

Johannes Wallroth

SCRATCH- PROJEKTE

FÜR TEENS

30 Spiele programmieren
von **easy** bis **knifflig**

ab
Klasse

5



INHALTSVERZEICHNIS

VORWORT	7
----------------------	----------

TEIL 1: ERSTE SCHRITTE MIT SCRATCH	9
---	----------

1.1 Grundlagen von Scratch	9
1.2 Erste Projekte mit Scratch	21

TEIL 2: SPIELE-PROJEKTE	31
--------------------------------------	-----------

2.1 Würfel	32
2.2 Zahl erraten	35
2.3 Rechenspiel	39
2.4 Äpfel fangen	42
2.5 Aquarium	47
2.6 Orangen-Jagd	50
2.7 Katzen und Maus	52
2.8 Klon-Pinsel	56
2.9 Klick-Spiel	59
2.10 Jumping Game	63
2.11 Autorennen	66
2.12 Schlange	69
2.13 Tennis	73
2.14 Steinbrecher	78
2.15 Flappy Cat	82
2.16 Froschhüpfer	85
2.17 Glücksspiel-Automat	90
2.18 Torwart	94
2.19 Adventure	97
2.20 Tierstimmen-Memory	102
2.21 Daleks	108
2.22 Kampf der Klone	112
2.23 Kugelsammler	115
2.24 Die gefräßige Krabbe	118
2.25 Ballonschießen	122
2.26 Jump and Run	126
2.27 Side-Scroller	133
2.28 Vertikal-Scroller	138
2.29 Plattformer	145
2.30 Plattformer mit Levels	149

TEIL 3: ÜBUNGEN UND MINI-PROJEKTE 153

3.1	Bunte Bälle	154
3.2	Tuschkasten	156
3.3	Besserer Würfel	158
3.4	Der durstige Schmetterling	160
3.5	Wort-Generator	162
3.6	Größe schätzen	164
3.7	Krabbelkäfer	166
3.8	Labyrinth	168
3.9	Fütterung	170
3.10	Wie programmiert man	172

ANHANG: PROGRAMMIEREN MIT DEM CALLIOPE 175

A.1	Würfel	178
A.2	Torzähler	180
A.3	Rechner	182
A.4	Bistabiler Schalter	183
A.5	Schere, Stein, Papier	184
A.6	Blinker	185
A.7	Reaktionsspiel	186
A.8	Einkaufsliste	188
A.9	Hieroglyphen-Generator	189
A.10	Codeschloss	190

Vorwort

Die blockbasierte Programmiersprache *Scratch* ist momentan eine der besten, wenn nicht sogar die beste Möglichkeit, Kindern und Jugendlichen das Programmieren beizubringen. Sie ist aufgrund ihrer intuitiven, grafischen Oberfläche mit sehr niedriger Lernschwelle leicht zugänglich und mit der Zentrierung auf Spiele besonders attraktiv für eine junge Zielgruppe. Aber auch viele Erwachsene spielen gern und haben Freude am Tüfteln und Kombinieren. Programmieren kann nicht nur eine ernste Beschäftigung oder sogar ein Beruf, sondern auch ein großes Abenteuer mit sehr viel Spaß sein. Dieses Buch möchte genau das zeigen.

»Brauchen wir noch ein Buch über Scratch?«, könnten nun einige Leser fragen. Angesichts der beachtlichen Anzahl bereits erschienener Bücher und Internetquellen zu diesem Thema sicher eine berechtigte Frage, aber aus meiner langjährigen Erfahrung als Leiter von Computer-AGs mit Schülern der 5. und 6. Klasse kann ich voller Überzeugung sagen: »Ja, unbedingt!«

Das vorliegende Buch versucht nämlich, eine Sache besser zu machen als einige andere: den oft viel zu hohen Schwierigkeitsgrad der Beispielprojekte auf ein Minimum abzusenken und den Lernstoff in möglichst viele kleine, leicht verständliche Projekte zu verpacken.

Angesichts der zunehmenden Bedeutung, die das Fach Informatik in den letzten Jahren an den Schulen gewonnen hat, ist dieses Buch aus dem Wunsch heraus entstanden, Anfängern bei ihren ersten Gehversuchen zu helfen und Lehrkräften eine Sammlung praktischer Beispiele für die Gestaltung ihres Unterrichts an die Hand zu geben – ausreichend für einige Monate. Nach meiner Erfahrung fehlt ein Buch wie dieses momentan noch in der Medienlandschaft.

Alle Programmbeispiele wurden von mir selbst entwickelt und für dieses Buch optimiert. Das Ziel ist, für jedes einzelne Projekt nicht länger als etwa eine Stunde aufzuwenden und dabei jedes Mal etwas Neues vermittelt zu bekommen – vor allem aber: Spaß zu haben!

Über den Autor

Johannes Wallroth ist ein klassischer Quereinsteiger: Er brachte sich das Programmieren um die Jahrtausendwende selbst bei – und hat seine Begeisterung dafür bis heute nicht verloren. Seitdem hat er umfangreiche Erfahrung in verschiedenen Programmiersprachen gesammelt, darunter C, C++, C#, Pascal/Delphi, Java, JavaScript, Python, PHP und natürlich Scratch.



Seit 2019 unterrichtet er Kinder der 5. und 6. Klasse in Arbeitsgemeinschaften verschiedener Berliner Schulen mit dem Calliope Mini und Scratch. Aus den dabei entstandenen Beispielprojekten und Notizen entwickelte sich im Laufe der Zeit die Idee zu diesem Buch. Die meisten hier vorgestellten Projekte entstanden ursprünglich zur Vorbereitung dieser AGs, die anderen wurden eigens für dieses Buch entwickelt.

Im Laufe der Jahre wurden auch die verwendeten Konzepte und Methoden immer weiter verbessert und an die Praxis angepasst. Vor allem den richtigen Mittelweg zwischen »zu einfach« und »zu schwer« zu finden und dabei möglichst viele Lernende »mitzunehmen«, ist für ihn bis heute eine kleine Kunst und eine stetige Herausforderung.

Johannes Wallroth lebt mit seiner Frau und seinem Sohn in Berlin. Über Feedback über seine Website programming.de freut er sich.

Berlin, April 2026

Teil 1:

Erste Schritte mit Scratch

1.1 Grundlagen von Scratch

Auch wenn Scratch mit seiner grafischen Oberfläche aus farbigen Blöcken eher wie ein Puzzlespiel für Kinder aussieht und nicht wie eine »richtige« Entwicklungsumgebung mit ihren zahllosen Optionen und ihrem kryptischen Programmcode, bietet es doch viele mächtige Möglichkeiten, die man anfangs kaum vermuten würde, und enthält alle Elemente, um Kindern, Jugendlichen und auch Erwachsenen spielerisch das Programmieren beizubringen. Das mit Scratch erworbene Wissen kann sogar später auf andere Programmiersprachen übertragen werden, denn die grundlegenden Konzepte, Werkzeuge und Bausteine unterscheiden sich weniger, als man denken mag.

Auch im Internet findet sich eine Fülle von veröffentlichten Projekten für Scratch, diese sind jedoch meiner Meinung nach leider oft zu umfangreich und kompliziert und daher zum Lernen für Anfänger selten geeignet. Dieses Buch soll sowohl für Lehrkräfte, für Eltern als auch für Schüler und Programmieranfänger ein handfester Ratgeber und Leitfaden sein, der schnell zu Erfolgserlebnissen führt. Wer will, kann damit auch systematisch lernen, etwas nachschlagen oder einfach Spaß mit den vorgestellten Projekten haben.

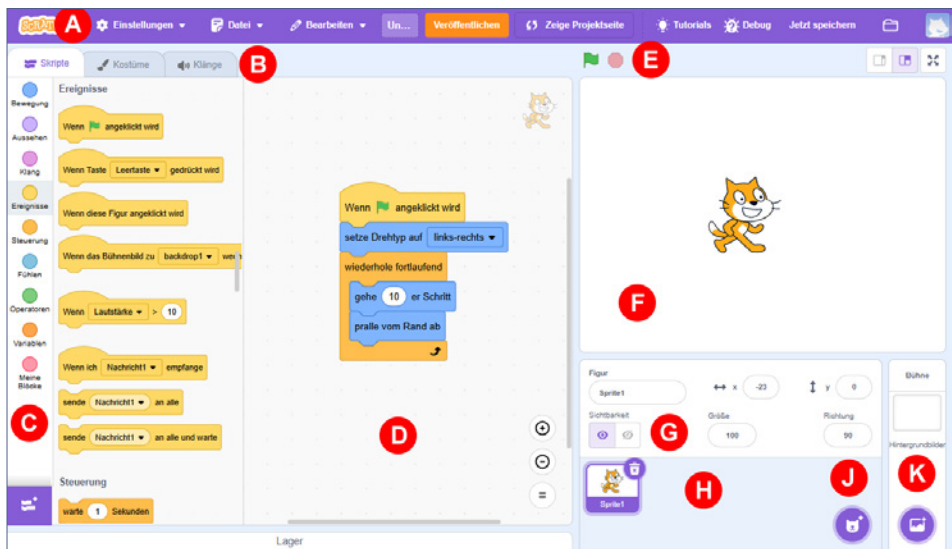
Dabei sollte immer versucht werden, die Projekte genau zu verstehen. Beim Nachprogrammieren und Experimentieren wird sich ganz von selbst allmählich ein Verständnis für die verschiedenen Konzepte und Möglichkeiten einstellen. Ideen für weitere Verbesserungen kommen dann oft ganz von allein, denn beim Programmieren ist die Grenze für das Mögliche nur die eigene Fantasie. Und nun viel Spaß und Erfolg mit diesem Buch!

Der Scratch-Editor

Du kannst mit Scratch in jedem beliebigen Webbrowser programmieren und musst nichts auf deinem Computer installieren. Du gibst diese Adresse ein:

scratch.mit.edu/projects/editor

... und kannst sofort loslegen. Der Editor öffnet sich mit einem neuen Projekt, einem weißen Hintergrund und einer Katze als vorgegebener Figur. Wenn sich noch ein Tutorial-Video öffnet, kannst du das schließen. Als Erstes solltest du in den **Einstellungen** die Sprache des Editors auf Deutsch einstellen und vielleicht auch (wie durchgehend in diesem Buch) den Farbmodus auf **Hoher Kontrast**, was die Lesbarkeit der Schrift auf den Blöcken verbessert.



- A** Menü mit Einstellungen zum Laden, Speichern, Umbenennen, Anmelden
- B** Tabs zur Auswahl des Bearbeitungsmodus für Skripte, Kostüme und Klänge
- C** Werkzeugkasten
Die Rubriken sind farblich sortiert und lassen sich nach Klick auf das runde Symbol auswählen: Bewegung, Aussehen, Klang usw.
- D** Arbeitsfläche
Hier werden die Programm-Blöcke für die ausgewählte Figur zusammengestellt.
- E** Symbole für Programm starten und Programm beenden

F Vorschau des Programms/Spiels

Oben rechts gibt es eine Schaltfläche für den Vollbildmodus, damit kannst du das Spiel dann »richtig« sehen. Außerdem gibt es eine Schaltfläche, um die Vorschau zu verkleinern und die Arbeitsfläche zu erweitern.

G Eigenschaften-Fenster der ausgewählten Figur

Dort kannst du deren Namen, Position, Sichtbarkeit, Größe und Richtung sehen und verändern. Wenn das Programm läuft, ändern sich auch dort die entsprechenden Werte. Nach Klick auf die Richtung kannst du hier auch den Drehtyp der Figur einstellen.

H Symbole für die im Projekt vorhandenen Figuren

Durch Anklicken kannst du die entsprechende Figur auswählen und ihre Eigenschaften und ihr Programmskript bearbeiten.

J Dieser Bereich verwaltet die Bühne.

Die Miniatur oben zeigt das aktive Hintergrundbild. Nach einem Klick darauf kannst du die Hintergrundbilder und das Programmskript der Bühne bearbeiten. Mit der runden Schaltfläche unten kannst du neue Hintergrundbilder auswählen. Dort kann auch eine Figur ganz neu im Grafik-Editor gezeichnet oder aus einem fertigen Bild hochgeladen werden. Es gibt eine reiche Auswahl fertiger Hintergrundbilder, die du jedoch auch verändern oder durch eigene Bilder ersetzen kannst.

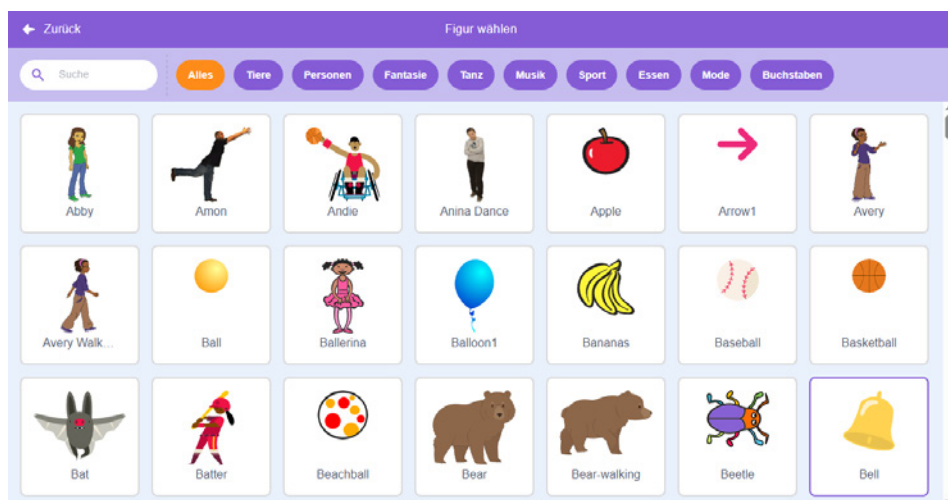
Die Figuren

Wenn ein neues Projekt bei Scratch geöffnet wird, befindet sich immer die Figur der Katze auf der Bühne, die du meistens sofort löschen kannst, denn es gibt noch viele andere Figuren zur Auswahl. Du kannst sogar Bilder importieren oder aber selbst deine Figuren zeichnen, und das wirst du später auch manchmal tun. Klicke jetzt einmal auf **Figur wählen** im Editor.



Figur wählen

Es öffnet sich eine umfangreiche Sammlung von Figuren, die nach Themen gruppiert ist.

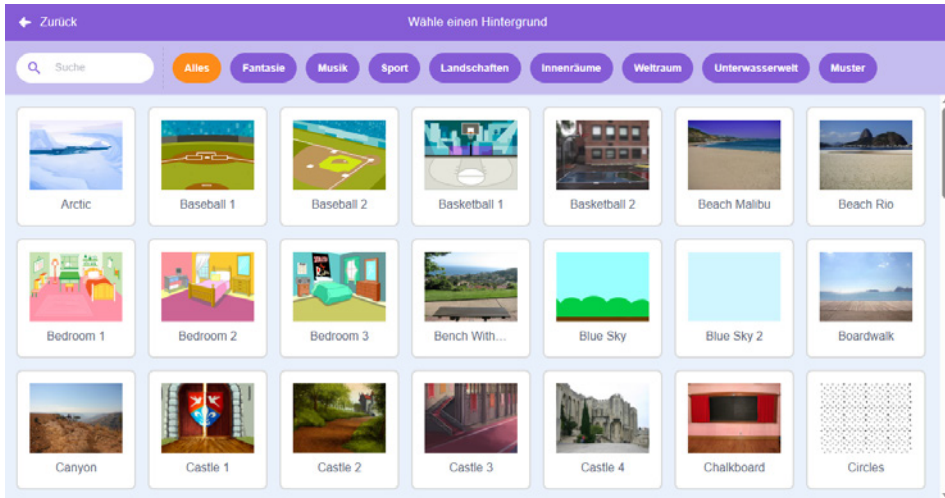


Wenn du darin deine Maus bewegst, siehst du, dass sich manche Figuren bewegen. In solchen Fällen hat eine Figur mehrere »Kostüme«, wie es in Scratch heißt – oder Phasen einer Animation, wie in einem Cartoon. Du wirst bald lernen, diese Animationen einzusetzen.

Jede Figur kann ihr eigenes »Programm« (in Scratch eine Ansammlung von Blöcken) haben und auf Aktionen der Maus, der Tastatur und andere Ereignisse reagieren, zum Beispiel auf den Zusammenstoß mit anderen Figuren. Figuren können sogar Nachrichten versenden, die von anderen Figuren im Programm empfangen werden können. Jede Figur kann »sehen«, wo sich die anderen Figuren befinden, und einiges mehr. Figuren können außerdem Sound-Effekte abspielen, wofür in Scratch eine umfangreiche Sammlung von Klängen zur Verfügung steht. Du kannst auch eigene Sound-Dateien hochladen und in deinem Spiel verwenden.

Die Bühne

Ganz unten rechts im Scratch-Editor findest du den Knopf für die Auswahl der Bühne mit den Hintergrundbildern für dein Projekt. Meistens wird dir ein einziges Hintergrundbild für dein Projekt genügen, aber es können auch mehrere sein, zwischen denen umgeschaltet wird.



Die Hintergrundbilder sind nach Themen gruppiert.



Wie bei den Figuren findest du auch hier die drei Tabs **Skripte**, **Hintergrundbilder** und **Klänge**, denn eine Bühne hat viele Optionen, die auch Figuren haben. Man kann sogar, wenn man will, die Bilder im eingebauten Grafik-Editor bearbeiten, zum Beispiel, um Elemente hinzuzufügen oder zu verändern. Die Bühne kann auch Programm-Skripte verwenden, etwa um Maus-Ereignisse oder Tastendrücke zu verarbeiten oder Grafikeffekte zu verwenden. Damit ergeben sich interessante Möglichkeiten. Alle Figuren können untereinander und mit der Bühne Nachrichten austauschen, um Aktionen auszulösen. Eine Figur kann auch das jeweils aktive Bühnenbild erkennen und ändern, entweder über dessen Namen oder die Nummer.

Der Werkzeugkasten

Ganz links im Scratch-Editor findest du den »Werkzeugkasten« mit den verschiedenen Blöcken, aus denen du deine Programme zusammenstellen kannst. Die Blöcke sind in Rubriken mit verschiedenen Farben gruppiert. Bis auf die Startblöcke hat jeder Block zumindest auf der oberen Seite eine Möglichkeit, an einem anderen Block »anzudocken«. Du ziehst die Blöcke aus dem Werkzeugkasten auf deine Arbeitsfläche und kannst sie dort in Stapeln und Gruppen anordnen. Nicht mehr benötigte Blöcke löschst du entweder mit Rechtsklick und **Lösche Block** oder du ziehst sie zum Löschen mit gedrückter linker Maustaste zurück auf den Werkzeugkasten.



Die Blöcke in jeder Gruppe werden beim Start des Projekts von oben nach unten abgearbeitet, sobald das entsprechende Ereignis eintritt. Ganz oben befindet sich fast immer ein Block aus der Rubrik »Ereignisse« – z.B. **Wenn Start-Flagge angeklickt wird**, **Wenn diese Figur angeklickt wird**, **Wenn Taste XY gedrückt wird**, **Wenn ich Nachricht empfangen wird** usw. Nach dem letzten Block endet die Ausführung des Stapels.

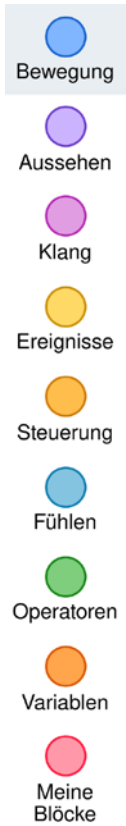
Schauen wir uns ein Beispiel an. Du behältst die Katze als einzige Figur auf der Bühne und ziehst die abgebildeten Blöcke auf deine Arbeitsfläche. Dann startest du dein Projekt mit

einem Klick auf die Start-Flagge über dem Vorschaufenster.

Der oberste Block wird sofort beim Programmstart (nur einmal) abgearbeitet und die Katze sagt: »Hallo!« Wenn du die Figur (die Katze) anklickst, wird der zweite Block abgearbeitet und die Katze sagt: »Das kitzelt« (oder was immer du in das Textfeld eingibst). Der dritte Block wird beim Drücken der Leertaste aufgerufen (du kannst die Taste natürlich auch ändern) und die Katze wechselt zum nächsten Kostüm (zur nächsten Animationsphase).

Alle drei Blöcke sind sogenannte Kopfblöcke. Danach folgen dann ein oder mehrere Stapelblöcke und manchmal noch ein Endblock, hinter dem kein weiterer Block stehen kann.

Grundlagen der Programmierung



Mit Scratch lernst du Programmieren auf eine objektbasierte Art kennen – einem wichtigen Konzept der Programmierung. Es bedeutet, dass Objekte in einem Computerprogramm Dinge aus der realen Welt repräsentieren können. Bei Scratch sind das die Figuren, in anderen Programmiersprachen spricht man von Objekten.

Jede Figur, die du in dein Projekt importierst, kann ihr eigenes Skript und eigene Eigenschaften und Fähigkeiten haben. Jede Figur kann auch mit anderen Figuren interagieren und Nachrichten austauschen. Zum Beispiel kann sie auf den Zusammenstoß mit einer anderen Figur reagieren. Auch die Bühne (der Hintergrund) kann eigene Skripte besitzen und etwa auf Mausklicks reagieren.

In jeder Programmiersprache (zu denen auch Scratch gehört) gibt es ähnliche Konzepte, Werkzeuge und Möglichkeiten, um Aufgaben zu lösen. Andere Sprachen arbeiten mit Programmcode in Textform und man muss im Kopf eine gestellte Aufgabe in den entsprechenden Text umformulieren und in den Editor schreiben. Dabei kann man sich heutzutage allerdings auf zahlreiche Hilfsmittel stützen, wie Code-Vervollständigung, Bibliotheken und integrierte Assistenten. Bei Scratch wirst du vom Schreiben des Programmcodes durch eine grafische Oberfläche komplett befreit und ziehst ganz einfach farbige Blöcke auf die Arbeitsfläche.

Der große Vorteil dabei ist, dass auf diese Weise ein einfacher Zugang zum Programmieren ermöglicht wird. Ein Nachteil dieses Konzepts ist, dass bei größeren Programmen leichter der Überblick verloren geht, da Blöcke viel mehr Platz benötigen als Text und es weniger Möglichkeiten gibt, den Code zu gliedern und zu gruppieren oder gerade nicht benötigte Teile auszublenden. Auch aus diesem Grund beschränkt sich dieses Buch auf möglichst kurze Programme, die noch ohne Scrollen auf einen Bildschirm passen.

Alle Werkzeuge, die in Scratch zur Verfügung stehen, sind über den Werkzeugkasten auf der linken Seite des Editors zugänglich. Durch Klicken auf die farbigen Kreise lassen sich die einzelnen Block-Kategorien schnell aufrufen, aber durch Scrollen erreicht man sie auch. Die Gruppierung in Kategorien erleichtert es, einen gesuchten Block schnell zu finden. Wir wollen nun einige Blöcke näher betrachten, die wir in fast allen Projekten benötigen werden. Diejenigen, die seltener vorkommen, kannst du später selbst erforschen. Das Internet liefert dir eine unerschöpfliche Fülle von Informationen über Scratch.



Bewegung

Beginnen wir mit der Rubrik **Bewegung**. Hier findest du Blöcke, die sich mit der Bewegung einer Figur auf der Spielfläche befassen. Bei Scratch gibt es zwei Haupt-Möglichkeiten, eine Figur zu bewegen: 1.) durch direkte Änderung ihrer x- und y-Koordinaten auf der Spielfläche (**ändere x/y um ...** und **setze x/y auf ...**) oder 2.) durch Vorgeben einer Richtung (einem Winkel zwischen 0° und 360°, wobei 0° oben ist, 90° rechts, 180° unten und 270° links) und anschließendem Aufruf des Befehls **gehe 10er Schritt** (die Zahl 10 ist nur eine Vorgabe und kann geändert werden, so wie alle Parameter in den Blöcken). Welche der beiden Möglichkeiten du benutzt, hängt von der gestellten Aufgabe ab. Mal ist die eine Möglichkeit einfacher, mal die andere. Das Ziel sollte immer sein, eine Aufgabe mit so wenig Blöcken wie möglich zu lösen und dabei aber auch die Übersichtlichkeit und Verständlichkeit im Auge zu behalten. Eine Figur kann außerdem mithilfe der Blöcke **gehe zu ...** und **gleite in 1 Sek. zu ...** an eine bestimmte Position, zum Mauszeiger oder zu einer anderen Figur bewegt werden.

Beachten wir den Unterschied zwischen **ändere x/y um 10** und **setze x/y auf 10** (wobei 10 nur der Vorgabewert ist, den du ändern kannst). Diese beiden Blöcke werden von Anfängern gern verwechselt! Wenn du einen Wert »setzt«, dann tritt dieser Wert sofort in Kraft und die Figur befindet sich dann an diesem Punkt. Wenn du die Funktion mehrfach aufrufst, wird die Position immer wieder auf den angegebenen Wert gesetzt.

Wenn du die Position hingegen änderst, wird sie vom bestehenden Wert ausgehend um den angegebenen Wert geändert und ändert sich immer weiter, wenn die Funktion mehrmals aufgerufen wird. Angenommen, x ist 180 und du sagst **ändere x um 10**, dann ist x danach 190, beim nächsten Aufruf 200 usw.



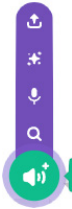
Aussehen

Unter **Aussehen** findest du hauptsächlich Blöcke, die mit dem Aussehen einer Figur oder eines Bühnenbilds zu tun haben. Auch die Bühne (der Hintergrund) kann Programmcode erhalten und teilt dabei viele Optionen mit den Figuren. Du findest hier auch die Möglichkeit, Texte über »Sprechblasen« (oder »Denkblasen«) auszugeben, ähnlich wie in einem Comic. Das ist in Scratch auch die einzige Möglichkeit für die Textausgabe. Du kannst hier eine Figur in der Größe ändern, ihr Effekte zuweisen (Farbe, Helligkeit, Durchsichtigkeit und mehr), sie zeigen oder verstecken (diese Option gibt es auch im Eigenschaften-Fenster, dort heißt sie **Sichtbarkeit**) und ihre Kostüme ändern. Du kannst auch hier wieder eine Eigenschaft zuweisen oder sie ändern, wie bei der Bewegung. Du kannst also etwa die Größe entweder auf 100% setzen oder sie zum Beispiel mit dem Parameter 10 bei jedem Aufruf um 10% vergrößern (mit -10 wird die Figur entsprechend um 10% kleiner). Du kannst, sofern die Figur mehrere »Kostüme« hat, das nächste vorhandene Kostüm auswählen (**wechsle zum nächsten Kostüm**) oder ein bestimmtes Kostüm auswählen (**wechsle zum Kostüm ...**).



Klang

Du kannst in Scratch Klänge abspielen und es gibt dafür eine umfangreiche Sammlung. Falls dir ein bestimmter Klang fehlt, kannst du sogar eigene Sound-Dateien hochladen und verwenden.



Klang wählen

Der Unterschied zwischen **spiele Klang ...** und **spiele Klang ... ganz** besteht darin, dass beim zuletzt genannten der nächste Block im Programm erst dann angesteuert wird, wenn der Sound ganz abgespielt wurde. Bei **spiele Klang ...** wird das Programm sofort fortgesetzt. In diesem Buch werden Klänge selten verwendet, um nicht vom Wesentlichen abzulenken.



Ereignisse

Mit einem Block aus der Kategorie **Ereignisse** startet fast jeder Blockstapel. Beim Start des Programms wird zuerst einmalig der Block **Wenn Start-Flagge angeklickt wird** aufgerufen und das, was sich dort befindet, ausgeführt, ein Block nach dem anderen. Es gibt einen Block, der Tastendrücke auswertet, einen für Mausklicks auf die Figur und sogar die Möglichkeit, Nachrichten zu senden und zu empfangen. Auf diese Weise kann eine Figur mit allen anderen Figuren und auch mit der Bühne kommunizieren und damit Ereignisse bei den anderen Figuren auslösen. Die Möglichkeiten sind grenzenlos.



Steuerung

In der Kategorie **Steuerung** befinden sich einige weitere wichtige Blöcke. Der **warte 1 Sekunde(n)**-Block wird eingesetzt, wenn das Programm an dieser Stelle verzögert werden soll, z.B. beim Wechsel der Kostüme (Animationsphasen). Scratch schafft eine Framerate von 30 FPS (*frames per second*) – das heißt, es werden höchstens 30 Bilder pro Sekunde angezeigt. Manchmal muss man daher künstlich eine Verzögerung einbauen, um den gewünschten optischen Effekt zu erzielen.

Hier finden wir auch insgesamt drei Varianten der **Schleife**, einem elementaren Baustein in jeder Programmiersprache, den wir oft benutzen werden. Alle Blöcke, die sich in einer Schleife befinden, werden mehrmals ausgeführt. Bei der **wiederhole fortlaufend**-Schleife geschieht das im Prinzip unendlich oft oder bis das Skript gestoppt wird. Die Variante **wiederhole 10-mal** (diese Zahl kann natürlich geändert werden) läuft so oft, wie es der Parameter (hier: 10) vorgibt, und die Variante **wiederhole bis ...** erwartet eine Bedingung als Parameter, zum Beispiel $x > 100$ – in diesem Fall würde die Schleife nur so lange laufen, wie die Variable x kleiner oder gleich 100 ist. Danach wird sie verlassen und das Programm springt zum Block unterhalb der Schleife, sofern es dort einen gibt.

In der Kategorie **Steuerung** findest du auch die **Verzweigung** oder **bedingte Ausführung** in verschiedenen Formen – sogenannte *Kontrollstrukturen* – den **falls ... dann**-Block und in seiner erweiterten Form **falls ... dann ... sonst**. Auch diese Blöcke wirst du sehr oft verwenden – sie ermöglichen die Ausführung von Code, wenn (falls) eine bestimmte Bedingung wahr ist. Diese Bedingung kann zum Beispiel die Position einer Figur an einer bestimmten Stelle der Bühne sein oder der Wert einer

Variablen. Der **dann ...**-Zweig wird ausgeführt, wenn die Bedingung nicht wahr ist, also in allen übrigen Fällen. In diesem Zweig könnte auch noch eine weitere Verzweigung eingebaut werden, die andere Fälle prüft.

Drei weitere interessante Bausteine befinden sich ganz unten in dieser Rubrik: die Anweisung **erzeuge Klon von ...**, das Ereignis **Wenn ich als Klon entstehe** und der Befehl **lösche diesen Klon**. Du wirst sie später genauer kennenlernen. Du kannst von einer Figur eine Anzahl identischer Kopien anfertigen (**erzeuge Klon von ...**) und mit dem Block **Wenn ich als Klon entstehe** legst du den Programmcode nach der Entstehung eines Klons fest – und zwar für jeden Klon einzeln. Dort kann etwa die Position der Klone auf neue Werte gesetzt werden, damit sich nicht alle Klone an derselben Position befinden wie ihre Eltern-Figur (denn sie erben auch deren Position bei ihrer Entstehung).



Fühlen

Fühlen ist ein schönes Wort für diese Kategorie, denn obwohl wir es in Scratch nur mit Computerfiguren zu tun haben, können wir uns doch vorstellen, es wären richtige lebendige Figuren, Tiere oder Menschen.

Nur unsere Fantasie (und natürlich auch unser Können) ist die Grenze für das Geschehen in unserer eigenen Computerwelt! Beginne mit **wird Mauszeiger berührt**. Hier kannst du den Mauszeiger durch »Rand« ersetzen oder eine beliebige andere Figur, die sich in deinem Programm befindet. Du kannst damit Kollisionen (Berührungen oder Zusammenstöße) registrieren und, wie du an der spitzen Form dieses Blocks erkennst, eignet er sich perfekt, um als Bedingung in einer Verzweigung (einer bedingten Ausführung) eingesetzt zu werden. Eckige Blöcke passen nur in eckige Lücken anderer Blöcke, ovale Blöcke nur in eine ovale Lücke. Außer auf die Kollision mit dem Rand oder einer anderen Figur kannst du auch auf den Zusammenstoß mit einer Farbe irgendwo auf dem Spielfeld reagieren – und diese Farbe kann sogar ein Teil der Bühne sein! Um den Wert einer bestimmten Farbe genau zu treffen, verwendest du die **Pipette** (ganz unten im Farbwahl-Dialog) und ermittelst damit die genaue Farbe durch Klick auf den Bildschirm.

Ein weiterer sehr nützlicher Block ist **Entfernung von Mauszeiger** – mit dem du auch die Entfernung zu anderen Figuren ermitteln kannst. Du kannst hier auch einen Tastendruck (Maus oder Tastatur) finden, der in einige Blöcke bei Steuerung passt, sowie die Mausposition. Mit dem Block **frage ... und warte** kann eine Texteingabe des Benutzers gestartet werden, deren Ergebnis anschließend in der Variable **Antwort** für die weitere Verarbeitung zur Verfügung steht.



Operatoren

In der Kategorie **Operatoren** finden wir **Arithmetische Operatoren** und **Vergleichs-Operatoren**. Es sind Bausteine, um selbst definierte oder von Scratch vorgegebene Variablen zu verbinden oder zu vergleichen, außerdem sogenannte boolesche Operatoren (und, oder, nicht) und einige andere. Die Operatoren für die Grundrechenarten kennst du aus dem Matheunterricht. Auch die Vergleichsoperatoren > (größer als), < (kleiner als) und = (ist gleich)

kennst du bestimmt. Mit den **booleschen Operatoren** »und« / »oder« kannst du zwei Bedingungen kombinieren. Bei »und« müssen beide gegebenen Ausdrücke wahr sein, damit der Gesamtausdruck wahr ist. Bei »oder« genügt bereits ein wahrer Ausdruck, damit auch das Ergebnis insgesamt wahr ist. Der »nicht«-Operator kehrt das übergebene Argument um: Wenn es wahr ist, gibt er falsch zurück und umgekehrt. In dieser Kategorie findest du auch die häufig verwendete **Zufallszahl** und weitere mathematische Funktionen.

Es gibt hier auch eine Reihe von Zeichenketten-Funktionen, die wir bei der Verwendung von Texten einsetzen können.



Variablen

In der Rubrik **Variablen** erstellen wir unsere eigenen Variablen. Du wirst sehr oft davon Gebrauch machen, denn du benötigst sie in fast jedem Programm. Eine neue Variable erstellst du beim Klick auf **Neue Variable**. Dabei kannst du auswählen, ob die Variable global gültig sein soll (**Für alle Figuren**) oder nur lokal für die aktuell ausgewählte Figur. Das brauchen wir vor allem, wenn wir mit Klonen arbeiten.

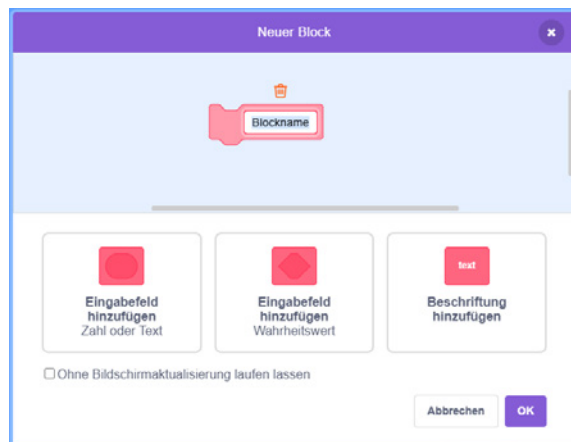
Du kannst Variablen jederzeit umbenennen (Rechtsklick auf die »Pille« mit dem Variablennamen). Wenn du im aktuellen Projekt noch keine eigenen Variablen hinzugefügt hast, findest du dort einen Platzhalter mit dem Namen »my variable«. Die beiden Funktionen, um Variablen zu verändern, sind: **setze Variable auf ...** und **ändere Variable um ...**

Diese beiden Blöcke werden manchmal verwechselt. Wenn du den Wert einer Variablen setzt, tritt der neue Wert sofort in Kraft und die Variable erhält den neuen Wert auch bei mehrmaligem Aufruf der Funktion. Wenn du den Wert hingegen änderst, wird er vom bestehenden Wert ausgehend um den angegebenen Wert geändert und ändert sich weiter um den entsprechenden Betrag, wenn die Funktion mehrmals aufgerufen wird. Wenn der Wert negativ ist, z.B. -5, wird der Wert der Variablen bei jedem Aufruf entsprechend kleiner.



Meine Blöcke

Als Letztes betrachten wir die Rubrik **Meine Blöcke**. Hier kannst du deine eigenen Blöcke (auch Funktionen oder Methoden genannt) definieren, die andere Codeblöcke einschließen. Sobald solch ein Block existiert, kann er im Programm einfach durch Aufruf seines Namens ausgeführt werden, also durch Einfügen des entsprechenden Blocks.



So kannst du mehrfach benötigte Programmteile einmal programmieren und dann beliebig oft wiederverwenden. Das verbessert die Übersicht bei einigen Programmen erheblich und spart außerdem Zeit. Du kannst dabei auch Parameter verwenden (Zahlen, Texte oder boolesche Werte). Sie werden dem Block beim Aufruf übergeben und verändern dessen Funktionsweise. Du kannst auch mehrere Parameter einsetzen.