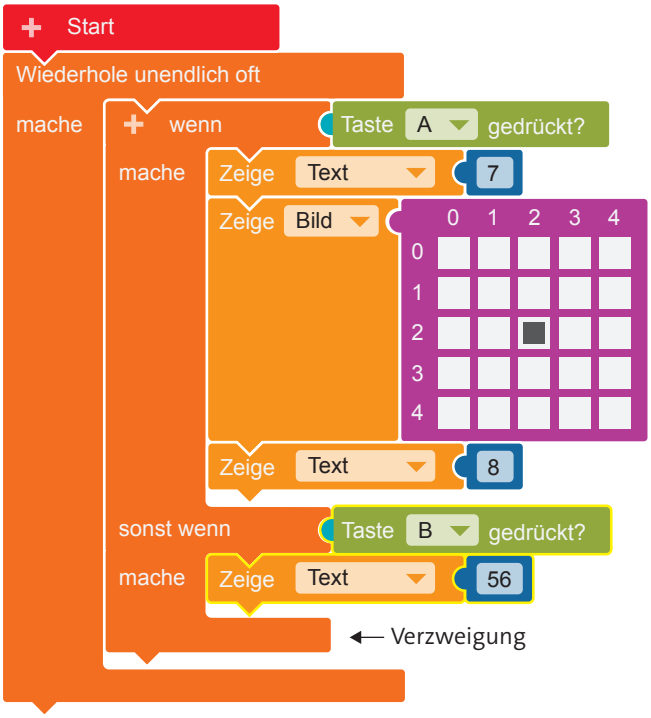
5. Das Multiplikationszeichen soll angezeigt werden:
Wählen Sie aus der Kategorie **Aktion** den Block „Zeige Bild“ und markieren Sie das Kästchen in der Mitte. Dieser Punkt erscheint auf dem LED-Bildschirm als Multiplikationszeichen.

   Geben Sie anschließend eine zweite Zahl (▶ Faktor 2) ein. Gehen Sie vor wie in Schritt 4.



6 Wenn die Taste B gedrückt wird (*wenn*), soll das Ergebnis der Multiplikationsaufgabe auf dem LED-Bildschirm angezeigt werden (*mache*):
Hierfür benötigen Sie eine weitere Verzweigung. Klicken Sie in der ersten Verzweigung auf das „+“ neben dem „wenn“.

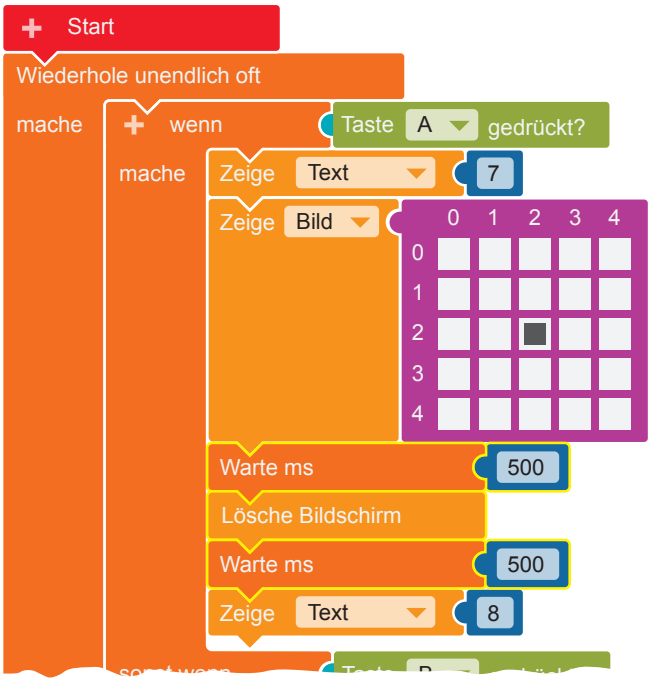
Wählen Sie zunächst aus der Kategorie **Sensoren** den Block „Taste B gedrückt?“ und fügen Sie ihn als Bedingung an die Verzweigung an.
Ergänzen Sie dann aus der Kategorie **Aktion** den Block „Zeige Text“. Ersetzen Sie den Text „Hallo“ durch den Block „0“ aus der Kategorie **Mathematik**. Tragen Sie hier das richtige Ergebnis der von Ihnen hinterlegten Multiplikationsaufgabe ein.



7 Beim Abspielen der Aufgabe ist das Multiplikationszeichen schwer zu erkennen. Durch das Einfügen von **Pausen** und Löschen des LED-Bildschirms wird die Lesbarkeit erhöht:
Wählen Sie aus der Kategorie **Kontrolle** den Block „Warte ms“ und legen Sie hier die Länge der Pause fest.

Um den Bildschirminhalt zu löschen, wählen Sie aus der Kategorie **Aktion** den Block „Lösche Bildschirm“.

Fügen Sie eine weitere Pause ein und testen Sie, welche Pausenlänge eine gute Lesbarkeit garantiert. (1000 ms = 1 Sekunde).

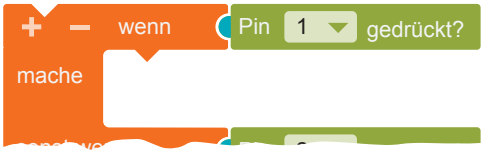


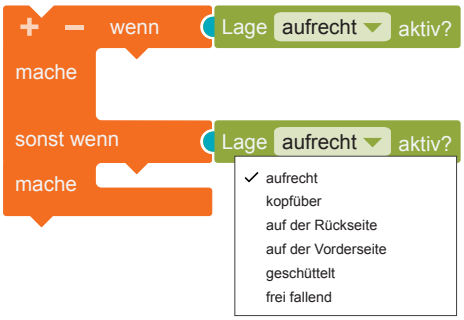
8 Erweitern Sie das Programm so, dass mehrere Aufgaben geübt werden können:
Legen Sie weitere Verzweigungen an und wählen Sie neue Bedingungen (**Sensoren**) aus, z. B.:

Sensoren ▶ „Pin 1 gedrückt?“

Sensoren ▶ „Lage aufrecht aktiv?“

Gehen Sie wie in Schritt 2–8 beschrieben vor.

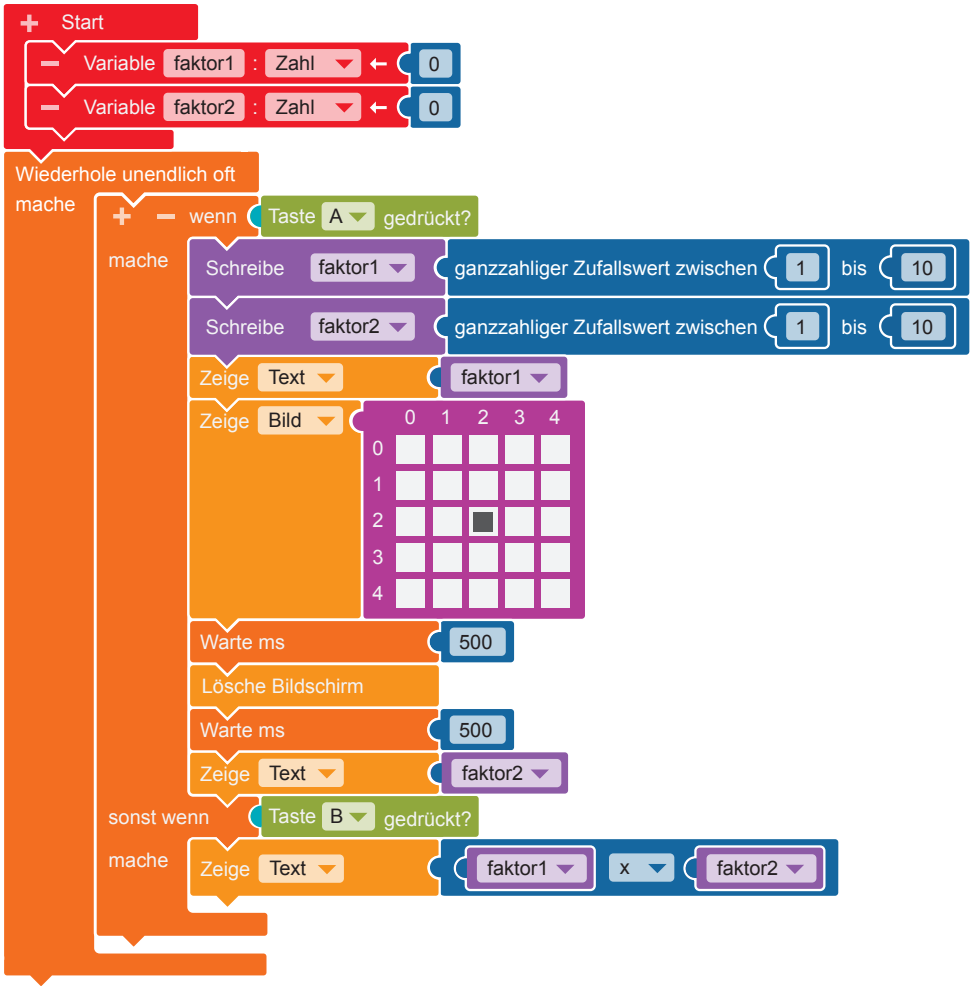




9 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

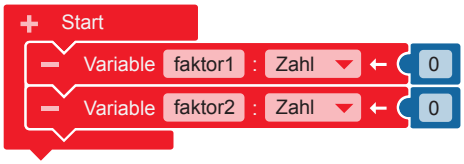
Code 2: Programmierung des per Zufall generierten 1x1-Kopfrechentrainers

So sieht der Code des gesamten Programms aus. Nachfolgend wird es Schritt für Schritt erarbeitet.

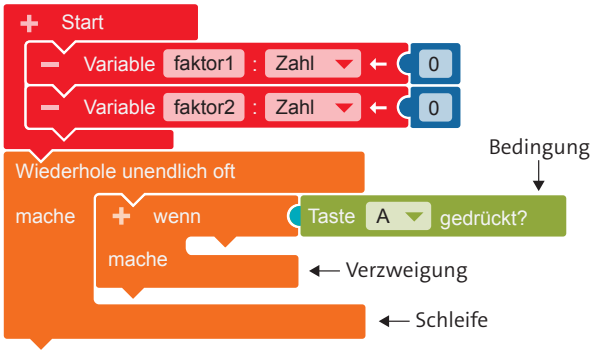


Schritte zur Erstellung des Programms für einen per zufallgenerierten 1x1-Kopfrechentrainer

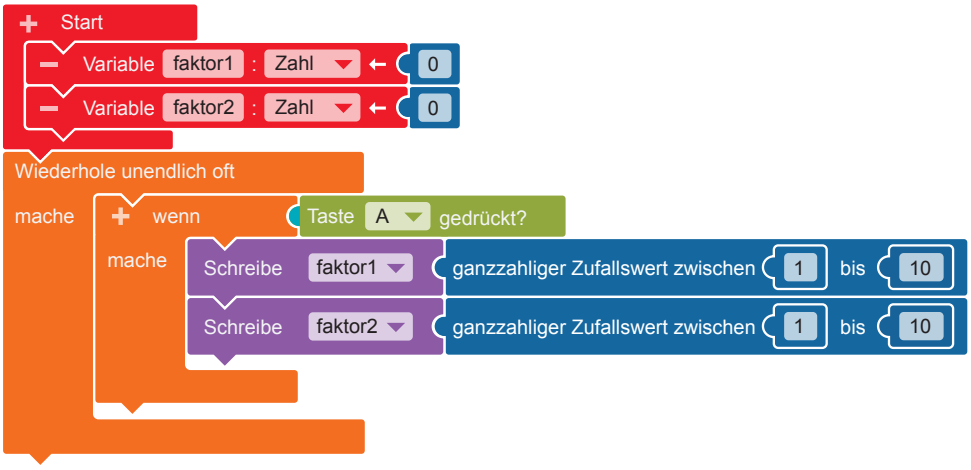
1 Damit immer wieder neue Aufgaben ausgegeben werden können, müssen zunächst zwei **Variablen** angelegt und definiert werden:
Sie benötigen Variablen für die beiden Faktoren (*faktor1* und *faktor2*).
Um eine neue Variable anzulegen, klicken Sie auf das „+“ links neben „Start“. Ein neuer Block erscheint.
Legen Sie darin den Namen (*faktor1/faktor2*) und Datentyp (hier: Zahl) der Variablen fest.



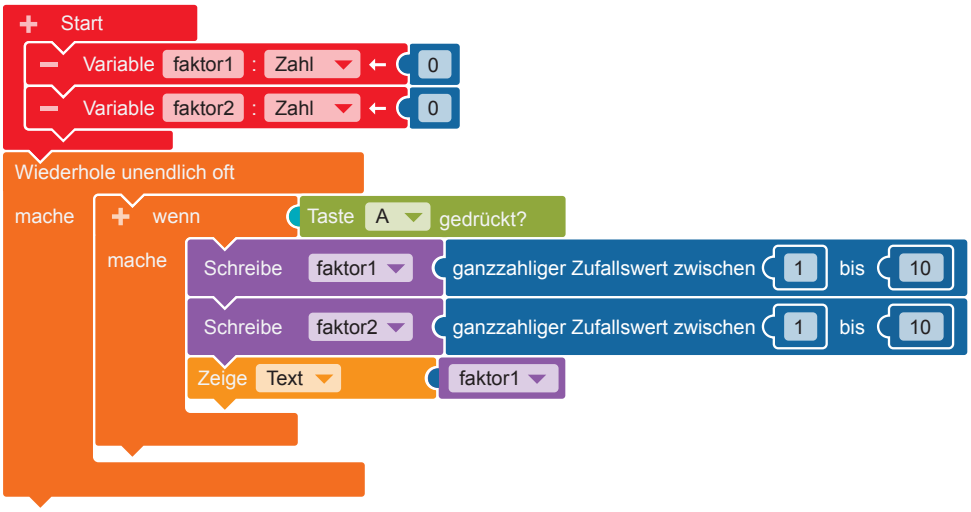
2 Wenn die Taste A gedrückt wird (*wenn*), soll eine per **Zufall** generierte Multiplikationsaufgabe angezeigt werden (*mache*):
Wie in Code 1 beschrieben, werden auch hier eine **Schleife** und eine **Verzweigung** benötigt:
Gehen Sie wie in Code 1 (Schritte 1–3) beschrieben vor.



3 Der Wert der Variablen muss festgelegt werden:
Wählen Sie aus der Kategorie **Variablen** den Block „Schreibe *faktor1*“ aus und fügen Sie ihn in die Verzweigung ein. Fügen Sie daran aus der Kategorie **Mathematik** den Block „ganzzahliger Zufallswert zwischen 1 bis 100“ und ersetzen Sie den Wert 100 durch die Zahl 10. Verfahren Sie ebenso mit dem *faktor2*.

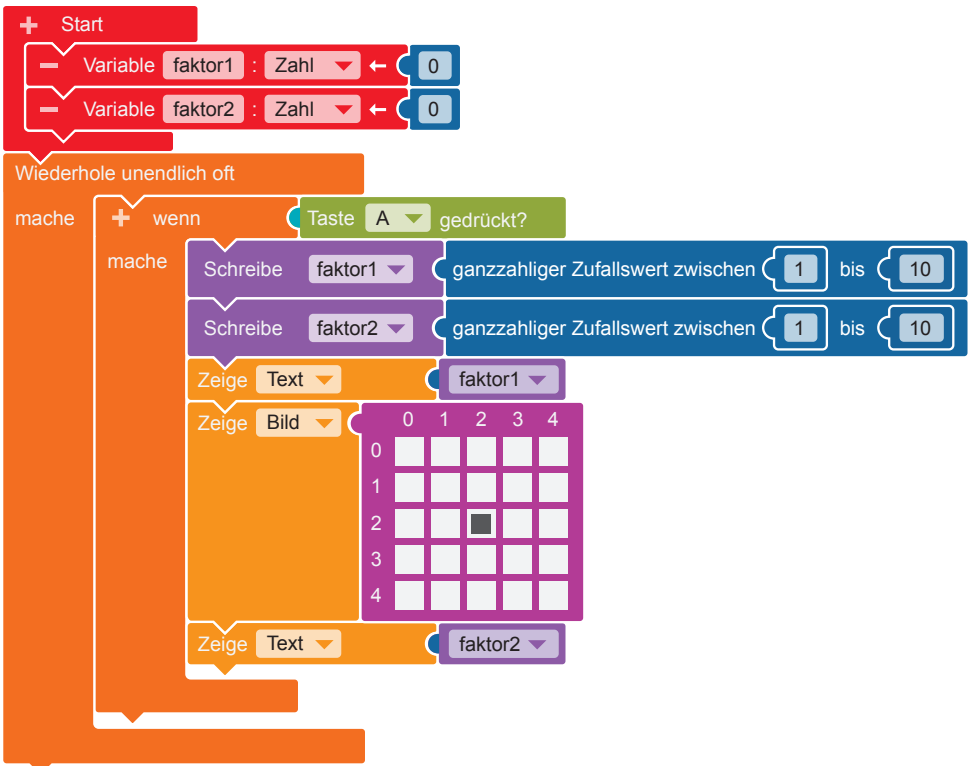


4 Der 1. Faktor (*faktor1*) soll auf dem **LED-Bildschirm** angezeigt werden:
Wählen Sie aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Text“ und ersetzen Sie hier den Text „Hallo“ durch die erste **Variable** . Wählen Sie hierzu aus der Kategorie **Variablen** den Block *faktor1*.



5 Das Multiplikationszeichen soll auf dem LED-Bildschirm angezeigt werden:
Wählen Sie aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Bild“ und markieren Sie das Kästchen in der Mitte.

Der 2. Faktor (*faktor2*) soll auf dem LED-Bildschirm angezeigt werden:
Gehen Sie wie in Schritt 4 vor. Wählen Sie aus der Kategorie **Variablen** den Block „*faktor2*“.

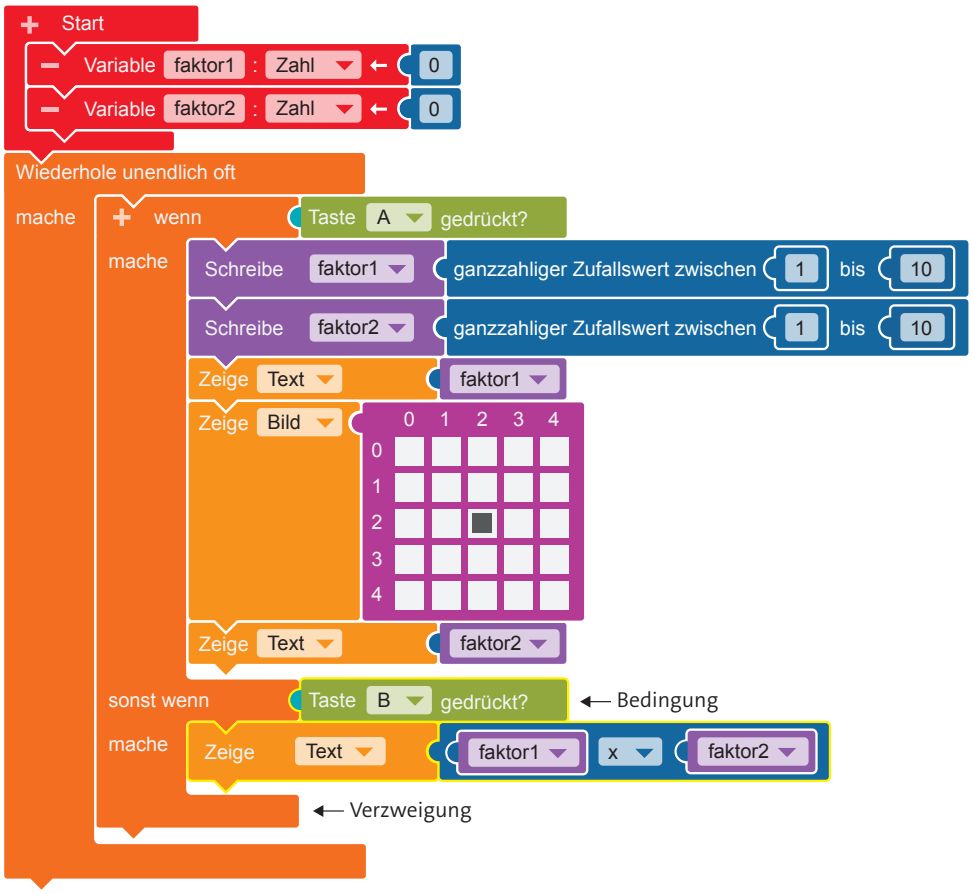


6 Wenn die Taste B gedrückt wird (*wenn*), soll das richtige Ergebnis der per Zufall generierten Aufgabe angezeigt werden (*mache*): Klicken Sie in der Verzweigung auf das „+“ neben dem „wenn“. Eine weitere Verzweigung erscheint. Wählen Sie aus der Kategorie **Sensoren** den Block „Taste B gedrückt?“ und fügen Sie ihn als Bedingung an die Verzweigung an.

Das Programm benötigt den **Befehl**, dass das Produkt der per Zufall generierten Faktoren auf dem LED-Bildschirm angezeigt wird:

Wählen Sie aus der Kategorie **Aktion ▶ Anzeige** „Zeige Text“ und ergänzen Sie hier aus der Kategorie **Mathematik** den Block  .

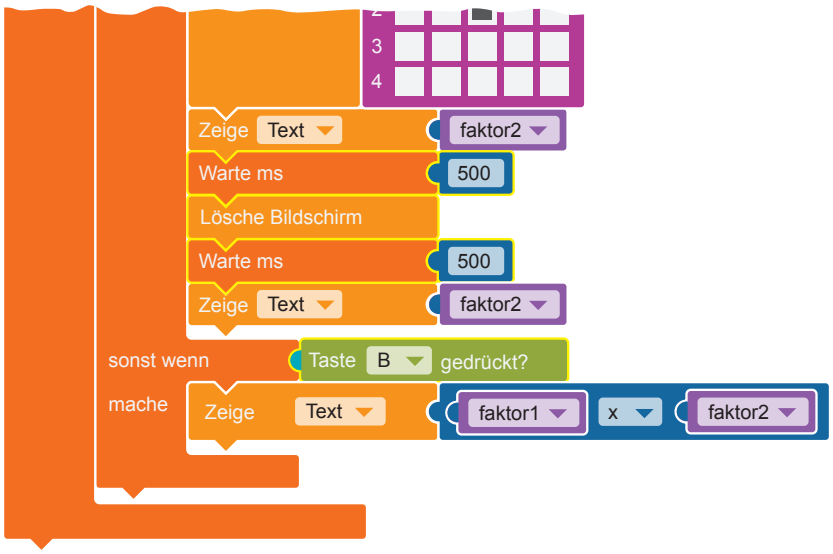
Nun müssen die beiden Variablen (hier: Faktoren) eingefügt werden: Wählen Sie aus der Kategorie **Variablen** die Blöcke *faktor1* und *faktor2*. Klicken Sie auf das Zeichen „+“ und wählen im Auswahlmenü das Multiplikationszeichen „×“ aus.



7 Wie in Code 1 ist die Multiplikationsaufgabe zunächst nicht gut lesbar. Fügen Sie an den entsprechenden Stellen **Pausen** ein und/oder löschen den LED-Bildschirm:

Kontrolle ▶ Warten
▶ „Warte ms“

Aktion ▶ Anzeige
▶ „Lösche Bildschirm“.



8 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

- Dieser Code eignet sich gut für die Partnerarbeit. Die Schülerinnen und Schüler können sich gegenseitig abfragen oder zu Hause und unterwegs mit ihrem selbst programmierten Kopfrechentrainer üben.
- Hinweis zur Handhabung: Wenn Sie als Sensoren die Touch-Pins ausgewählt haben, gehen Sie beim Ausführen mit dem *Calliope mini* wie folgt vor: Mit einer Hand halten Sie den Minus-Pin fest, mit einem Finger der anderen Hand berühren Sie den betreffenden Pin.

Erweiterungsmöglichkeiten

- Programmieren weiterer Rechenoperationen: Für den Einsatz in der Grundschule bietet sich eine Modifizierung des Programms zum Additionstrainers an. Die Subtraktion und die Division gestalten sich etwas komplizierter, da u. a. das Ergebnis ggf. als Minus- oder Kommazahl dargestellt werden muss.
- Rechenaufgaben funken: Die Kategorie Nachrichten bietet die Möglichkeit der Kommunikation mehrerer *Calliope mini*s – Mikrocontroller untereinander. So können Rechenaufgaben von *Calliope mini* zu *Calliope mini* oder gleichzeitig zu verschiedenen Platinen gesendet werden.

Nachbarzahlen bestimmen mit dem *Calliope mini*

Die Übung

Der *Calliope mini* wird zu einem Rechner programmiert, der nach Aufforderung eine Zufallszahl ausgibt und per Tastendruck deren beide Nachbarzahlen (Vorgänger und Nachfolger) anzeigt.

Fachbezug

Mathematik

Die Schülerinnen und Schüler überlegen, nach welcher Rechenvorschrift Nachbarzahlen ermittelt werden und wie diese Aufgabe programmiertechnisch umgesetzt werden kann. Mit dem selbst programmierten Rechner können sie schließlich die Ermittlung des Vorgängers und Nachfolgers von Zufallszahlen unter Selbstkontrolle üben.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Zahlen und Operationen* Zahleigenschaften und Zahlbeziehungen erkennen, beschreiben und darstellen sowie Gesetzmäßigkeiten in arithmetischen Mustern erkennen, beschreiben und fortsetzen, indem sie einen *Calliope-mini*-Nachbarzahlen-Trainer korrekt programmieren.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Argumentieren* mathematische Zusammenhänge erkennen und beschreiben sowie eigene Denk- und Lösungswege begründen.

Anforderungen

Programmierschwerpunkte:

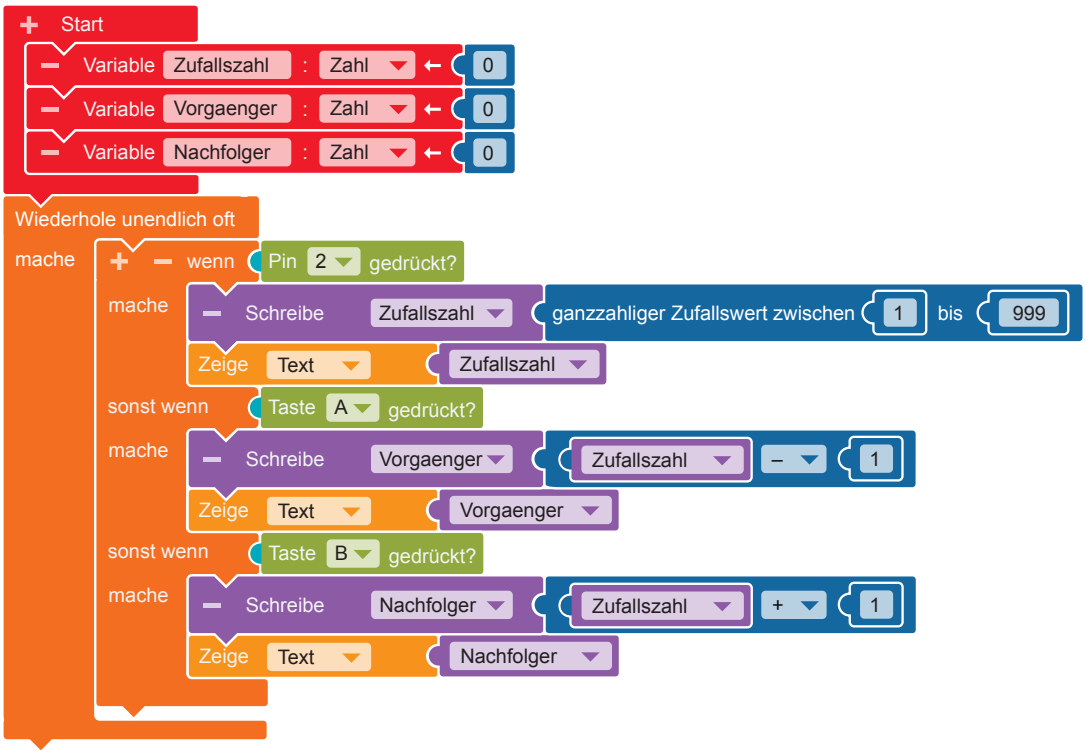
- Eingabe über Touch-Pins und Tasten, Ausgabe über LED-Bildschirm
- Strukturen: bedingte Anweisung, Endlosschleife, Verzweigung
- mit Variablen arbeiten
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle, Mathematik, Variablen

Programmierschwierigkeit:

- mittleres Niveau
- *NEPO*®-Level: Experte

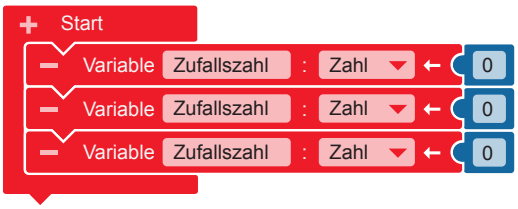
Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird dieser Schritt für Schritt erarbeitet.

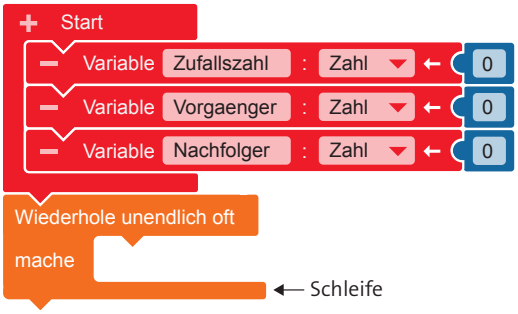


Schritte zur Erstellung des Programms zur Ausgabe von Nachbarzahlen

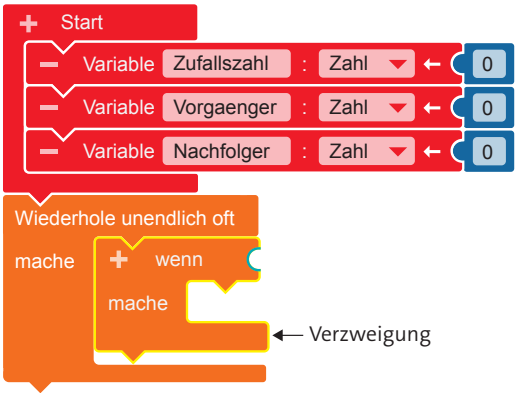
1 Damit beim Üben mit dem Nachbarzahlen-Trainer immer wieder neue Zahlen per Zufall ausgegeben werden können, müssen zunächst verschiedene Variablen angelegt und definiert werden:
Um eine neue **Variable** anzulegen, klicken Sie auf das „+“ links neben „Start“. Ein neuer Block erscheint. Legen Sie darin links den Namen (z. B. *Zufallszahl*, *Vorgaenger*, *Nachfolger*) und rechts den Datentyp (Zahl) der Variablen fest.



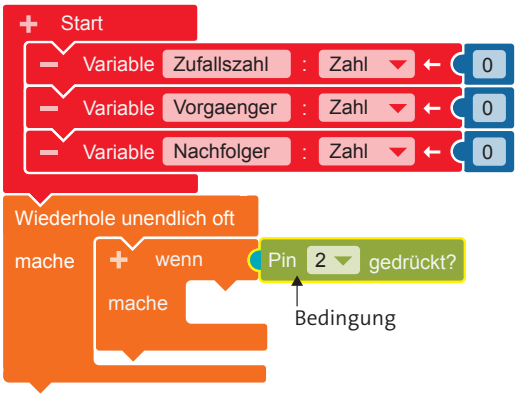
2 Die Zufallszahlen sollen unbegrenzt ausgegeben werden:
Hierfür benötigen Sie eine Endlos**schleife**. Wählen Sie aus der Kategorie Kontrolle ▶ Schleifen den Block „Wiederhole unendlich oft/mache“ und fügen Sie ihn an den „Start“-Block an.



3 Wenn Pin 2 gedrückt wird (*wenn*), soll eine Zufallszahl ausgegeben werden (*mache*):
Um diese **Bedingung** aufzustellen, benötigen Sie eine **Verzweigung**. Wählen Sie aus der Kategorie Kontrolle ▶ Entscheidungen den Block „wenn/mache“ und fügen ihn Sie in die Schleife ein.

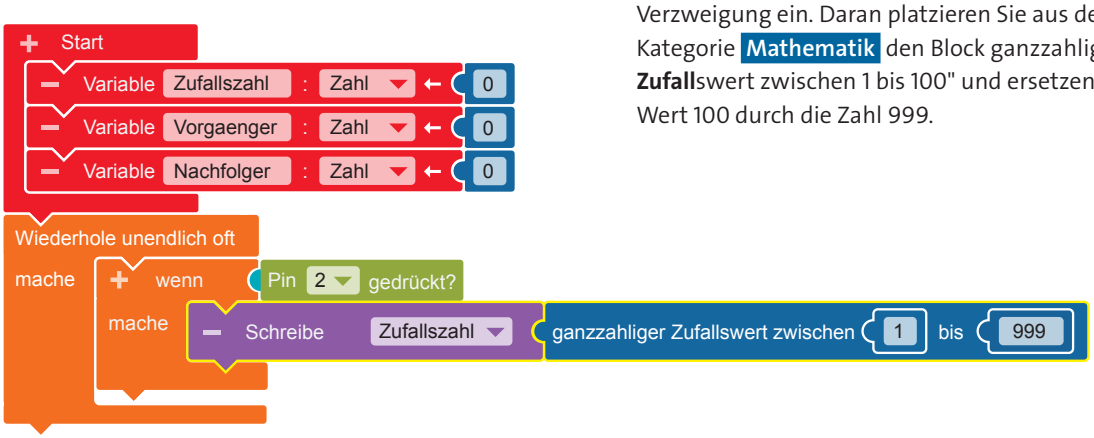


4 Die gewünschte Eingabemöglichkeit (hier über den Sensor „Pin 2“) wird ausgewählt:
Nehmen Sie aus der Kategorie Sensoren den Block „Pin 2 gedrückt?“ und fügen Sie ihn als Bedingung (blauer Bereich) an die Verzweigung an.



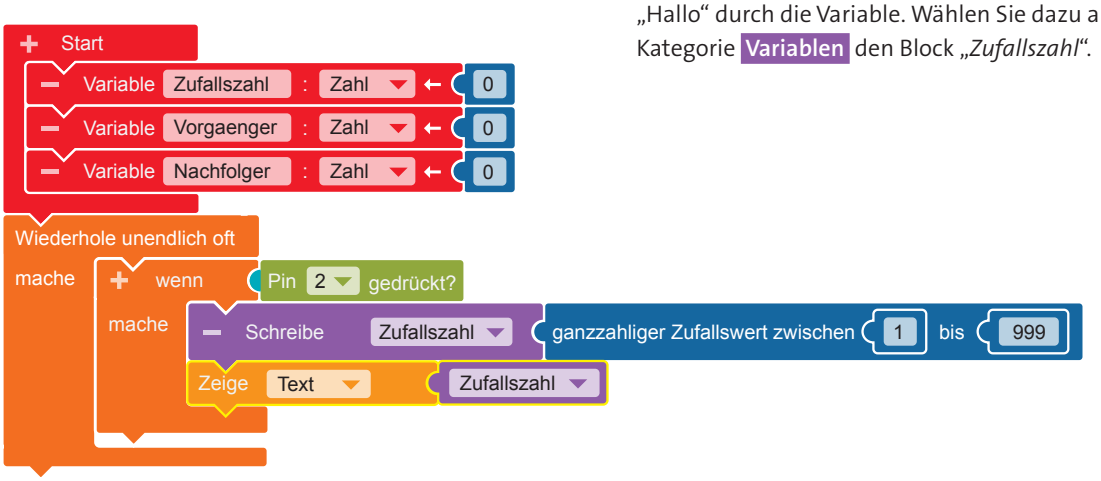
5 Der Wert der Variablen *Zufallszahl* muss definiert werden:

Wählen Sie hierzu aus der Kategorie **Variablen** den Block „Schreibe *Zufallszahl*“ und fügen Sie ihn in die Verzweigung ein. Daran platzieren Sie aus der Kategorie **Mathematik** den Block ganzzahliger **Zufall**swert zwischen 1 bis 100" und ersetzen den Wert 100 durch die Zahl 999.



6 Die Zufallszahl soll auf dem LED-Bildschirm angezeigt werden:

Wählen Sie dazu aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Text“ und ersetzen Sie den Text „Hallo“ durch die Variable. Wählen Sie dazu aus der Kategorie **Variablen** den Block „*Zufallszahl*“.

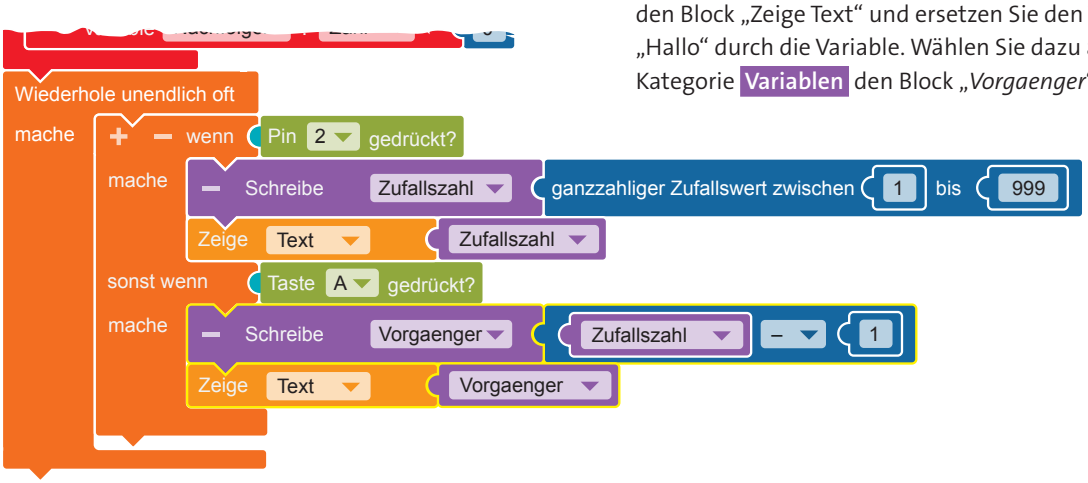


7 Der Vorgänger der ausgegebenen Zufallszahl soll ermittelt werden:
Dafür benötigen Sie eine weitere Verzweigung. Klicken Sie in der Verzweigung auf das „+“ neben dem „wenn“.

Die gewünschte Eingabemöglichkeit (hier über den Sensor „Taste A“) wird ausgewählt:
Wählen Sie aus der Kategorie **Sensoren** den Block „Taste A gedrückt?“ und fügen Sie ihn als Bedingung an die Verzweigung an.

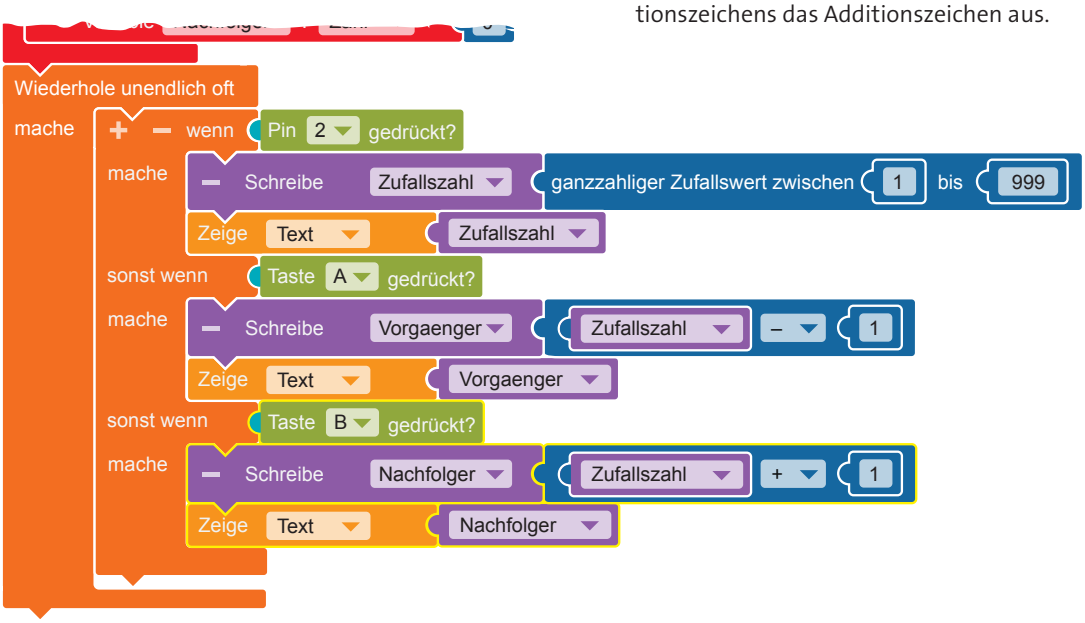
8 Der Wert der Variablen *Vorgaenger* muss definiert werden (Zufallszahl – 1 = Vorgänger):
Wählen Sie aus der Kategorie **Variablen** den Block „Schreibe Vorgaenger“ und platzieren Sie daran aus der Kategorie **Mathematik** den Block  .
Wählen Sie aus der Kategorie **Variablen** den Block „*Zufallszahl*“ und ziehen Sie die Variable in die linke Lücke des Mathematik-Blocks.

Wählen Sie aus der Kategorie **Mathematik** den Block „0“ und setzen Sie ihn in die rechte Lücke. Ersetzen Sie hier die Null durch eine 1. Klicken Sie auf das Zeichen „=“ und wählen Sie im **Ausklappmenü** das Subtraktionszeichen „–“ aus.
Der Vorgänger soll auf dem LED-Bildschirm angezeigt werden:
Wählen Sie dazu aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Text“ und ersetzen Sie den Text „Hallo“ durch die Variable. Wählen Sie dazu aus der Kategorie **Variablen** den Block „*Vorgaenger*“.



9 Der Nachfolger der ausgegebenen Zufallszahl soll ermittelt und auf dem LED-Bildschirm dargestellt werden (Zufallszahl + 1 = Nachfolger):

Gehen Sie wie im Schritt 8 vor. Ersetzen Sie lediglich die Variablen (*Vorgaenger* durch *Nachfolger*) und wählen Sie im Auswahlmenü anstelle des Subtraktionszeichens das Additionszeichen aus.



12 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

Neben dem Üben mit Selbstkontrolle, können sich die Schülerinnen und Schüler auch paarweise abfragen.

Erweiterungsmöglichkeiten

- Weitere Summanden und Minuenden einsetzen (z. B. + 3, – 3, + 10, etc.).
- Verdoppeln und Halbieren (× 2, ÷ 2)

Der *Calliope mini* und das Nim-Spiel

Die Übung

Der *Calliope mini* soll in diesem Programmierbeispiel ein einfaches mathematisches Spiel, das Nim-Spiel, umsetzen. Beim Nim-Spiel werden von einer Anfangsmenge (im Ursprungsspiel Hölzchen, Steine oder Spielfiguren) abwechselnd ein, zwei oder drei Elemente weggenommen. Der Spieler, der das letzte Element nimmt, gewinnt. Das Spiel wird für zwei Spieler programmiert, wobei die Rolle des zweiten Spielers der *Calliope mini* übernimmt. Der zweite Spieler (d. h. der *Calliope mini*) agiert nicht taktisch, sondern nimmt jeweils eine zufällige Anzahl (ein, zwei oder drei) an Elementen weg. Die verbliebene Zahl der Elemente wird nach jedem Spielzug auf dem LED-Bildschirm angezeigt. Wenn eine Eingabe des ersten Spielers erwartet wird, leuchtet die RGB-LED gelb, ist der *Calliope mini* am Zug, leuchtet die RGB-LED blau. Die Anzahl wegzunehmender Elemente bestimmt der erste Spieler durch ein wiederholtes Drücken der Taste A. Mit Druck auf die Taste B wird ein Spielzug abgeschlossen.
Hat der *Calliope mini* gewonnen und der menschliche Spieler verloren, wird ein trauriges Smiley angezeigt. Gewinnt jedoch der menschliche Spieler, erscheint ein lachendes Smiley auf dem LED-Bildschirm.

Fachbezug

Mathematik

Die Schülerinnen und Schüler merken beim Nim-Spiel schnell, dass es kein Glücksspiel ist. Durch genaues Beobachten, Beschreiben und Begründen der verschiedenen Spielverläufe können sie individuell ihre Gewinnstrategie entwickeln. Auf diese Weise trägt das Nim-Spiel zu einem mathematischen Grundverständnis bei.

Verortung im Lehrplan

Inhaltsbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Zahlen und Operationen* Knobelaufgaben durch Probieren lösen (z. B. ungeordnetes und systematisches Probieren), indem sie das Nim-Spiel mit dem *Calliope mini* taktisch geschickt spielen.

Prozessbezogene Kompetenzen

Die Schülerinnen und Schüler können im Bereich *Problemlösen* Lösungsstrategien entwickeln, Lösungsstrategien nutzen (z. B. systematisches Probieren) sowie Zusammenhänge erkennen und nutzen.

Anforderungen

Programmierschwerpunkte:

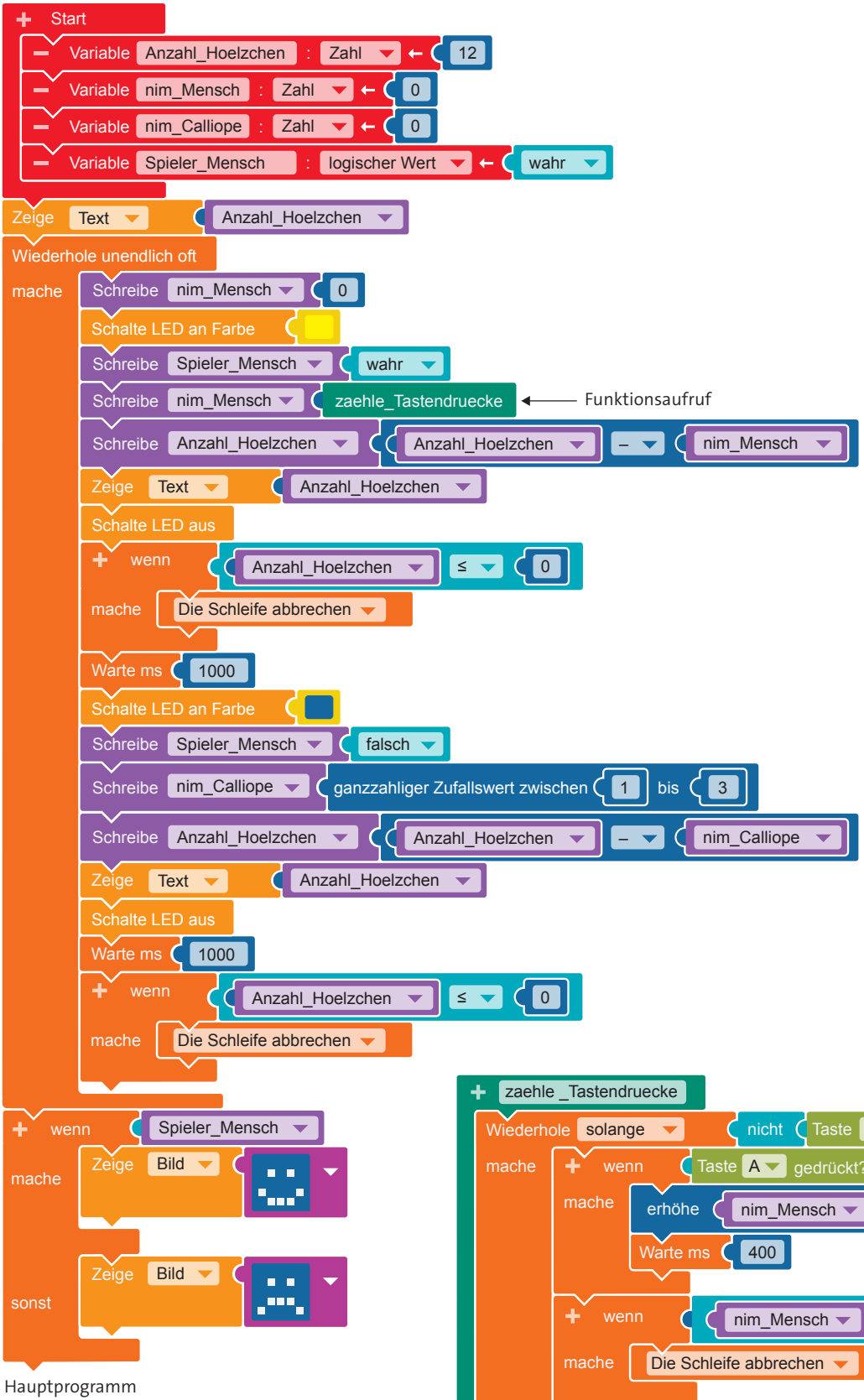
- Eingabe über Tasten, Ausgabe über LED-Bildschirm und RGB-LED
- Strukturen: bedingte Schleife, Endlosschleife, Verzweigungen und Bedingungen
- mit logischen Variablen umgehen
- Funktionen (Tastendrücke zählen)
- verwendete *NEPO*®-Kategorien: Aktion, Sensoren, Kontrolle, Logik, Mathematik, Variablen, Funktionen

Programmierschwierigkeit:

- fortgeschrittenes Niveau
- *NEPO*®-Level: Experte

Der Code

So sieht der Code des gesamten Programms aus. Nachfolgend wird er Schritt für Schritt erarbeitet.

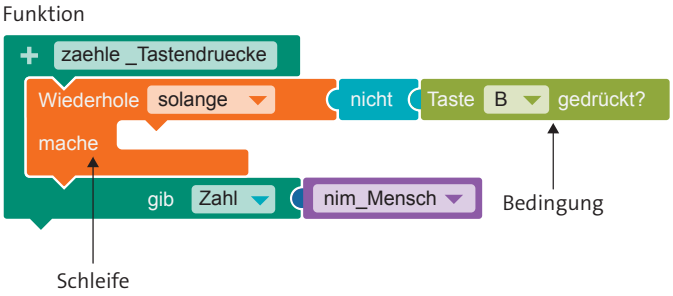


Schritte zur Erstellung des Programms für das Nim-Spiel

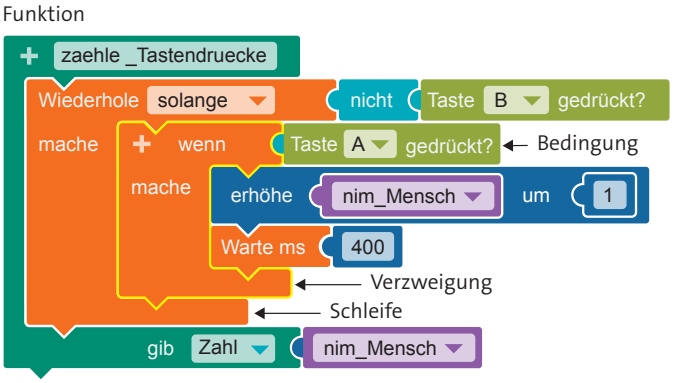
1 Für den Spielverlauf muss die Anfangsmenge der Hölzchen (a), die vom menschlichen Spieler weggenommenen Hölzchen (b), die vom Programm weggenommenen Hölzchen (c), sowie der aktuelle Spieler (d) in jeweils einer **Variablen** gespeichert werden:
Dazu werden vier Variablen erstellt, drei davon sind vom **Datentyp** „Zahl“ mit den Namen

(a) *„Anzahl_Hoelzchen“*. Klicken Sie auf die „0“ im blauen Feld und tragen Sie den Wert „12“ ein.
(b) *„nim_Mensch“*
(c) *„nim_Calliope“*
(d) Außerdem benötigt man noch eine Variablen *„Spieler_Mensch“* (d) vom Datentyp „logischer Wert“. Diese kann die beiden Werte *wahr* (Mensch ist am Zug) oder *falsch* (*Calliope mini* ist am Zug) annehmen.

2 Da es nicht so einfach ist zu zählen, wie oft der menschliche Spieler die Taste A hintereinander gedrückt hat, wird diese Aufgabe in eine **Funktion** ausgelagert. Dadurch bleibt das Hauptprogramm übersichtlicher:
Fügen Sie aus der Kategorie **Funktionen** den Block „mache Etwas ... gib Zahl“ neben dem Hauptprogramm auf die freie Fläche ein und nennen Sie diese Funktion *zaehle_Tastendruecke* (Eingabe über die Tastatur).
In der Kategorie **Funktionen** wird dadurch ein neuer Block mit dem soeben eingegebenen Namen angelegt.
Fügen Sie in diese Funktion aus der Kategorie **Kontrolle ▶ Schleifen** den Block „Wiederhole solange/mache“ ein.
Die **Bedingung** der **Schleife** legen Sie aus den Kategorien **Logik** mit dem Block „nicht“ und **Sensoren** mit dem Block „Taste B gedrückt?“ fest.

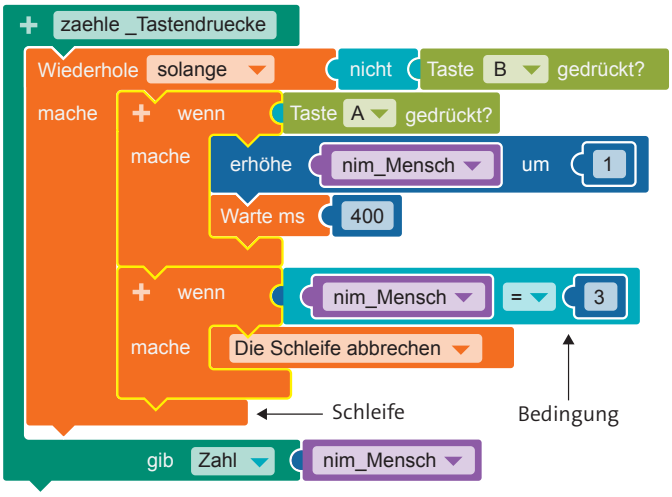


3 Fügen Sie in diese **Schleife** eine **Verzweigung** ein.
Wählen Sie dafür aus der Kategorie **Kontrolle ▶ Entscheidungen** den Block „wenn/mache“ aus, an den Sie aus der Kategorie **Sensoren** den Block „Taste A gedrückt?“ als Bedingung andocken.
Aus der Kategorie **Mathematik** wird nun der Block erhöhe um eingefügt. Die Platzhalter werden mit dem Block *„nim_Mensch“* aus der Kategorie **Variablen** und dem Zahlenwert „1“ aus der Kategorie **Mathematik** gefüllt. Somit werden die Tastendrücke gezählt.
Ergänzen Sie aus der Kategorie **Kontrolle ▶ Warten** den Block „Warte ms (400)“.
Dieser dient dazu, einen längeren Tastendruck nicht als doppelten Tastendruck zu zählen.
Die Funktion gibt den Wert der **Variablen** *nim_Mensch* an das Hauptprogramm zurück.



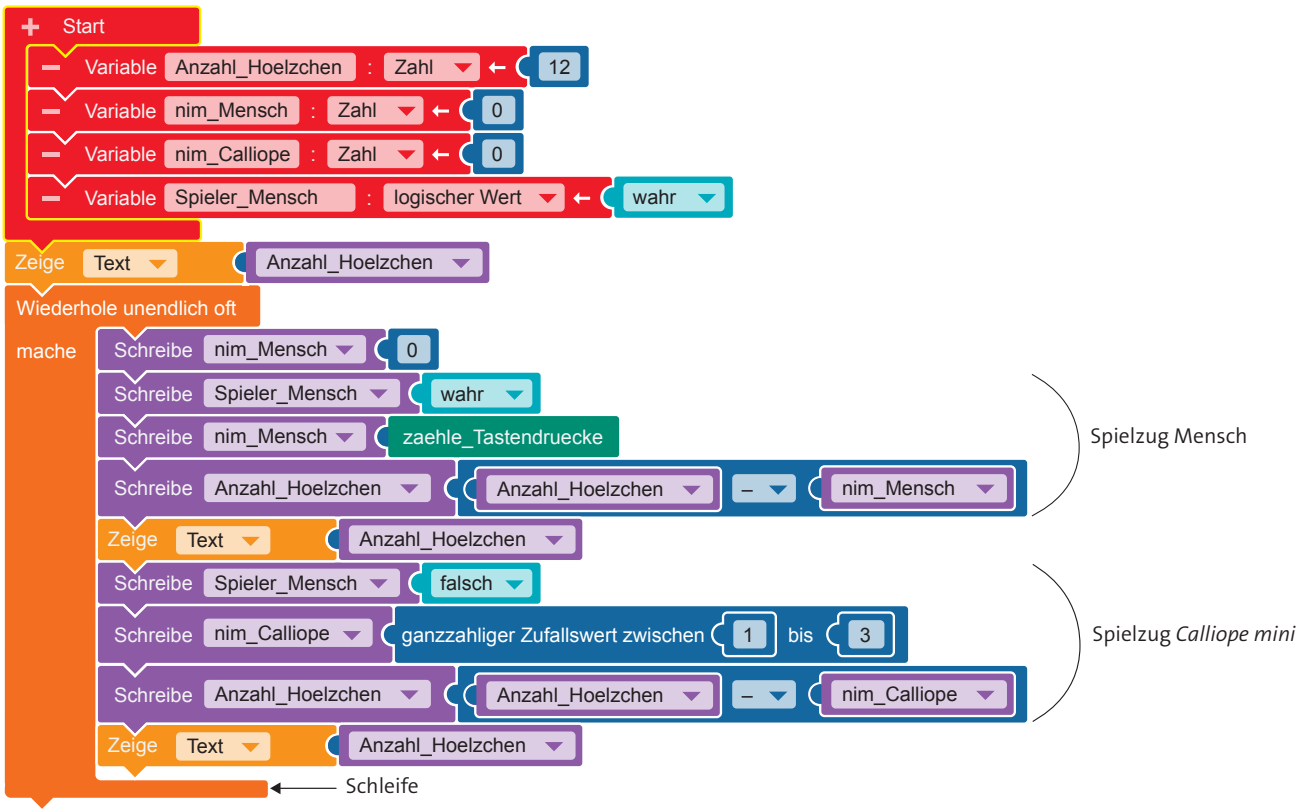
4 Damit nicht mehr als drei Tastendrücke gewertet werden, wird noch eine zweite Verzweigung aus der Kategorie **Kontrolle ▶ Entscheidungen** mit dem Block „wenn/mache“ eingefügt.
Die Bedingung besteht aus dem Gleichheitsblock aus der Kategorie **Logik**. Eingesetzt wird die Variable *nim_Mensch* und aus der Kategorie **Mathematik** die Zahl „3“.
Ist diese Bedingung erfüllt, wird die Schleife „Wiederhole solange nicht Taste B gedrückt?“ durch den Block „Die Schleife abbrechen“ aus der Kategorie **Kontrolle ▶ Schleifen** beendet und die Funktion ebenfalls verlassen.

Funktion



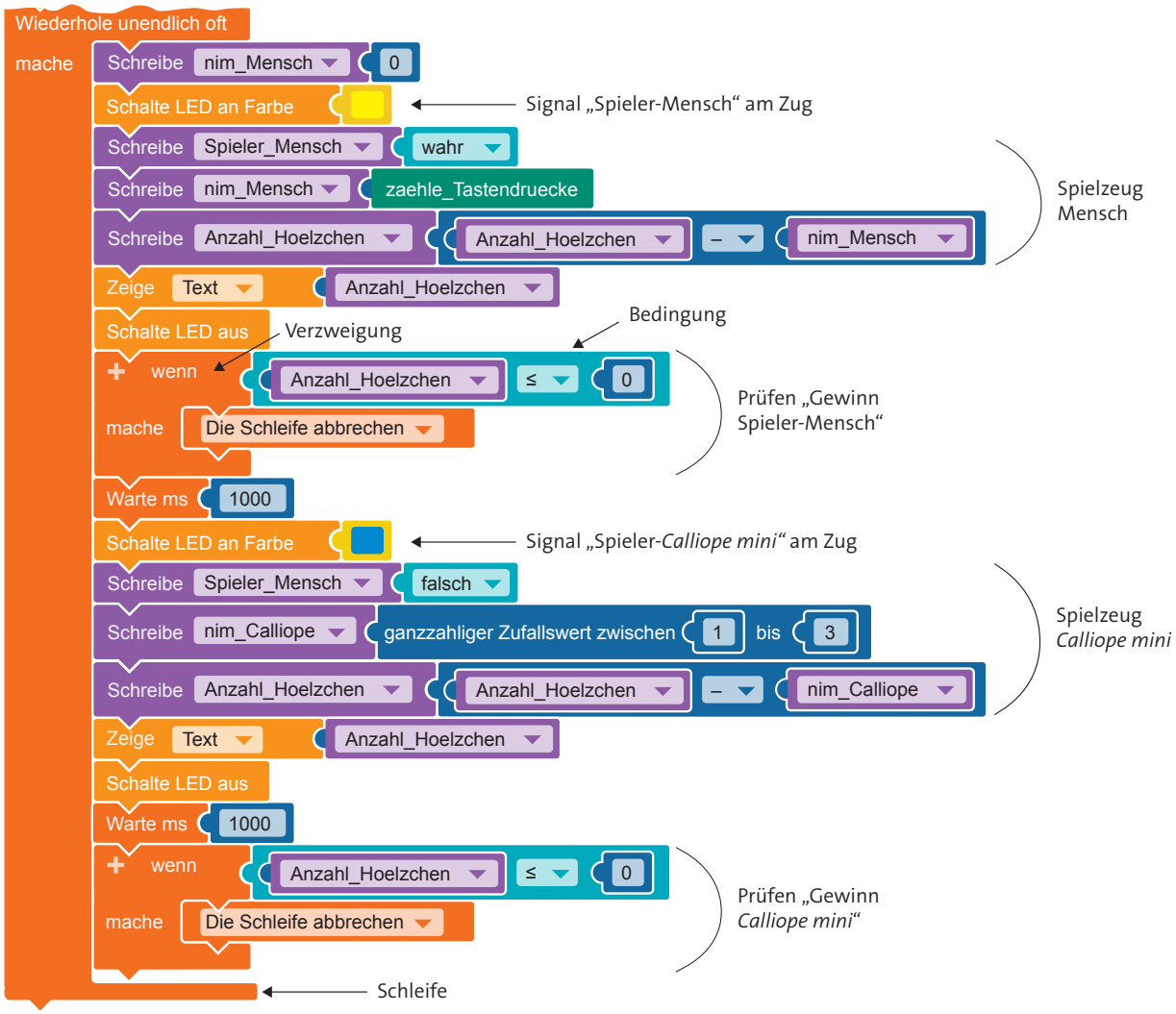
5 Im Hauptprogramm wechseln sich der menschliche Spieler und der *Calliope mini* jeweils ab. Deshalb müssen die Befehle für die jeweiligen Spielzüge aufeinander folgen:
Aus der Kategorie **Variablen** wird sieben mal der „Schreibe“-Block verwendet.
Die Variable *nim_Mensch* muss vor jedem Zug wieder auf den Wert „0“ zurückgesetzt werden – ein neuer Zug steht an.
Die **Variable** *Spieler_Mensch* bekommt aus der Kategorie **Logik** den Wert „wahr“ – der Spieler-Mensch ist am Zug und die **Variable** *nim_Mensch* bekommt aus der Kategorie **Funktion** den Wert der Funktion *zaehle_Tastendruecke* – es dürfen nicht mehr als drei Hölzchen weggenommen werden.

Nach dem abgeschlossenen Zug wird durch Subtraktion aus der Kategorie **Mathematik** die Anzahl der Hölzchen entsprechend reduziert.
Verfahren Sie für den *Calliope mini*-Spielzug ebenso. Der einzige Unterschied besteht darin, die Variable *nim_Calliope* nicht auf „0“ zurückzusetzen, da sie jedes Mal durch den Block „ganzzahliger Zufallswert zwischen 1 bis 3“ aus der Kategorie **Mathematik** überschrieben wird.
An drei Stellen im Spielverlauf (am Anfang und nach jedem Spielzug einer der beiden Spieler) soll der Wert der Variablen *Anzahl_Hoelzchen* als Text auf dem LED-Bildschirm erscheinen. Wählen Sie dafür aus der Kategorie **Aktion ▶ Anzeige** den Block „Zeige Text“ und fügen sie ihn wie unten abgebildet ein.

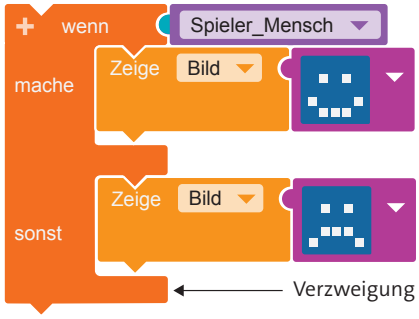


6 Der *Calliope mini* soll feststellen und ausgeben, wer gewonnen hat:
Wenn die Anzahl der Hölzchen gleich null ist, hat der Spieler gewonnen, der gerade am Zug ist. Die Endlos-**schleife** wird in diesem Fall abgebrochen.
Fügen Sie nach dem Spielzug-Mensch bzw. Spielzug-*Calliope mini* aus der Kategorie **Kontrolle ▶ Entscheidungen** den Block „wenn/mache“ ein. Die Bedingung fügen Sie aus der Kategorie **Logik** „≤“ ein und vergleichen die Variable *Anzahl_Hoelzchen* mit dem Zahlenwert „0“ aus der Kategorie **Mathematik**. Den Block „Schleife abbrechen“ finden sie in der Kategorie **Kontrolle ▶ Schleifen**.

Hilfreich ist es, jedem der Spieler eine Farbe der RGB-LED zuzuordnen, die immer leuchtet, während dieser am Zug ist. Dazu benötigen Sie jeweils einen Block zum An- und Ausschalten aus der Kategorie **Aktion ▶ Statusleuchte**. Da der *Calliope mini* sehr schnell rechnet, bauen Sie eine künstliche Verzögerung von einer Sekunde (= 1000 Millisekunden) aus der Kategorie **Aktion ▶ Warten** bei jedem Spieler ein. Die Zahl „1000“ finden Sie in der Kategorie **Mathematik** im Block „0“ und ändern den Wert auf „1000“.



7 Am Spielende soll durch Anzeige verschiedener Symbole auf dem LED-Bildschirm verdeutlicht werden, welcher Spieler gewonnen hat:
Dazu wird eine Verzweigung aus der Kategorie **Kontrolle ▶ Entscheidungen** mit dem Block „wenn/mache/sonst“ benötigt, die den Wert der **Variablen** *Spieler_Mensch* nach Beendigung der Spielschleife prüft. Wenn dieser Wert wahr ist, hat der menschliche Spieler gewonnen.
Über das Bild eines lachenden Smileys aus der Kategorie **Aktion ▶ Anzeige** und der entsprechenden Wahl über das **Ausklappmenü** wird dieses angezeigt.
Setzen Sie ein trauriges Smiley für den Fall ein, dass bei Spielende der *Calliope mini* am Zug war, also die **Variable** *Spieler_Mensch* den Wert „falsch“ besitzt.



8 Übertragen Sie den Programmcode auf Ihren *Calliope mini*, indem Sie das Programm herunterladen und auf dem *Calliope mini* speichern (http://calliope.cc/anleitungen/anleitung-2).

Hinweise und Infos

Einsatzszenarien im Unterricht

- Finden von Gewinnstrategien, z. B. der Anzahl zu nehmender Hölzchen in Spielzügen, die einen Gewinn ermöglichen.
- Unterschied zwischen Zufall (Glück) und Strategie thematisieren.
- Fragestellungen zur Diskussion: Ist ein Computer intelligent? Wenn er es nicht ist, warum kann er trotzdem in die Rolle eines Spielers schlüpfen?
- Die Bedeutung von Spielregeln in verschiedenen Zusammenhängen vergleichen (z. B. Sport, Brettspiele).

Erweiterungsmöglichkeiten

- Veränderung der Spielregeln, z. B. es dürfen nur ein oder zwei Hölzchen weggenommen werden. Untersuchen, was sich im Programm oder an der Spielstrategie ändert.
- Änderung der Hölzchenzahl zu Spielbeginn.
- Man kann den vom *Calliope mini* simulierten Spieler intelligenter machen, z. B. indem man durch weitere Verzweigungen versucht, bestimmte Hölzchenanzahlen zu vermeiden oder andere zu erreichen.

Glossar

Anweisung Die Blöcke im *NEPO*®-Editor, die oben links eine dreieckige Einkerbung und unten links eine dreieckige Erweiterung haben, sind Anweisungs-, bzw. Befehlsblöcke.
Gleichbedeutend ist der Begriff „Instruktion".
Programme setzen sich aus Anweisungsfolgen (Sequenzen) und Kontrollstrukturen (z. B. Schleifen, Verzweigungen) zusammen.

Ausgabe Als Ausgabe werden alle Aktionen des *Calliope mini* verstanden, die man hören oder sehen kann. Als Ausgabe können Töne mit dem Lautsprecher erzeugt, Bilder oder Texte auf dem LED-Bildschirm angezeigt werden oder die RGB-LED kann in einer Farbe leuchten.
Nicht sichtbare Ausgaben des *Calliope mini* sind Nachrichtenübertragungen per Funk, Motorsteuerungssignale oder elektrische Spannungen an den Pins.

Ausklappmenü Einige Programmblöcke haben ein Ausklappmenü, z. B. bei Sensoren ▶ „Taste A gedrückt?". Dort lässt sich nach der Auswahl des Blockes mit einem Klick auf den Auswahlbereich auch die Taste B auswählen. Die aktuelle Auswahl wird jeweils durch einen Haken angezeigt.

Bedingung Zur Festlegung einer Bedingung wird ein Vergleichsblock aus der Kategorie Logik benötigt. Links und rechts des Vergleichsoperators müssen Variablen oder konstante Werte (z. B. eine Zahl, ein Text) eingesetzt werden. Die beiden verglichenen Werte müssen gleichen Datentyps sein. Das Ergebnis einer Bedingung ist ein Wahrheitswert (wahr oder falsch). Bedingungen können beliebig mittels Logik ▶ „und/oder" kombiniert oder durch Logik ▶ „nicht" negiert werden.

Befehl siehe Anweisung

Block Ein Block ist ein grafisches Programmelement des *NEPO*®-Editors. Diese Blöcke finden sich in den Kategorien. Sie werden z. B. mit der Maus ausgewählt und zusammengesetzt.

Code Ein Programm besteht im *NEPO*®-Editor aus mindestens einem Hauptprogramm (mit „Start" beginnend) und beliebig vielen Funktionen. Das Wort „code" kann als englischsprachige Kurzform des Begriffes „programming code" aufgefasst und in das deutsche Wort „Programm" übersetzt werden. Kodierung bedeutet hier, dass jede Programmiersprache ihre eigenen fest kodierten Befehle hat.
Ein Programm hat im *NEPO*®-Editor zwei Darstellungsformen: (1) Die bearbeitbaren farbigen Programmblöcke und (2) den nicht veränderbaren Quellcode (einsehbar über das Symbol rechts oben im Editorfenster „< >"). Dieser entspricht exakt den Programmblöcken.

Datentyp Dies ist der Wertebereich, den eine Variable haben kann. Für einen Datentyp gibt es jeweils definierte Operationen, wie z. B. die arithmetischen Operationen für den Datentyp Zahl.
Beispiele für elementare Datentypen sind: ganze Zahlen, Buchstaben, logische Werte.
Beispiele für zusammengesetzte Datentypen sind: Zeichenketten, Listen.

Editor Für den *Calliope mini* gibt es drei Editoren (*Calliope mini* Editor, PXT, *NEPO*® im *Open Roberta Lab*®). Mithilfe eines Editors kann ein Programm für den *Calliope mini* erstellt oder verändert werden. Die Programmerstellung kann entweder mittels grafischer Programmblöcke oder textbasiert erfolgen.

Eingabe Als Eingabe wird alles das bezeichnet, was von den Sensoren und Tasten des *Calliope mini* in einem Programm weiterverarbeitet werden kann.

Funktion Wird eine Anweisungsfolge häufiger benötigt, ist es sinnvoll, diese in einer Funktion auszulagern. Diese Funktion bekommt einen eindeutig zu definierenden Namen und ist unter diesem beliebig oft als Anweisung im Hauptprogramm aufrufbar. Es entsteht ein neuer Programmblock. Eine Funktion kann einen Rückgabewert sowie Parameter beinhalten.

Index Der Index bezeichnet die Position eines Listenelements innerhalb einer Liste. Das erste Element erhält den Index null, für jedes weitere Element wird der Index um eins hochgezählt. Reichen die Indizes z. B. von 0 bis 3, enthält die Liste 4 Elemente.

Kategorie Die Anweisungen, Kontrollstrukturen, Bedingungen, Funktionen sowie die Variablen sind im *NEPO*®-Editor zu inhaltlichen Kategorien zusammengefasst. Die verschiedenen Kategorien sind in unterschiedlichen Farben dargestellt. Die dazugehörigen Blöcke haben jeweils dieselbe Farbe.

LED-Bildschirm Als LED-Bildschirm werden die 25 roten Leuchtdioden auf der Vorderseite des *Calliope mini* bezeichnet, die quadratisch in fünf Zeilen und fünf Spalten angeordnet sind. Auf ihm lassen sich als Standbild einfache Bilder, sowie Buchstaben und Zahlen sowie in Laufschrift Text darstellen (Kategorie Aktion ► Anzeige).

Liste Wenn in einem Programm mehrere Variablen gleichen Datentyps und gleicher Bedeutung verwendet werden, empfiehlt sich der sequenzielle Datentyp Liste, der beliebig viele Behälter eines festzulegenden Datentyps (Zahl, logischer Wert, Zeichenkette) hat. Im *NEPO*®-Editor werden zunächst drei Listenelemente angezeigt. Durch klicken auf die „+“ oder „–“ Symbole kann deren Anzahl erhöht oder verringert werden.
Es gibt spezielle Funktionen für Listen, so z. B. das Suchen nach einem Listenelement.

Pausen Da der *Calliope mini* wie jeder Computer sehr viel schneller rechnen und Befehle ausführen kann als ein Mensch, ist es zuweilen sinnvoll, ihn für eine bestimmte Zeitspanne anzuhalten, um Zwischenstände des Programms sichtbar zu machen. Dazu können im *NEPO*®-Editor in der Kategorie Kontrolle ► Warten entsprechende Blöcke gewählt werden.

Pin Mit Pins sind die sechs Ecken des *Calliope mini* gemeint. Die beiden oberen Ecken („+“ und „–“) erlauben den Anschluss einer Spannungsquelle (Batterie), die vier unteren („P0“ bis „P3“) sind berührungsempfindlich (Touch-Pins) (siehe z. B. Beispiel „Minipiano“). An P0 bis P3 können auch externe Sensoren angeschlossen werden.

Programm siehe Code

RGB-LED Eine RGB-LED ist eine Leuchtdiode (LED), die in mehreren Farben leuchten kann.
Im Kern sind es drei Leuchtdioden, deren Farbanteile (R-Rot, G-Grün, B-Blau) additiv gemischt werden. Die Farbe Weiß entsteht durch gleichzeitiges Leuchten aller drei Farbanteile (Kategorie Aktion ► Statusleuchte).

Schleife (Endlos-/ bedingte/Zähl-) Eine Schleife lässt eine Anweisungsfolge wiederholt ausführen. Die Schleifenbedingung muss erfüllt (wahr) sein, damit die Anweisungsfolge ausgeführt wird. Ist sie nicht erfüllt, wird der Programmablauf hinter der Schleife fortgesetzt.
Eine *Endlosschleife* hat keine Bedingung, sondern statt dessen den konstanten logischen Wert „wahr“.
Eine *Zählschleife* beinhaltet anstelle einer Bedingung eine Zählvorschrift, die angibt, wie oft die Schleife durchlaufen wird.

Sensor Der *Calliope mini* hat verschiedene elektronische Fühler, die Eigenschaften der Umgebung ermitteln können (Kategorie Sensoren).
So können die Orientierung im Erdmagnetfeld (Kompasssensor), die Temperatur, die Helligkeit, die Lage des *Calliope mini* im Raum, Geräusche (Mikrofon), die Berührungen der vier Pins sowie der Tasten A und B ermittelt werden.

Simulation Ist kein *Calliope mini* zur Hand, lässt sich ein im *NEPO*®-Editor geschriebenes Programm durch einen Klick auf die Schaltfläche „SIM“ am rechten oberen Fensterrand simulieren. Ein Fenster mit einem *Calliope mini* öffnet sich und das Programm kann über eine „Play“-Schaltfläche am unteren Fensterrand gestartet werden.

Stelle siehe Index

Taste Im Unterschied zu einem Schalter springt eine Taste nach einem Tastendruck wieder in ihre Ausgangslage zurück. Somit besitzt eine Taste genau zwei Zustände: gedrückt und nicht gedrückt. Der *Calliope mini* hat die Tasten A (blau) und B (rot), die in Programmen abgefragt werden können, sowie die Reset-Taste (weiß).

Touch-Pin siehe Pin

Variable Eine Variable ist ein Behälter für einen bestimmten Wert, der anfangs definiert wird (hierfür klicken Sie beim Startblock auf „+“) und später beliebig oft geändert werden kann. Eine Änderung des Wertes erfolgt über eine Zuweisung im Programm. Der Wert einer Variablen wird für die Zeit der Programmausführung gespeichert.
Jede Variable benötigt einen eindeutigen Namen, der mit einem Buchstaben beginnen muss. Die Werte einer Variablen gehören zu einem bei der Variablendeklaration festzulegenden Datentyp (Zahl, Zeichenkette, logischer Wert etc.).

Vergleich Ein Vergleich ist zumeist Bestandteil einer Bedingung. Die aktuellen Werte zweier Variablen können miteinander verglichen werden. Außer der Gleichheit gibt es die Vergleichsoperatoren „ungleich“, „größer als“, „kleiner als“, „größer oder gleich“ und „kleiner oder gleich“.

Verzweigung Abhängig vom Wahrheitswert einer Bedingung (wahr oder falsch) wird der lineare Ablauf eines Programms in zwei unterschiedliche Programmabschnitte verzweigt (Kategorie Kontrolle ▶ Entscheidungen). Je nachdem, welcher Wahrheitswert vom *Calliope mini* ermittelt wird, verändert sich der Programmablauf. In jeder der beiden Verzweigungen können beliebig viele Anweisungen oder weitere Kontrollstrukturen verwendet werden. Nach einer Verzweigung läuft das Programm wieder linear weiter.

Wahrheitswert siehe Bedingung und Verzweigung

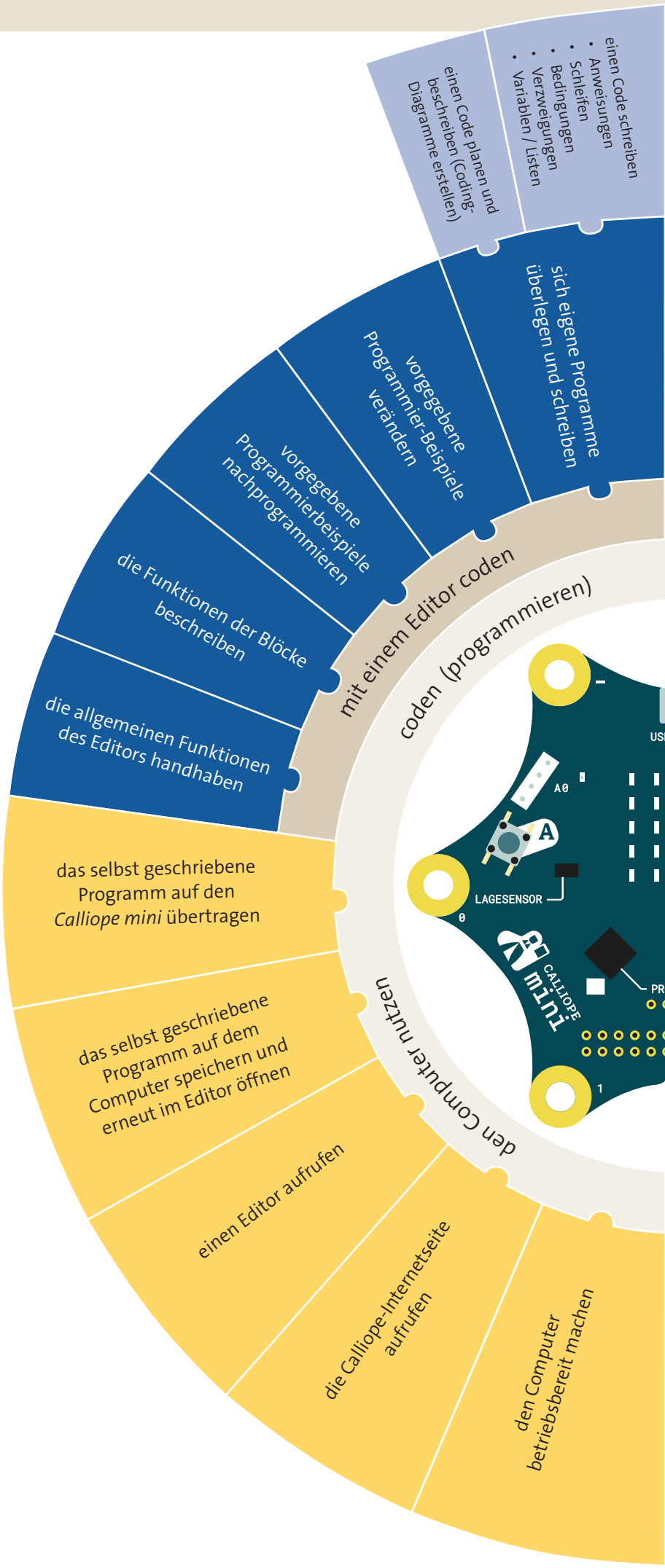
Warten siehe Pausen

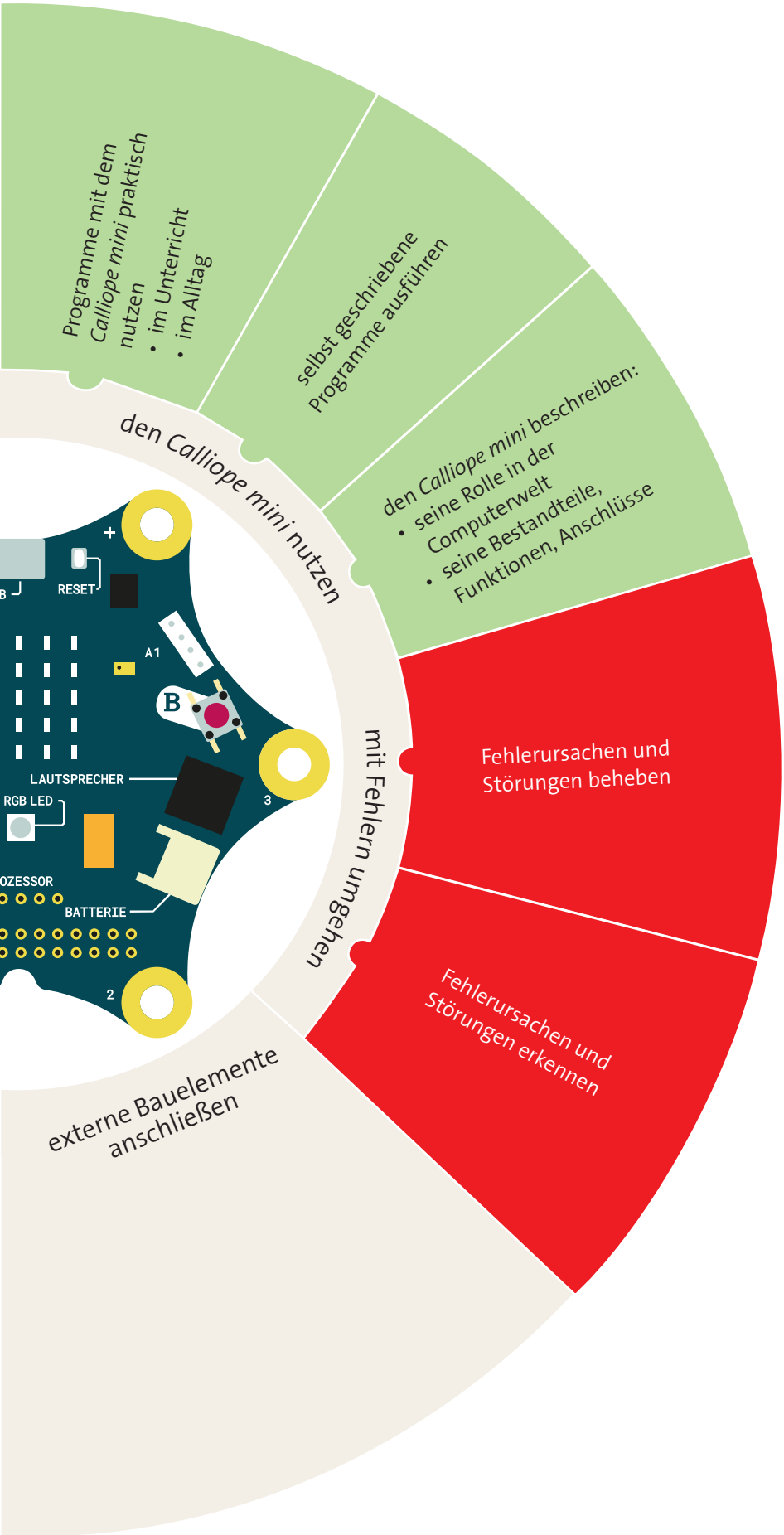
Zeichenkette Eine Zeichenkette ist eine Folge von Buchstaben, Satz- oder Sonderzeichen. Im *NEPO*®-Editor wird sie auch als Text bezeichnet und in Anführungsstriche gestellt. Variablen können u. a. dem Datentyp „Zeichenkette“ zugehören.

Zeitgeber Der Prozessor startet zeitgleich mit dem auf ihn überspielten Programm einen Zeitgeber, der wie eine Stoppuhr mit der Genauigkeit von Millisekunden (Tausendstelsekunden) funktioniert. Der aktuelle Stand dieses Zeitgebers kann beliebig oft über den Block „gib Wert Zeitgeber 1“ in der Kategorie Sensoren abgefragt werden. Auch ist es möglich, diesen Zeitgeber mit dem Block „setze Zeitgeber 1 zurück“ erneut zu starten.

Zufall Ein zufälliger Zahlenwert kann vom Prozessor des *Calliope mini* erzeugt werden. Hierbei kann ein Zahlenbereich, innerhalb dessen die Zufallszahl liegt, vom Programmierer festgelegt werden.

Die vorliegende Lernlandkarte umfasst alle wichtigen Kompetenzen, die benötigt werden, um den Computer und einen Editor selbstständig zu nutzen und den *Calliope mini* zu programmieren. Sie wurde auf der Grundlage vielfältiger Praxiserfahrung entwickelt. Da im Rahmen der offiziellen Richtlinien und Lehrpläne (noch) kein verbindliches Curriculum für das Coden vorliegt, kann die Lernlandkarte keinen Anspruch darauf erheben, dass die Lernenden am Ende einer Unterrichtsreihe alle Kompetenzen vollständig beherrschen. Vielmehr ist sie in Form eines Überblicks und kleinschrittigen Wegweisers als didaktische Richtlinie für ein größeres Coding-Unterrichtsvorhaben zu nutzen.





Die Lernlandkarte ist insgesamt in die drei elementaren Hauptbereiche *den Computer nutzen*, *coden* und *den Calliope mini nutzen* aufgeteilt und wird um die beiden Bereiche *mit Fehlern umgehen* und *externe Bauelemente anschließen* ergänzt. Innerhalb der Hauptbereiche wird, jeweils von unten nach oben ablesbar, der lineare Kompetenzzuwachs deutlich. Die Kompetenz *externe Bauelemente anschließen* bietet viele Möglichkeiten, den *Calliope mini* praktisch einzusetzen, ist aber nicht weiter ausgeführt, da sie im vorliegenden Lehrwerk noch nicht konkretisiert wurde.
Die Anordnung der Lernlandkarte als Kreisdiagramm macht deutlich, dass der Kompetenzerwerb nicht nur linear aufgebaut ist, sondern darüber hinaus auf einem lebendigen Wechselgefüge der einzelnen Kompetenzen fußt.

Im vorliegenden Lehrermaterial werden Sie als Lehrperson angesprochen, ohne Ihnen Möglichkeiten für das Übertragen des dargebotenen Materials in den Unterricht aufzuzeigen. Aus Rückmeldungen von Lehrkräften, die im Unterricht bereits mit dem Coden begonnen haben, haben wir Erfahrungen und Empfehlungen zusammengetragen:

- Machen Sie sich mit dem Aufbau und den Möglichkeiten von *NEPO®* und des *Calliope mini* vertraut.
- Probieren Sie die Codingbeispiele selbst aus.
- Nutzen Sie hierzu den freien Internetzugang zum Editor. Sie können das Programmieren mit *NEPO®* auch ohne den *Calliope mini* trainieren und Ihren Code simulieren.
- Werden Sie routiniert im Speichern der Codes auf Ihrem Computer, um Ihre Codes zu archivieren und üben Sie das Übertragen des Codes auf den *Calliope mini*.
- Variieren Sie bestehende Codes nach eigenen kreativen Ideen.
- Schreiben Sie eigene (kleine) Programme.
- Suchen Sie im Kollegium nach Mitstreitern und werden Sie gemeinsam kreativ.
- Das eigenhändige Ausprobieren dient nicht nur der Entfaltung Ihrer Kreativität, sondern auch der Entwicklung von Handlungssicherheit, wenn bei Ihren Schülerinnen und Schülern im Unterricht Fragen und Fehler auftreten.

Wir hoffen, dass der Funke auf Sie übergesprungen ist und Sie das Coden in Ihrer Klasse ausprobieren möchten. Unverzichtbare Voraussetzung ist die Bereitstellung der benötigten digitalen Geräte für die Programmierung.

Nachfolgend schildern uns Lehrkräfte ihre ersten Schritte sowie Bedingungen, damit das Coden innerhalb des Schulalltags der Grundschule gelingt:

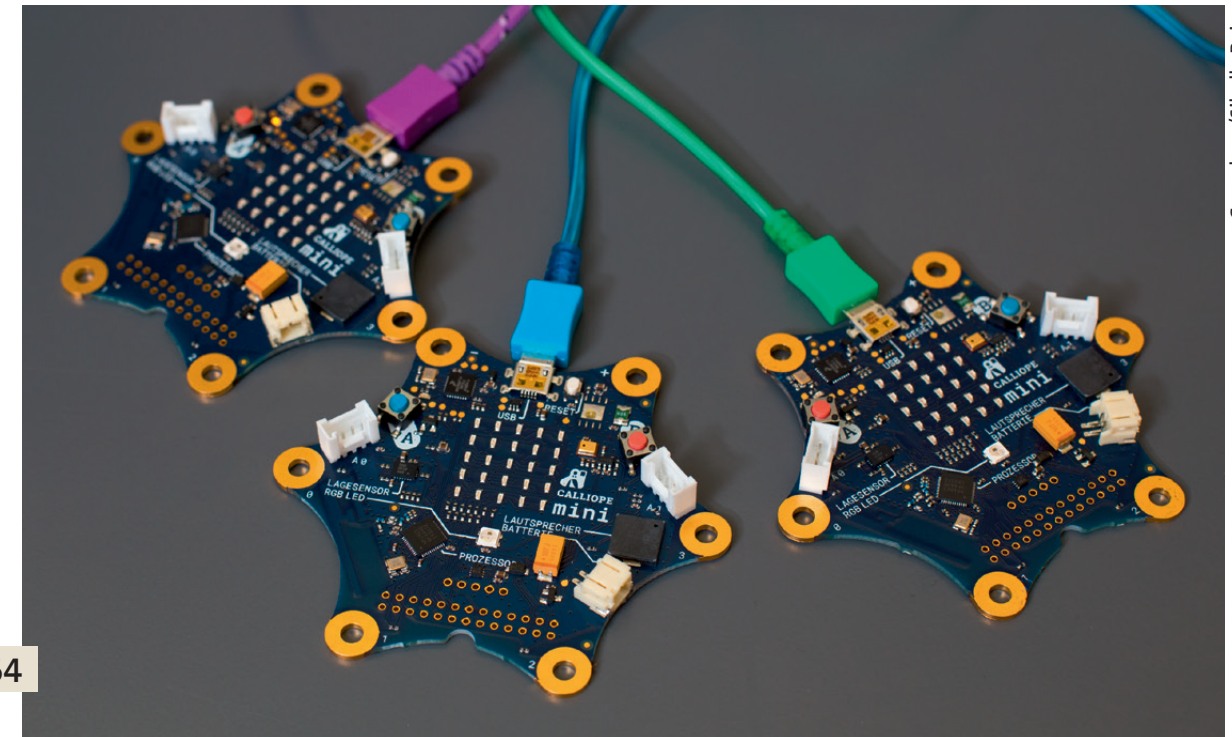
- zwei Schüler pro Gerät (z. B. Laptop, PC) zum Coden
- einen *Calliope mini* für jeden Schüler
- Einstieg mit geringerer Schülerzahl, z. B. im Rahmen einer Nachmittags-AG mit 10–12 Kindern an 5–6 Computern. Ideal wäre zu Beginn des Coding-Kurses ein erwachsener Helfer, um die bestehenden Einstiegshürden der Kinder zu überwinden (z. B. mangelndes Computer-Basiswissen, sich vertippen, mangelnde Routine im Umgang mit den Blöcken, ...).
- Ermutigen Sie Ihre Schüler, in Absprache mit den Eltern, auch von zu Hause aus den freien Internetzugang zum Editor zu nutzen. Die Kinder können das Programmieren auch ohne den *Calliope mini* trainieren. Der Editor *NEPO®* bietet eine gute Simulation der geschriebenen Programme.

Ihr Feedback ist uns wichtig!

Sie haben die Beispiele zu Hause oder vielleicht sogar mit Ihrer Klasse ausprobiert? Lassen Sie uns an Ihren Erfahrungen, Ideen und Tipps teilhaben. So helfen Sie mit, die Materialien optimal hinsichtlich der Anforderungen im Unterricht auszubauen. Wir freuen uns auf Ihr Feedback! Schreiben Sie uns unter calliope@cornelsen.de

Hinweis auf Schülermaterial

- einsetzbar ab Klasse 3 für die Fächer Deutsch, Sachkunde und Mathematik
- aufbauend auf den Beispielen im vorliegenden Lehrermaterial
- erscheint im Herbst 2017



Cornelsen/Sibylle Baier

CALLIOPE

Calliope mini ist ein Produkt der Calliope gGmbH
Mit dem Mikrocontroller *Calliope mini* soll es jedem Schulkind in Deutschland ab der 3. Klasse möglich sein, einen spielerischen Zugang zur digitalen Welt zu bekommen. Denn nur wenn wir über digitale Kenntnisse verfügen, können wir alle aktiv an der Gesellschaft teilhaben und sie mitgestalten.

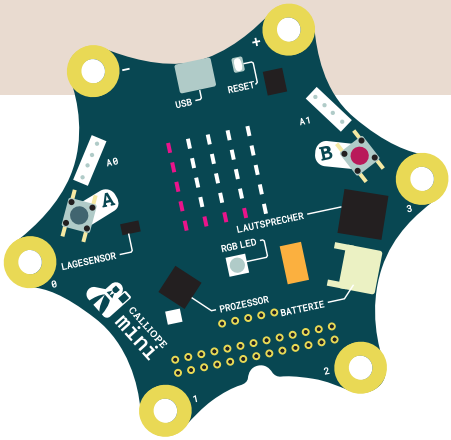
Dafür arbeiten im Team von Calliope Fachleute aus dem IT- und Bildungsbereich interdisziplinär zusammen.

Mehr Informationen zur Initiative finden Sie unter calliope.cc

OPEN Roberta®

Das *Open Roberta Lab* ist eine frei verfügbare Programmierplattform, auf der Kinder, Jugendliche und Erwachsene – auch ohne Vorkenntnisse – programmieren lernen können. Schülerinnen und Schüler erwecken den *Calliope mini* mit der grafischen Programmiersprache *NEPO*® intuitiv zum Leben. Open Roberta® ist eine technologische Weiterentwicklung der Fraunhofer-Initiative „*Roberta*® – Lernen mit Robotern", die seit 2002 digitale Bildung in Deutschland fördert. Das *Open Roberta*-Projekt wurde vom Fraunhofer IAIS in Kooperation mit Google initiiert. *Roberta*, *Open Roberta* und *NEPO* sind eingetragene Marken der Fraunhofer-Gesellschaft für angewandte Forschung e. V.

Hier geht es zum *Open Roberta Lab*: lab.open-roberta.org



Coden mit dem Calliope mini

Die Lehrermaterialien zu **Coden mit dem Calliope mini – Programmieren in der Grundschule** richten sich an Lehrerinnen und Lehrer ab Klasse 3.

- 11 Coding-Beispiele zu Inhalten aus den Lehrplänen der Fächer Sachkunde, Deutsch und Mathematik der Klassen 3 und 4.
- Als Coding-Neuling programmieren Sie Schritt für Schritt Ihre ersten eigenen Programme.
- Sie bauen Ihre Coding-Kompetenz systematisch anhand der Inhaltsbeispiele auf.
- Erleben Sie den Calliope mini als faszinierendes Werkzeug für Ihren schulischen Unterricht.

Probieren Sie es aus und erleben Sie selbst, wie begeisternd und einfach Coden ist!

Cornelsen

ISBN 978-3-06-600012-2