

2.  
Auflage

Al Sweigart

# Cooler Spiele mit Scratch 3

Lerne programmieren und  
baue deine eigenen Spiele



dpunkt.verlag

# Inhalt

**Cover**

**Titel**

**Impressum**

**Widmung**

**Inhalt**

**Danksagungen**

**Einleitung**

Für wen ist dieses Buch gedacht?

Der Aufbau dieses Buches

Die Projekte durcharbeiten

Zusatzmaterial auf der Begleitwebsite

Korrekturen und Ergänzungen

**Kapitel 1: Erste Schritte mit Scratch**

Scratch ausführen

Der Offline-Editor

Figuren und der Scratch-Editor

Der Zeichenbereich

Codeblöcke

Blöcke hinzufügen

Blöcke löschen

Programme ausführen

Deine Programme vorführen

Wenn du Hilfe brauchst

Das Fenster »Tutorien«

Schau hinein

Zusammenfassung

## **Kapitel 2: Regenbogenlinien im Weltraum**

Das Projekt skizzieren

A. Den Weltraum-Hintergrund anlegen

1. Aufräumen und die Bühne vorbereiten

B. Drei bewegliche Punkte erstellen

2. Einen Punkt zeichnen

3. Code für die Figur Dot 1 hinzufügen

4. Die Figur Dot 1 kopieren

C. Die Regenbogenlinien zeichnen

5. Code für den zeichnenden Punkt hinzufügen

Das vollständige Programm

Turbo-Modus

Zusammenfassung

Wiederholungsfragen

## **Kapitel 3: Ein Labyrinthspiel**

Das Projekt skizzieren

A. Die Katze bewegen

1. Bewegungscode zu der Figur hinzufügen

2. Den Bewegungscode für die Katze duplizieren

B. Die Labyrinth der einzelnen Levels gestalten

3. Die Labyrinthbilder herunterladen

4. Das Bühnenbild ändern

5. Das erste Labyrinth einrichten

C. Verhindern, dass die Katze durch die Wände geht

6. Prüfen, ob die Katze die Wand berührt

D. Ein Ziel am Ende des Labyrinths anlegen

7. Die Apfelfigur erstellen

8. Erkennen, wann der Spieler den Apfel erreicht

9. Code zur Nachrichtenverarbeitung zur Figur Maze hinzufügen

Das vollständige Programm

Version 2.0: Zwei-Spieler-Modus

Die Apfelfigur duplizieren

Den Code von Apple2 ändern

Die orangefarbene Katze kopieren

Den Code für die blaue Katze ändern

Zurück an den Start

Cheat-Modus: Durch die Wände gehen

Den Code hinzufügen, mit dem die orangefarbene Katze Wände durchdringt

Den Code hinzufügen, mit dem die blaue Katze Wände durchdringt

Zusammenfassung

Wiederholungsfragen

## **Kapitel 4: Basketball mit Schwerkraft**

Das Projekt skizzieren

A. Die Katze springen und fallen lassen

1. Den Schwerkraftcode zur Katze hinzufügen

2. Code für die Landung auf dem Boden hinzufügen

3. Den Springcode zur Katze hinzufügen

B. Die Katze nach rechts und links gehen lassen

4. Gehcode zur Katze hinzufügen

C. Einen schwebenden Reifen erstellen

5. Die Figur für den Reifen erstellen

6. Die Hitbox-Figur erstellen

D. Die Katze Körbe werfen lassen

7. Die Figur für den Ball erstellen

8. Den Code für den Ball hinzufügen
9. Einen Treffer erkennen
10. Den Fehler bei der Punktwertung beheben

Das vollständige Programm

Cheat-Modus: Bewegungsloser Reifen

Zusammenfassung

Wiederholungsfragen

## **Kapitel 5: Ein Brick-Breaker-Spiel mit Schliff**

Das Projekt skizzieren

A. Einen Schläger erstellen, der sich nach rechts und links bewegt

1. Die Figur für den Schläger erstellen

B. Einen Ball erstellen, der an den Wänden abprallt

2. Die Figur für den Tennisball erstellen

C. Den Ball vom Schläger abprallen lassen

3. Code für Schlägerberührungen zum Tennisball hinzufügen

D. Klone des Backsteins erstellen

4. Die Backsteinfigur hinzufügen

5. Die Backsteinfigur klonen

E. Den Ball von den Steinen abprallen lassen

6. Code für Ballberührungen zur Backsteinfigur hinzufügen

F. Meldungen für Sieg und Niederlage anzeigen

7. Den Code für den Tennisball ändern

8. Die Figur »Game Over« erstellen

9. Die Figur für die Siegesmeldung erstellen

Das vollständige Programm

Version 2.0: Zeit für den letzten Schliff

Einen coolen Hintergrund zeichnen

Musik hinzufügen

Den Schläger bei Ballberührung blinken lassen  
Das Erscheinen und Verschwinden der Backsteine animieren  
Einen Klangeffekt zum Ausblenden der Steine hinzufügen  
Einen Klangeffekt für den Tennisball hinzufügen  
Eine Spur hinter dem Tennisball herziehen  
Das Erscheinen der Figur »Game Over« animieren  
Das Erscheinen der Siegesmeldung animieren

Zusammenfassung

Wiederholungsfragen

## **Kapitel 6: Asteroidenkacker**

Das Projekt skizzieren

A. Ein Raumschiff erstellen, das umhergestoßen wird

1. Die Figur für das Raumschiff erstellen

B. Für eine umlaufende Bewegung an den Rändern sorgen

2. Den Code für die umlaufende Bewegung zur Raumschifffigur hinzufügen

3. Code für Zufallsbewegungen zur Raumschifffigur hinzufügen

C. Mit der Maus zielen und mit der Leertaste schießen

4. Die Figur für die Energiegeschosse erstellen

D. Umherschwebende Asteroiden erstellen

5. Die Asteroidenfigur erstellen

E. Getroffene Asteroiden in zwei Hälften teilen

6. Code zum Zerteilen der Asteroiden hinzufügen

7. Den Code für die Nachricht »asteroid blasted« hinzufügen

F. Den Punktestand verfolgen und einen Timer erstellen

8. Die Figur »Out of time« erstellen

G. Ein getroffenes Raumschiff explodieren lassen

9. Die Figur für die Explosion hochladen

10. Den Code für die Explosion erstellen

11. Den Explosionscode zur Raumschifffigur hinzufügen

Version 2.0: Begrenzter Munitionsvorrat

Cheat-Modus: Energiespirale

Zusammenfassung

Wiederholungsfragen

## **Kapitel 7: Ein anspruchsvolles Jump-&-Run-Spiel**

Das Projekt skizzieren

A. Die Katze fallen und landen lassen

1. Die Figur für das Gelände erstellen

2. Code zum Fallen und Landen hinzufügen

3. Die Katze horizontal bewegen und vertikal umlaufen lassen

4. Die Verzögerung beim Anheben aus dem Boden beseitigen

B. Die Bewegung an steilen Hängen und Wänden gestalten

5. Den Code für steile Wände hinzufügen

C. Die Katze verschieden hoch springen lassen

6. Den Springcode hinzufügen

D. Decken erkennen

7. Eine niedrige Plattform zur Geländefigur hinzufügen

8. Den Code für die Deckenerkennung hinzufügen

E. Eine Hitbox für die Katzenfigur verwenden

9. Ein Hitboxkostüm zur Katzenfigur hinzufügen

10. Den Hitbox-Code hinzufügen

F. Die Gehanimation verbessern

11. Der Katzenfigur neue Kostüme hinzufügen

12. Einen Block für den Kostümwechsel erstellen

G. Das Level gestalten

13. Das Bühnenbild hinzufügen

14. Das Hitbox-Kostüm für die Figur »Ground« gestalten

15. Den Code für die Figur »Ground« hinzufügen

16. Weiteren Umlaufcode zur Katzenfigur hinzufügen

H. Krabben und Äpfel hinzufügen

17. Die Apfelfigur und den Code dafür hinzufügen

18. Die Krabbenfigur erstellen

19. Die künstliche Intelligenz für die Gegner gestalten

20. Die Figur »Time's up« hinzufügen

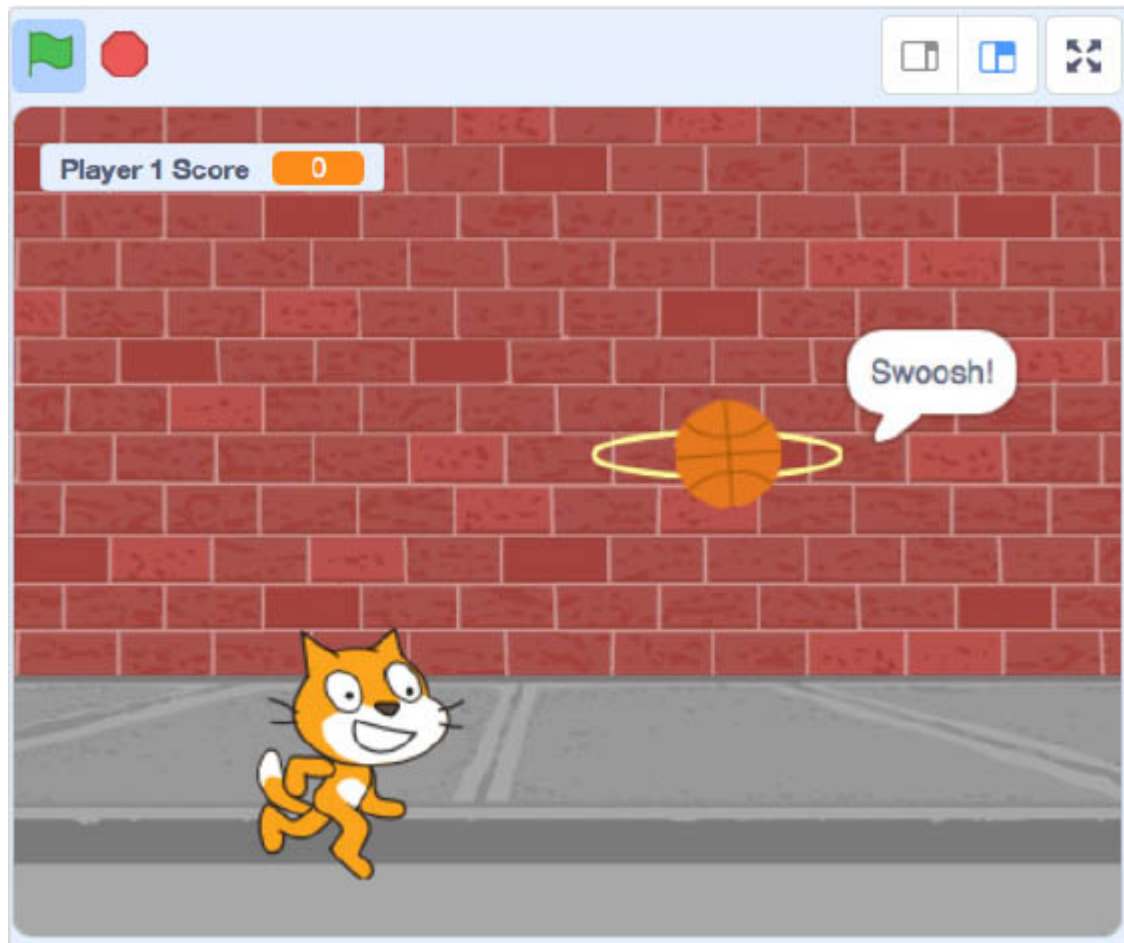
Zusammenfassung

Wiederholungsfragen

**Wie geht es jetzt weiter?**

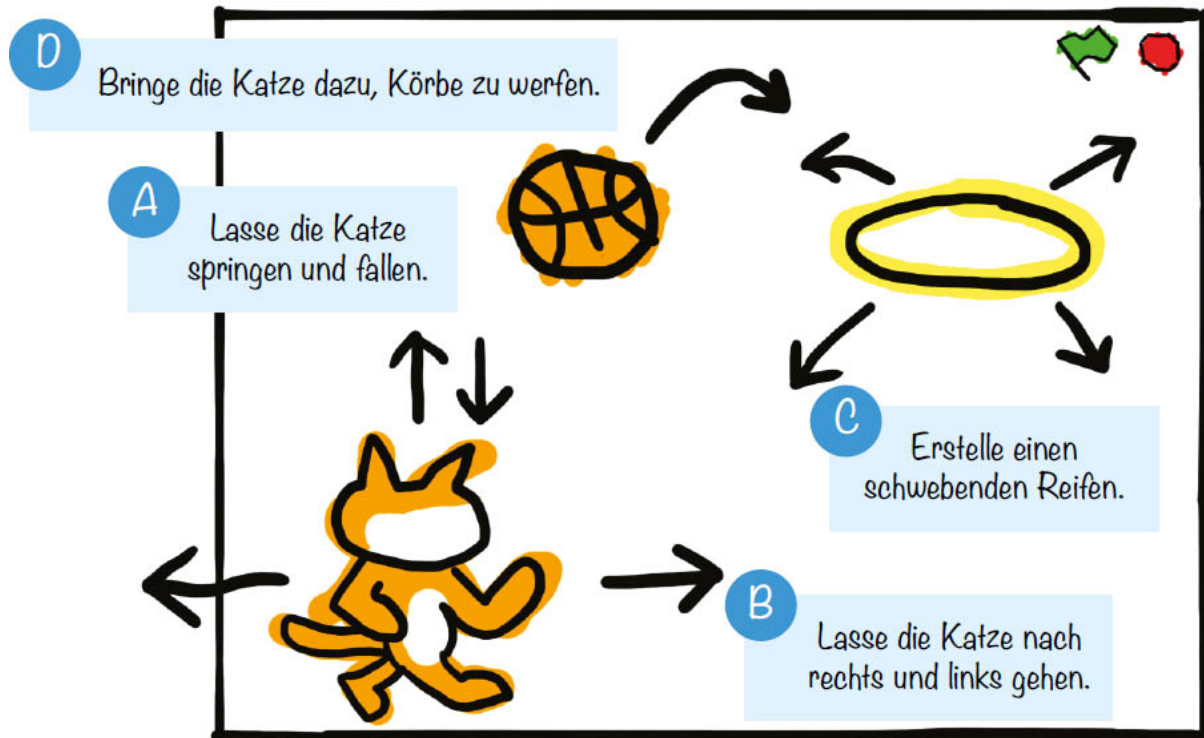
**Index**

Bevor du mit dem Programmieren beginnst, solltest du dir das fertige Spiel auf <https://www.nostarch.com/scratch3playground/> anschauen.



## Das Projekt skizzieren

Als Erstes wollen wir skizzieren, was in dem Spiel passieren soll. Der Spieler steuert die Katze, die sich nach rechts und links bewegen und springen kann. Das Ziel besteht darin, Bälle in einen Ring zu werfen, der sich zufällig auf der Bühne bewegt.



Um Zeit zu sparen, kannst du die Skelettdatei für das Projekt als Ausgangspunkt verwenden. Sie befindet sich als *basketball-skeleton.sb3* in der ZIP-Datei, die du von <https://www.nostarch.com/scratch3playground/> herunterladen kannst, indem du auf den Link *Download Scratch 3 Programming Playground's resources* rechtsklickst und **Ziel speichern als** auswählst. Entpacke anschließend die ZIP-Datei. In der Skelettdatei sind bereits alle Figuren geladen und tragen die im folgenden Abschnitt verwendeten Namen, sodass du nur noch die Codeblöcke zusammenstellen musst.

## A Die Katze springen und fallen lassen

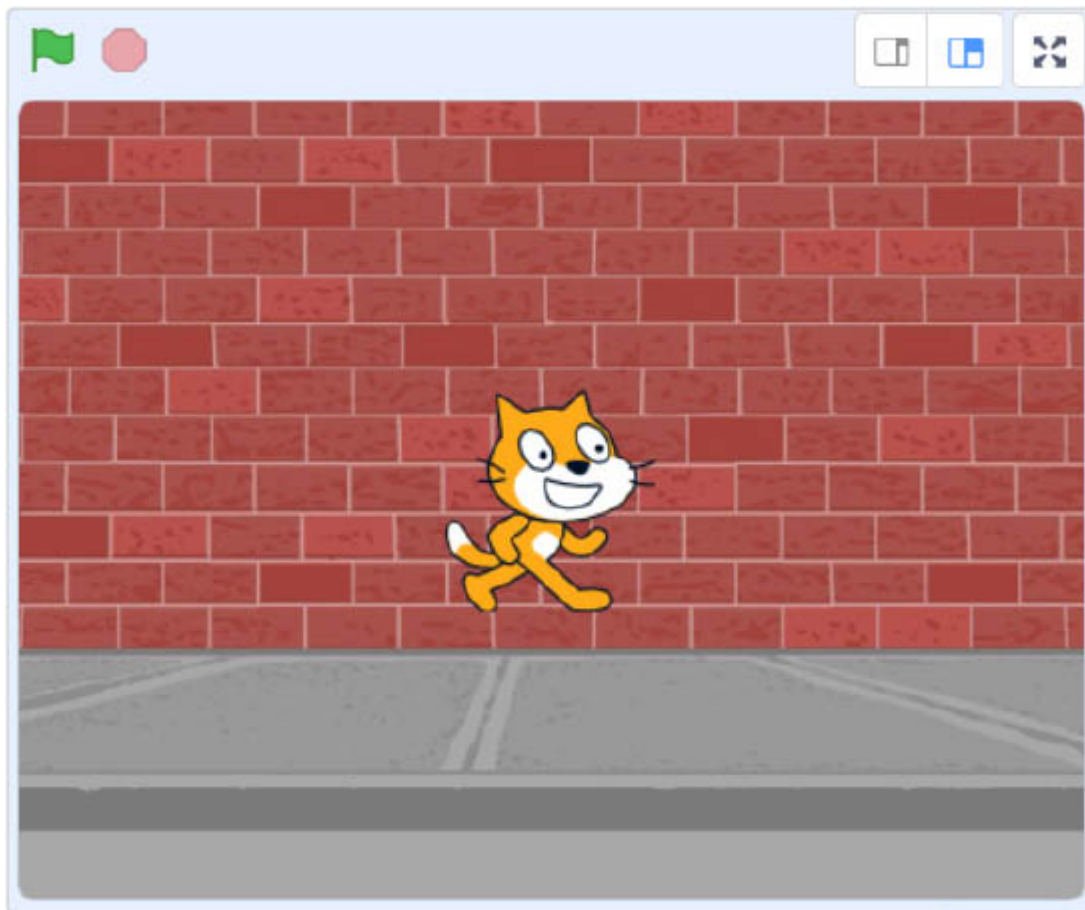
Als Erstes wollen wir für Schwerkraft sorgen, damit die Katze springen und herunterfallen kann.

### 1. Den Schwerkraftcode zur Katze hinzufügen

Benenne im Figuren-Eingabefeld *Figur1* in *Cat* um. Sofern du als Scratcher angemeldet bist, kannst du im Textfeld oben im Scratch-Editor anstelle von *Untitled* den neuen Namen *Basketball* für das Programm eingeben..

Klicke unter *Bühne* auf **Bühnenbild wählen**, um die Bühnenbild-Bibliothek zu öffnen. Wähle darin **Wall 1** aus und klicke auf **OK**. Die Bühne sieht jetzt wie

folgt aus:



Um in dem Programm für Schwerkraft zu sorgen, brauchst du *Variablen*. Eine Variable kannst du dir als einen Behälter zum Speichern von Zahlen oder Text vorstellen, die du später in dem Programm verwenden kannst. Für unser Spiel erstellen wir eine Variable mit einer Zahl, die angibt, wie schnell die Katze fällt.

Wähle dazu als Erstes die Figur *Cat* in der Liste aus und öffne die Registerkarte *Skripte*. Klicke dann in der orangefarbenen Gruppe *Variablen* auf die Schaltfläche **Neue Variable**. Dadurch wird das gleichnamige Fenster eingeblendet. Gib als Variablenname *y velocity* ein (»y-Geschwindigkeit«). Diese Variable besagt, wie schnell sich etwas nach oben oder unten bewegt. Ist *y velocity* eine positive Zahl, bewegt sich die Katze nach oben, ist es eine negative Zahl, sinkt sie nach unten. Vergewissere dich, dass **Nur für diese Figur** ausgewählt ist. (Wenn *Für alle Figuren* ausgewählt ist, hast du die Bühne markiert, nicht die Katze!) Klicke anschließend auf **OK**.

Neue Variable

Neuer Variablenname:

y velocity

Für alle Figuren  Nur für diese Figur

Abbrechen OK

Achte darauf, dass  
»Nur für diese Figur«  
ausgewählt ist!

Jetzt erscheinen in der Gruppe *Variablen* mehrere neue Blöcke. Einer davon ist der abgerundete Variablenblock *y velocity*.

## Variablen

Neue Variable



y velocity

setze y velocity auf 0

ändere y velocity um 1

zeige Variable y velocity

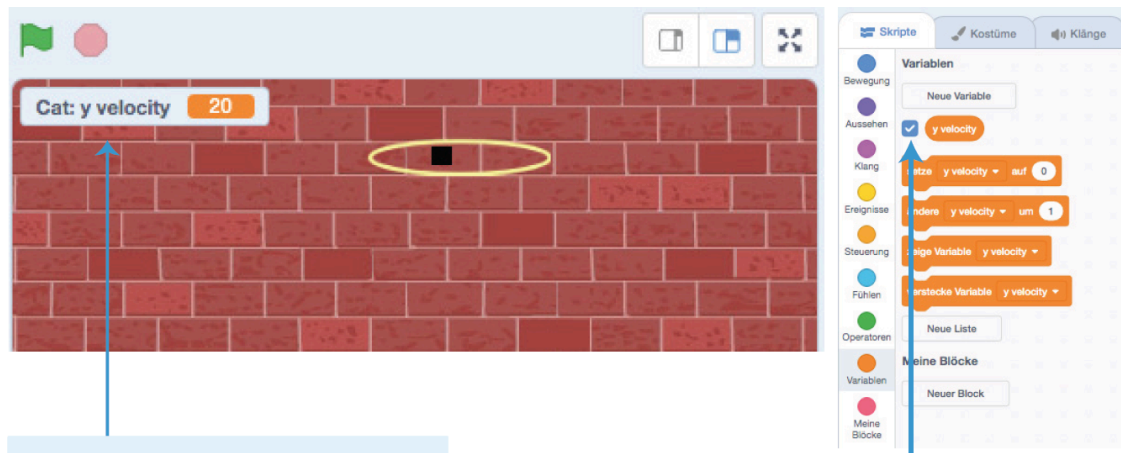
verstecke Variable y velocity

### »Für alle Figuren« und »Nur für diese Figur«

Als du die Variable *y velocity* erstellt hast, musstest du **Nur für diese Figur** auswählen. Diese Einstellung sorgt dafür, dass die Variable, die du anlegst, nur von der Katze verwendet werden kann. Mit *Für alle Figuren* dagegen erstellst du Variablen, die alle Figuren nutzen können.

Um zu prüfen, was für eine Art von Variable du angelegt hast, setzt du ein Häkchen in das Kästchen neben dem abgerundeten Variablenblock in

der Palette, sodass die Variable auf der Bühne angezeigt wird. Hast du *Nur für diese Figur* ausgewählt, erscheint vor dem Variablennamen der Name der Figur. Bei *Für alle Figuren* dagegen wird nur der Variablennamen angegeben.



Die Angabe des Figurennamens *Cat* vor dem Variablennamen *y velocity* zeigt, dass dies eine Variable nur für eine bestimmte Figur ist.

Setze ein Häkchen in dieses Kästchen, um die Variable auf der Bühne anzuzeigen.

Sollte der Figurenname *Cat* nicht vor dem Variablennamen angezeigt werden, klickst du mit der rechten Maustaste in der orangefarbenen Gruppe *Variablen* auf den Block **y velocity** und wählst **Variable löschen**. Erstelle die Variable *y velocity* dann neu, aber achte diesmal darauf, dass **Nur für diese Figur** ausgewählt ist.

Du kannst den Block *y velocity* und auch jeden anderen Variablenblock überall dort platzieren, wo du eine Zahl oder Text eingeben kannst. Der Codeblock, in den du die Variable einfügst, verwendet dann die in der Variable gespeicherte Zahl bzw. den Text. Wenn du Variablen verwendest, kannst du die darin gespeicherten Zahlen oder Texte *im laufenden Programm* ändern.

Um einen Wert in deiner Variable zu speichern, verwendest du den orangefarbenen Block **setze ... auf**. Wenn du beispielsweise die Variable *Begrüßung* erstellt hast, kannst du mit **setze ... auf** den Wert *Hallo!* darin aufnehmen. Dann kannst du *Begrüßung* in einen **sage**-Block einfügen, was das Gleiche bewirkt, als würdest du *Hallo!* direkt in diesen Block schreiben. (Füge diese Blöcke nicht zu dem Programm *Basketball* hinzu, und erstelle hier auch nicht die Variable *Begrüßung* – dies ist nur ein Beispiel!)

Mit dem Block **setze ... auf** speicherst du den Wert **Hallo!** in der Variable **Begrüßung**.



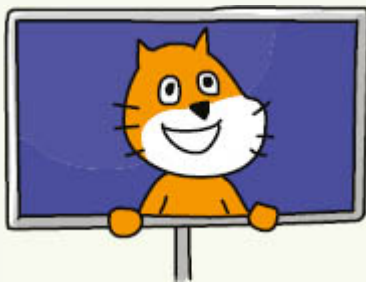
Diese beiden Blöcke bewirken das Gleiche.

Um den Text der Begrüßung im Programmverlauf zu ändern, fügst du einen weiteren **setze ... auf**-Block hinzu. Bei Variablen, die Zahlen enthalten, kannst du auch den Block **ändere ... um** verwenden, um etwas von der Zahl abzuziehen oder zu ihr zu addieren.

Die Schwerkraft sorgt dafür, dass ein Objekt nach unten *beschleunigt*. Wenn die Katzenfigur im Spiel nach unten sinkt, muss sich diese Sinkgeschwindigkeit also im Programmverlauf ändern. Füge den folgenden Code zur Figur *Cat* hinzu, um in deinem Scratch-Programm die Schwerkraft wirken zu lassen. Das ist die Mindestmenge an Code, die du brauchst, damit eine Figur unter Schwerkrafteinfluss fällt. Du kannst ihn zu jeder Figur hinzufügen, die fallen muss.



Wenn du auf die grüne Fahne klickst, wird die Variable *y velocity* auf 0 gesetzt. Dann tritt das Skript in eine Endlosschleife ein. Darin wird die y-Position (die vertikale Position) der Katze um den Wert von *y velocity* geändert und *y velocity* um  $-2$ . Mit jedem Schleifendurchlauf ändert sich die y-Position schneller, sodass die Katze immer schneller fällt.



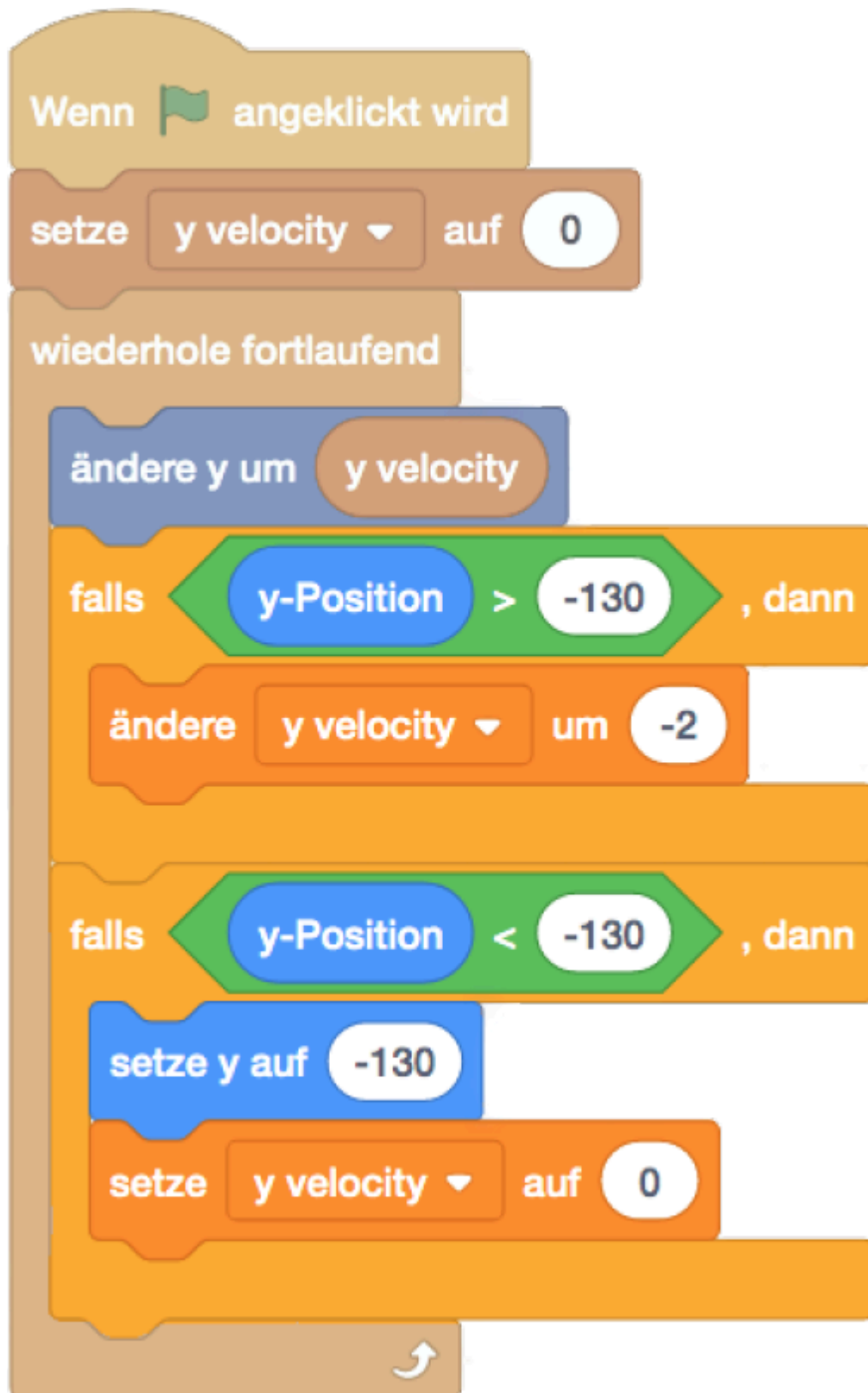
### Zeit zu speichern!

Ziehe als Erstes die Katzenfigur zum oberen Bildschirmrand. Klicke dann auf die grüne Fahne und beobachte, wie die Katze fällt. Um sie abermals fallen zu lassen, musst du auf das rote Stoppschild klicken, die Katze

wieder zum oberen Bühnenrand ziehen und erneut auf die grüne Fahne klicken. Speichere dein Programm.

## 2. Code für die Landung auf dem Boden hinzufügen

Die Katze fällt jetzt zwar, aber wir wollen auch, dass sie damit aufhört, wenn sie den Boden erreicht. Dazu ergänzen wir den Code für die Figur *Cat* wie folgt:



In diesem Code verwenden wir als **y-Position** des Bodens den Wert  $-130$ . Ist die y-Position der Katze größer als  $-130$  (befindet sie sich also über dem Boden), so wird *y velocity* um  $-2$  geändert, und die Katze fällt weiter. Schließlich fällt die Katze aber unter die Bodenebene, sodass ihre y-Position kleiner als  $-130$  ist.

Wenn das geschieht, wird die Figur auf die y-Position  $-130$  zurückgesetzt und *velocity* auf  $0$  gesetzt, damit die Katze nicht weiter fällt.



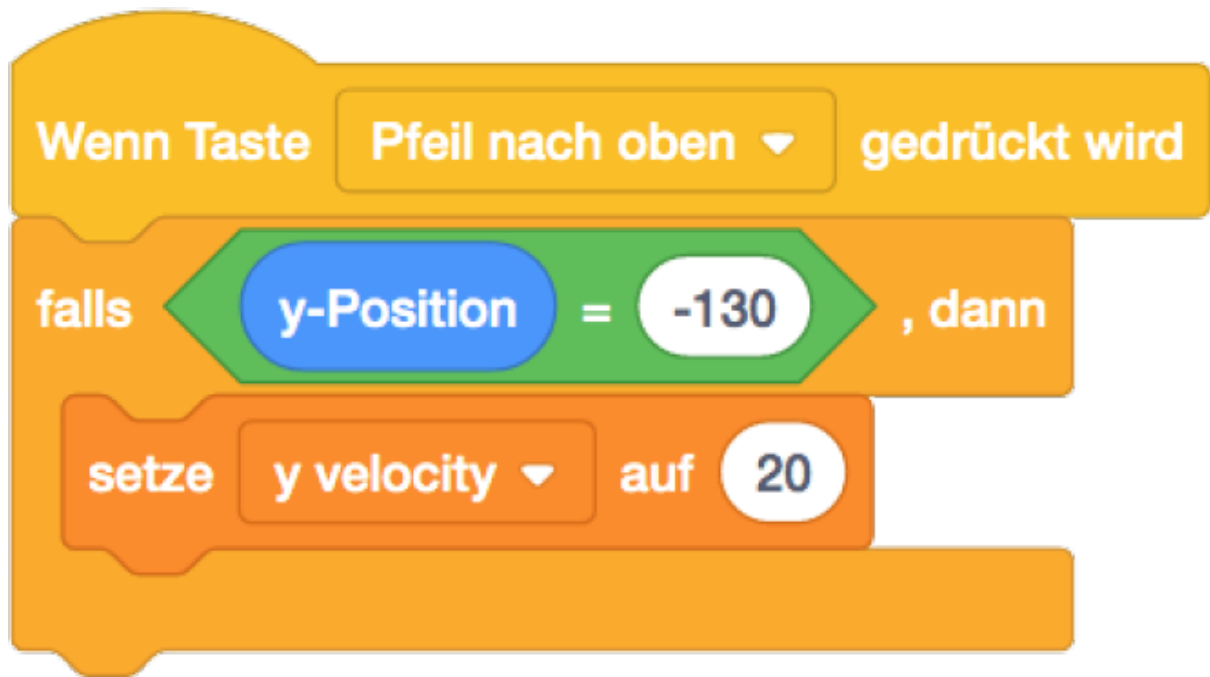
### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Ziehe die Katze mit der Maus nach oben und lass sie fallen. Vergewissere dich, dass sie bis zum Boden fällt, aber nicht über den unteren Bühnenrand hinaus sinkt. Du kannst auch verschiedene andere Bodenhöhen ausprobieren, indem du statt  $-130$  eine andere Zahl verwendest. Klicke auf das rote Stoppschild und speichere dein Programm.

Wenn du mit dem Testen fertig bist, solltest du die Variable *y velocity* auf der Bühne unsichtbar machen, indem du in der orangefarbenen Gruppe *Variablen* das Häkchen aus dem Kästchen neben dem Variablenblock nimmst.

### 3. Den Springcode zur Katze hinzufügen

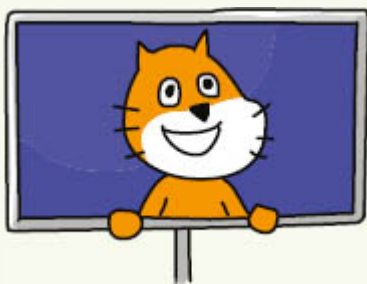
Da die Katze jetzt über Schwerkraftcode verfügt, ist es einfach, sie springen zu lassen. Füge der Figur *Cat* folgenden Code hinzu:



Wenn du jetzt die Taste mit dem nach oben weisenden Pfeil drückst, wird *y velocity* auf die positive Zahl 20 gesetzt, sodass die Figur nach oben springt. Allerdings wird die Variable *y velocity* immer noch bei jedem Durchlauf der Endlosschleife um 2 verringert. Die Katze steigt also zunächst 20 Einheiten in die Höhe, beim nächsten Durchlauf nur noch 18, dann 16 usw. Der Block **falls ... dann** prüft, ob sich die Katze überhaupt am Boden befindet, denn es sollte nicht möglich sein, die Katze zum Springen zu veranlassen, wenn sie bereits in der Luft schwebt.



Wenn *y velocity* auf 0 gesetzt wird, hat die Katze den höchsten Punkt ihrer Flugbahn erreicht. Da die Variable bei jedem weiteren Durchlauf um  $-2$  geändert wird, beginnt die Katze anschließend wieder zu fallen, bis sie den Boden berührt. Probiere verschiedene Zahlenwerte für die Blöcke **setze *y velocity* auf** und **ändere *y velocity* um** aus. Finde heraus, wie du die Katze höher oder weniger hoch springen lassen kannst (aber immer oberhalb des Bodens) oder wie du die Schwerkraft stärker und schwächer machst.

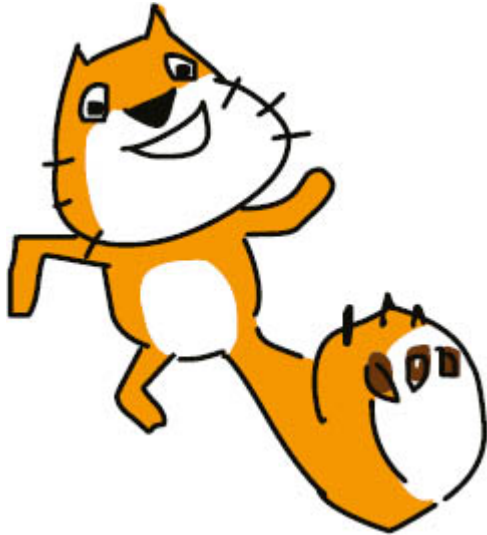


### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Drücke die Taste mit dem Pfeil nach oben und vergewissere dich, dass die Katze nach oben springt und wieder nach unten fällt. Klicke auf das rote Stoppschild und speichere dein Programm.

## **B** Die Katze nach rechts und links gehen lassen

Als Nächstes fügen wir der Figur *Cat* Code hinzu, sodass der Spieler sie mithilfe der Tastatur nach rechts und links gehen lassen kann.



### **4. Gehcode zur Katze hinzufügen**

Füge unterhalb des Codes für die Figur *Code* folgenden weiteren Code hinzu:



In der Schleife **wiederhole fortlaufend** prüft das Programm, ob die Tasten mit dem Pfeil nach links oder rechts gedrückt wurden. Wenn ja, wechselt die Katzenfigur zum nächsten Kostüm und ändert ihre x-Position um  $-10$  (nach links) bzw.  $10$  (nach rechts). Die Figur *Cat* hat zwei Kostüme, die du dir anschauen kannst, indem du auf die Registerkarte *Kostüme* über der Blockpalette klickst. Dadurch, dass die Katze im Block **nächstes Kostüm** ständig das Kostüm wechselt, sieht es so aus, als würde sie gehen.



### Zeit zu speichern!

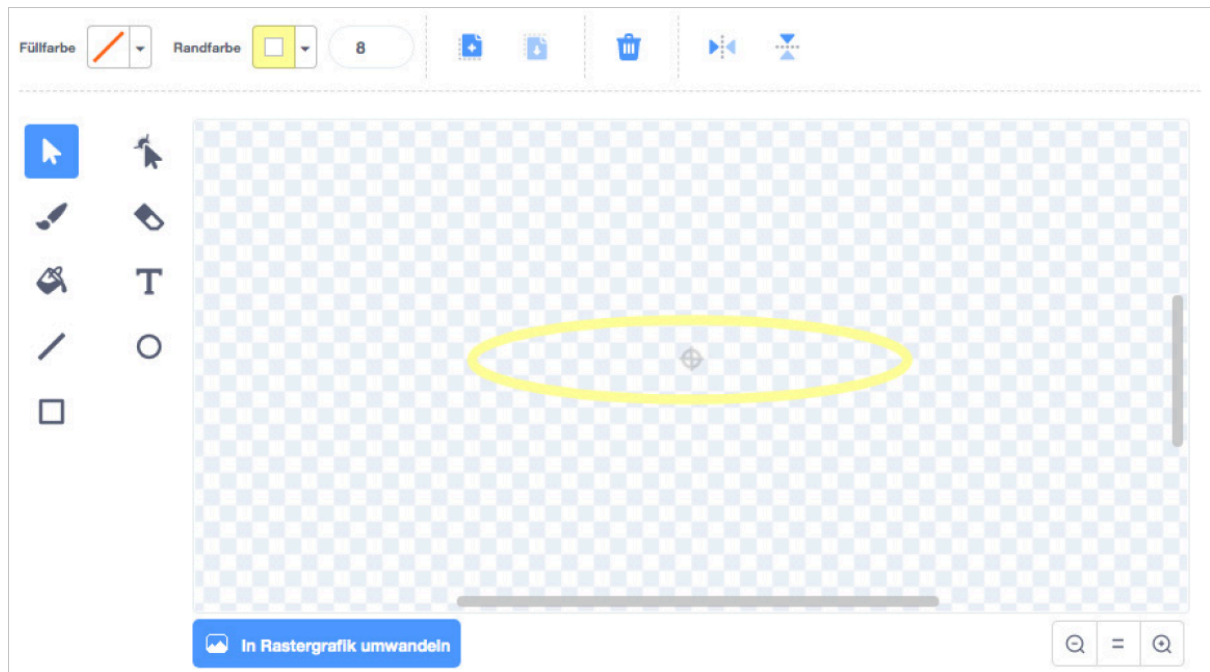
Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Drücke die Tasten mit dem Pfeil nach links und rechts und vergewissere dich, dass die Katze in die richtige Richtung geht. Wenn die Katze sich nach links bewegt, sieht es so aus, als ginge sie rückwärts; das ist aber so in Ordnung. Klicke auf das rote Stoppschild und speichere dein Programm.

## C Einen schwebenden Reifen erstellen

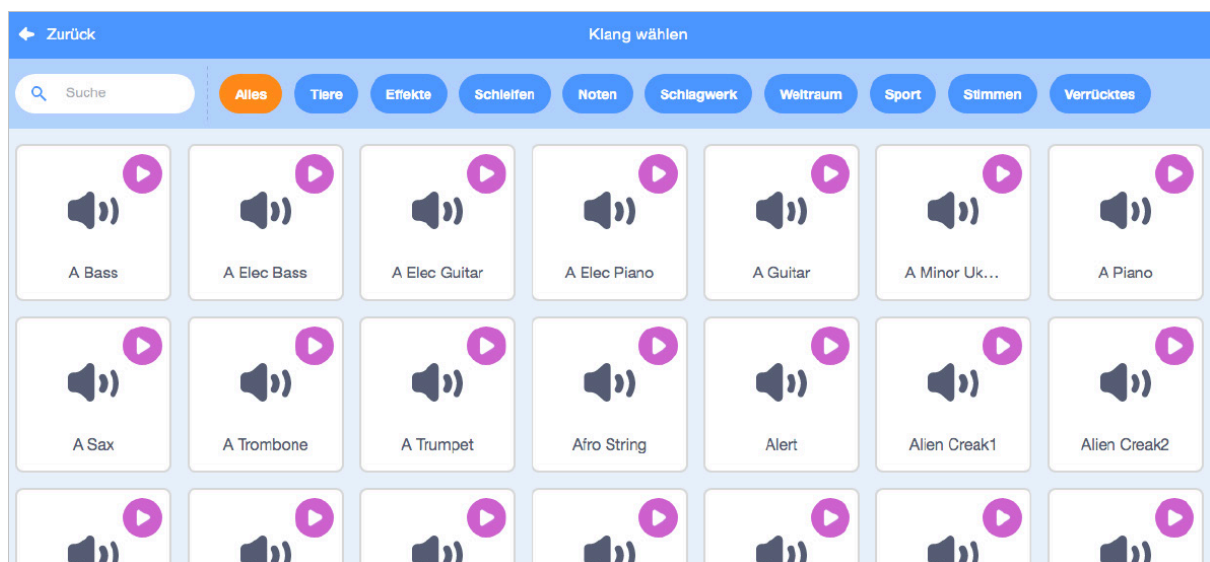
Nachdem wir nun die Figur *Cat* fertiggestellt haben, gehen wir zur nächsten Figur über, nämlich dem Reifen, durch den wir die Bälle werfen wollen.

### 5. Die Figur für den Reifen erstellen

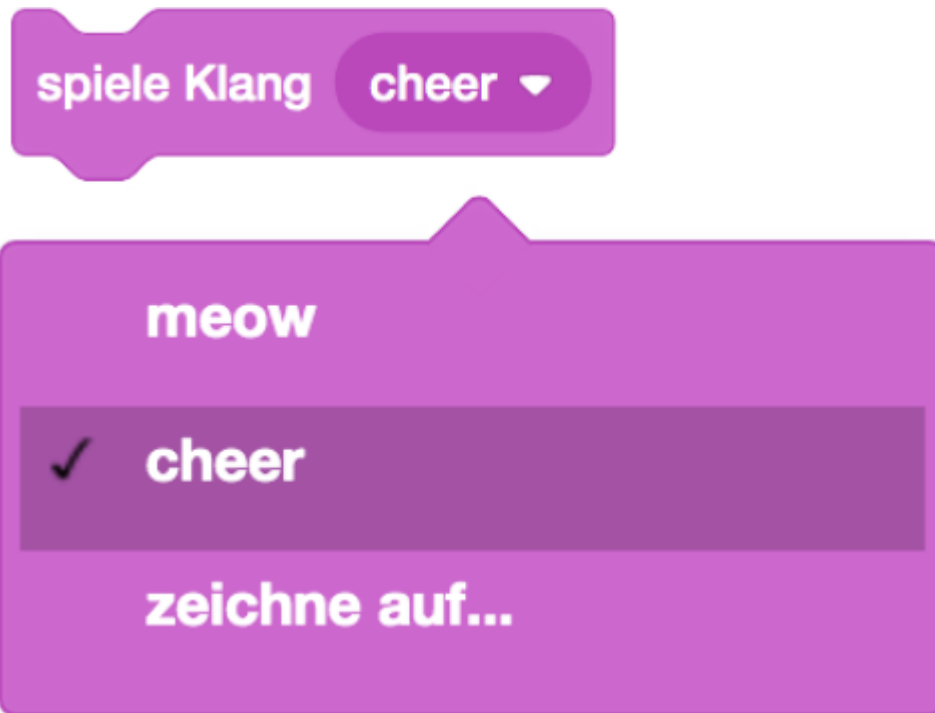
Klicke auf die Schaltfläche **Malen**, die erscheint, sobald du den Mauszeiger auf den Button **Figur wählen** bewegst oder darauf tippst. Die Schaltflächen der Zeichenwerkzeuge erscheinen auf der linken Seite des Zeichenbereichs. Wähle eine gelbe Farbe und verwende das Ellipse-Werkzeug, um einen Kreis zu zeichnen. Um die Linie des Kreises dicker zu machen, erhöhe die Zahl im Eingabefeld neben den Feldern Füll- und Randfarbe. Achte darauf, dass sich das Fadenkreuz des Zeichenbereichs im Mittelpunkt der Ellipse befindet.



Gib der Figur im Figuren-Eingabefenster den Namen *Hoop* (»Reifen«). Diese Figur soll einen Jubelklang abspielen, wenn der Spieler einen Korb wirft. Daher laden wir als Nächstes den Klang *cheer*. Klicke oberhalb der Blockpalette auf die Registerkarte **Klänge** und dann, über den Button mit dem Lautsprechersymbol *Klang wählen*, auf die gleichnamige Schaltfläche. In dem Auswahlfenster **Klang wählen** wählst du dann *cheer* aus und klickst auf **OK**.



Der Klang *cheer* erscheint jetzt als Auswahlmöglichkeit im Block **spiele Klang**, den wir im Folgenden zu der Figur *Hoop* hinzufügen.



Füge *Hoop* den folgenden Code hinzu, damit die Figur zufällig über die obere Hälfte der Bühne gleitet und bei einem Treffer den Jubelklang abspielt. In dem zweiten Skript musst du dafür sorgen, dass eine Nachricht empfangen werden kann. Klicke dazu auf das schwarze Dreieck des Blocks **Wenn ich ... empfangen** und wähle **Neue Nachricht**. Gib als Namen für die Nachricht *swoosh* ein.

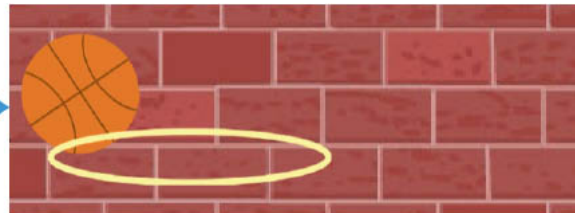


Skript 1 sorgt dafür, dass der Reifen in jeder Sekunde an eine neue Position schwebt. Ein beweglicher Reifen macht das Spiel viel kniffliger und spannender! Skript 2 spielt den Klang *cheer* ab und zeigt »Swoosh!« an, wenn es die Nachricht *swoosh* empfängt. Wir werden später dafür sorgen, dass die Figur für den Ball diese Nachricht sendet, wenn sie im Korb landet.

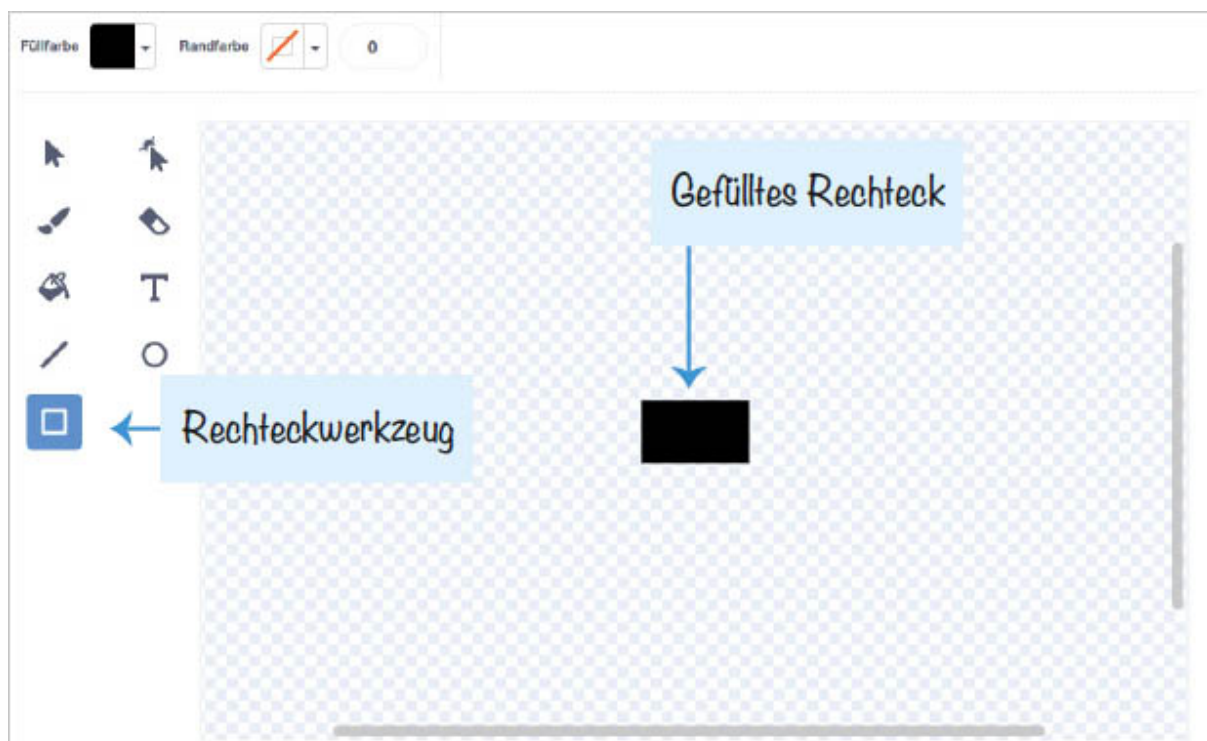
## 6. Die Hitbox-Figur erstellen

Als Nächstes müssen wir überlegen, wie der Code erkennen kann, ob der Spieler einen Korb geworfen hat. Wir könnten zwar ein Programm schreiben, das prüft, ob der Ball den Reifen *berührt*, aber dabei würde auch schon ein einfaches Anstoßen an den Ring als Korb zählen. In Wirklichkeit wollen wir aber nur dann einen Treffer anerkennen, wenn der Ball durch die Mitte des Reifens geht. Aber wie machen wir das?

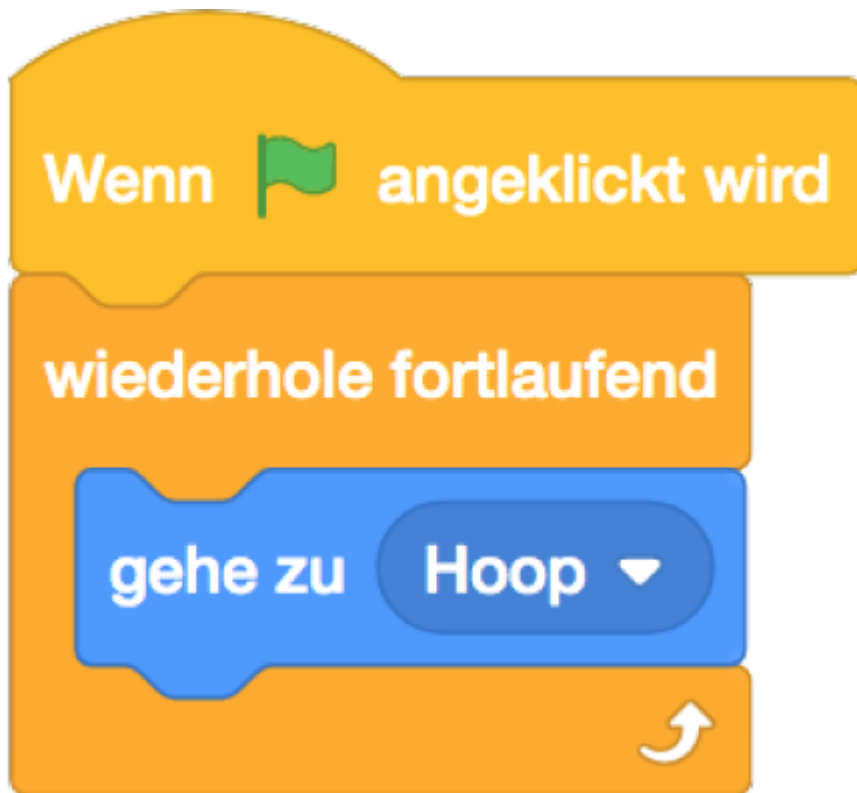
Wenn wir einfach prüfen, ob der Ball den Reifen berührt, würde auch dies als Korb gewertet. So wird Basketball aber nicht gespielt!



Eine mögliche Lösung besteht darin, eine *Hitbox* zu verwenden. Das ist ein Begriff aus der Spieleentwicklung für einen rechteckigen Bereich, der herangezogen wird, um zu bestimmen, ob zwei Spielobjekte miteinander zusammengestoßen sind. Zeichnen wir also eine Hitbox-Figur! Klicke auf die Schaltfläche **Malen**, die erscheint, sobald du auf den Button **Figur wählen** gehst. Male mit dem Rechteckwerkzeug und der Auswahl für ein gefülltes Rechteck ein kleines schwarzes Quadrat auf das Fadenkreuz. Nenne die Figur *Hitbox*. Sie sieht wie folgt aus:

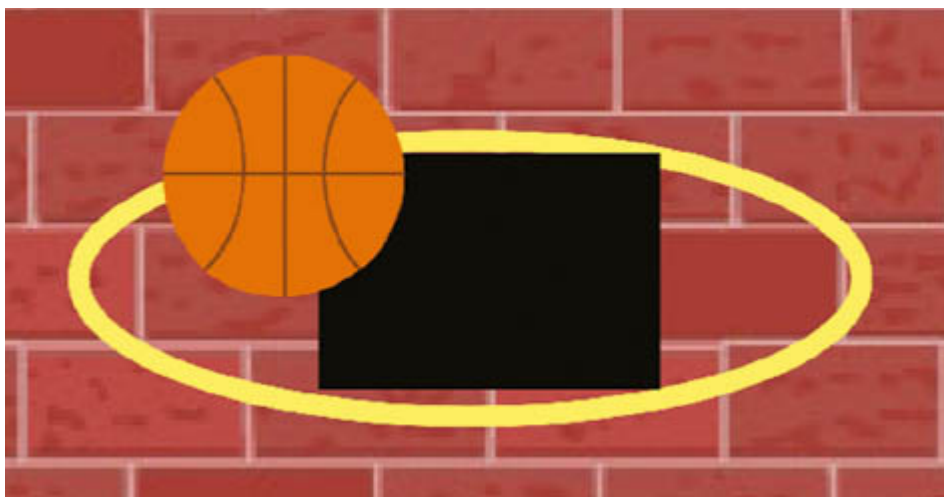


Füge der Figur *Hitbox* den folgenden Code hinzu:



Dadurch folgt die Hitbox dem Reifen, wohin auch immer dieser sich bewegt.

Das Skript, das wir in Schritt 9 schreiben werden, sorgt dafür, dass nur dann ein Treffer gewertet wird, wenn der Ball die Figur *Hitbox* trifft, nicht *Hoop*. Der Ball muss also viel näher an die Mitte des Reifens herankommen, damit ein Korbwurf gewertet wird.

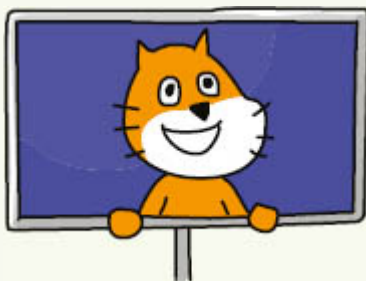


Es würde allerdings ziemlich merkwürdig aussehen, wenn wir das schwarze Viereck in der Mitte des Reifens beließen. Daher wollen wir die Hitbox unsichtbar machen. Füge den Block **setze Effekt Durchsichtigkeit aufzu** der Figur *Hitbox* hinzu und gib als Wert 100 an (in der Palette erscheint dieser Block zunächst als

setze Effekt Farbe auf, anstatt »Farbe« kann dann »Durchsichtigkeit« eingestellt werden).



Es gibt einen Unterschied zwischen den Blöcken **verstecke dich** und **setze Durchsichtigkeit-Effekt auf**. Würdest du die Hitbox mit **verstecke dich** unsichtbar machen, dann könnten die *wird ... berührt*-Blöcke, die wir später verwenden, niemals eine Berührung zwischen dem Ball und der Hitbox erkennen und der Spieler könnte keinen Treffer erzielen. Dagegen macht **setze Durchsichtigkeit-Effekt auf** die Figur zwar ebenfalls unsichtbar, lässt es aber weiterhin zu, dass die *wird ... berührt*-Blöcke ihre Anwesenheit wahrnehmen.



### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Vergewissere dich, dass der Reifen über die Bühne wandert und dass sich

die Hitbox immer in seiner Mitte befindet. Klicke auf das rote Stoppschild und speichere dein Programm.

## D Die Katze Körbe werfen lassen

Als Nächstes fügen wir einen Ball hinzu, den die Katze werfen kann. Ebenso wie die Katze braucht auch der Ball Schwerkraftcode, damit er zu Boden fallen kann.

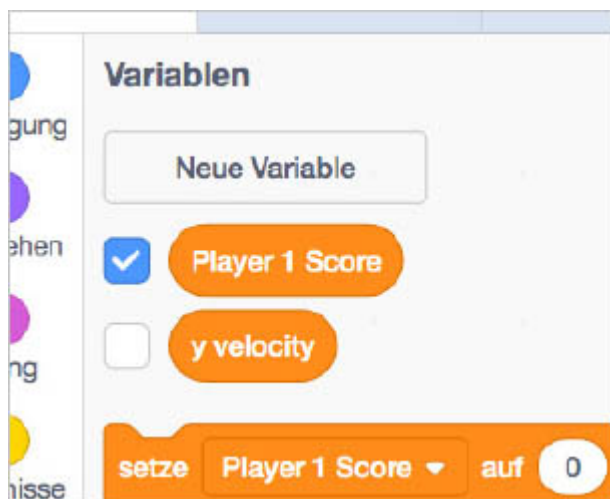
### 7. Die Figur für den Ball erstellen

Klicke im Bereich unten rechts auf die Schaltfläche **Figur wählen**, um die Figurenbibliothek zu öffnen. Wähle dort **Basketball** aus.

Anschließend klickst du oberhalb der Blockpalette auf die Registerkarte **Klänge** und dort im Bereich links unten auf **Klang wählen**, um die Klangbibliothek zu öffnen. Wähle den Klang *pop* aus. Öffne dann wieder den Skriptbereich, indem du über der Blockpalette auf **Skripte** klickst.

Als Nächstes erstellst du in der orangefarbenen Gruppe *Variablen* zwei neue Variablen. Klicke auf die Schaltfläche **Neue Variable** und nenne die Variable *y velocity*. Vergewissere dich, dass **Nur für diese Figur** ausgewählt ist, bevor du auf **OK** klickst. Da die Variablen jeweils nur für die vorliegende Figur gelten, handelt es sich bei *y velocity* für die Katze und für den Ball um zwei verschiedene Variablen, auch wenn sie denselben Namen haben.


Klicke erneut auf **Neue Variable**, um eine weitere Variable namens *Player 1 Score* («Punktestand Spieler 1») zu erstellen. Diesmal aber wählen wir die Einstellung **Für alle Figuren**. (Wir schreiben *Player 1 Score* groß, da wir die Variable auf der Bühne anzeigen wollen.) Nimm das Häkchen aus dem Kästchen neben *y velocity*, um diese Variable auf der Bühne zu verbergen. Jetzt erscheinen die neuen Variablenblöcke in der Gruppe *Variablen*.



## 8. Den Code für den Ball hinzufügen

Mit dem Klang *pop* und den beiden Variablen kannst du nun den folgenden Code zur Figur *Basketball* hinzufügen:

1

```
Wenn  angeklickt wird
  setze Player 1 Score auf 0
  verstecke dich
```

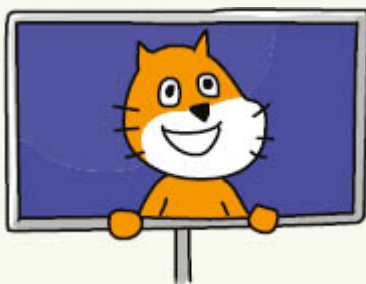
2

```
Wenn Taste Leertaste gedrückt wird
  spiele Klang pop
  gehe zu Cat
  setze y velocity auf 24
  zeige dich
  wiederhole bis  $y\text{-Position} < -130$ 
    ändere x um 8
    ändere y um 10
    ändere y velocity um -2
    drehe dich um 6 Grad
  verstecke dich
```

Skript ① sorgt dafür, dass der Spieler zu Anfang einen Punktestand von 0 hat, und versteckt den Basketball.

Der Code in Skript ② ähnelt dem für die Katze. Wenn der Spieler die Leertaste drückt, erscheint der Ball vor der Katze und beginnt sich weiter nach vorn und nach oben zu bewegen. Ebenso wie die Variable *y velocity* der Katze bei einem Sprung wird hier die Variable *y velocity* des Balls auf eine positive Zahl gesetzt, damit es so aussieht, als ob die Katze den Ball nach oben wirft.

Der Block **wiederhole bis y-position < -130** sorgt dafür, dass der Ball fällt, bis er den Boden berührt. Danach wird der Ball wieder versteckt, bis der Spieler erneut die Leertaste drückt.



### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Drücke die Leertaste, damit die Katze den Ball wirft. Vergewissere dich, dass der Ball wieder verschwindet, wenn er den Boden berührt. Klicke auf das rote Stoppschild und speichere dein Programm.

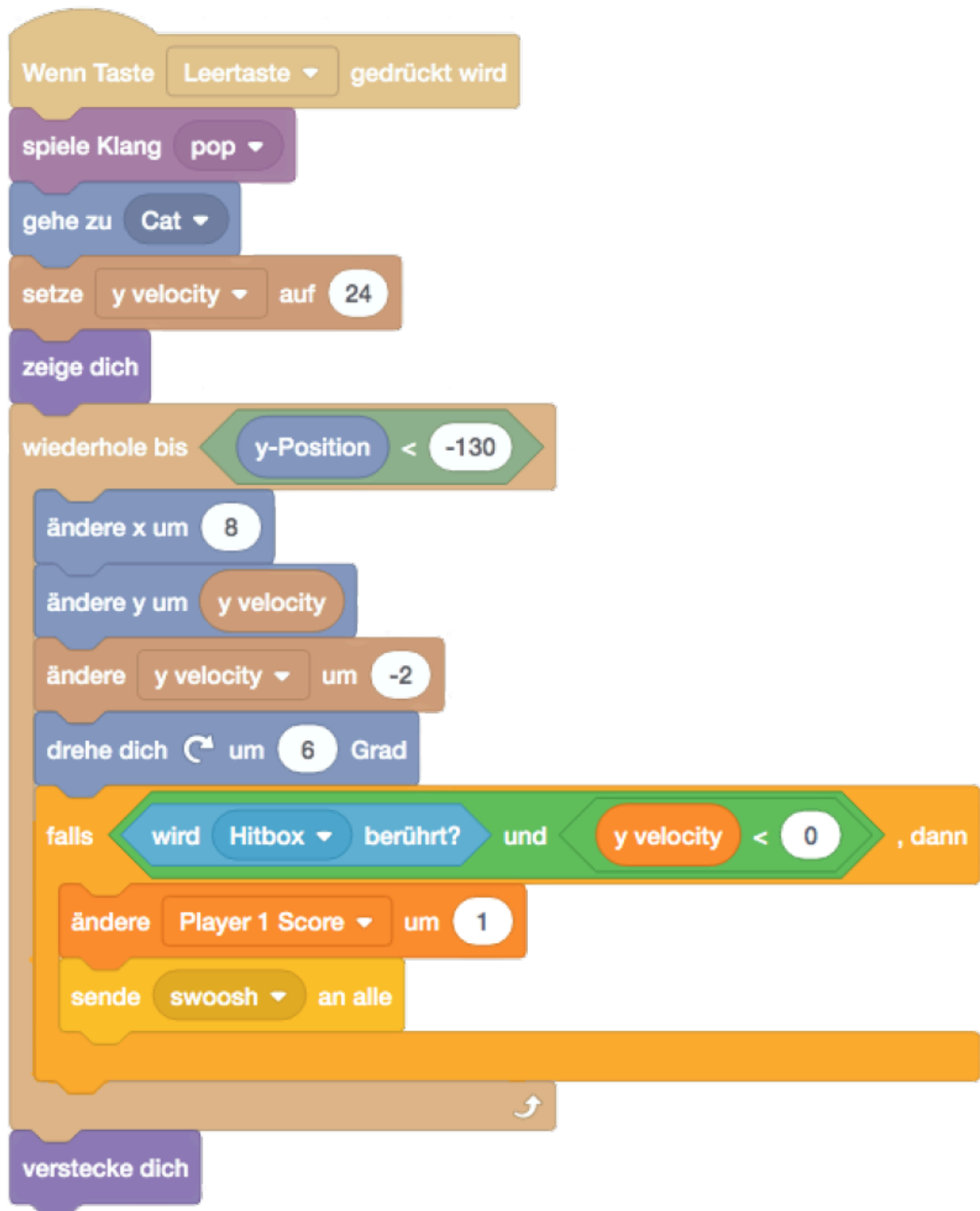
## 9. Einen Treffer erkennen

Als Nächstes fügst du den Code hinzu, um zu prüfen, ob der Ball die Hitbox getroffen hat. Dadurch kannst du erkennen, ob der Spieler einen Korb geworfen hat. Wenn ja, erhöhst du den Wert der Variable *Player 1 Score*. Es gibt jedoch einen Haken an der Sache: Es wird kein Korb gezählt, wenn der Ball den Reifen von unten durchdringt.

Denke daran, dass der Block **ändere y um y velocity** den Ball nach oben bewegt, wenn die Variable *y velocity* positiv ist. Ist *y velocity* gleich 0, bewegt sich der Ball gar nicht, und hat die Variable einen negativen Wert, fällt der Ball nach unten.

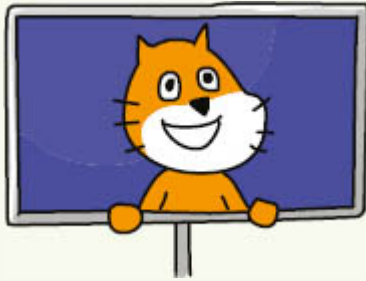
Daher musst du dem Code für die Figur *Basketball* einen weiteren **falls ... dann**-Vergleich hinzufügen. Der Punktestand soll nur dann erhöht werden, wenn

der Ball die Hitbox trifft (**wird Hitbox berührt?**) und nach unten fällt (**y velocity < 0**).



Der **und**-Block kombiniert zwei Bedingungen. Damit Scratch die Codeblöcke innerhalb des **falls ... dann**-Blocks ausführt, müssen *beide* Bedingungen erfüllt sein. Es reicht nicht, dass die Figur die Hitbox berührt oder dass *y velocity* kleiner 0 ist. Sowohl **wird Hitbox berührt?** als auch **y velocity < 0** müssen wahr sein.

Nur wenn das der Fall ist, wird die Variable *Player 1 Score* um 1 erhöht und die Nachricht *swoosh* gesendet.



### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Wirf einige Bälle. Die Variable *Player 1 Score* sollte nur dann erhöht werden, wenn der Ball die Reifenmitte berührt und sich dabei abwärts bewegt. Außerdem sollte dann die Sprachblase »Swoosh!« angezeigt und der Jubelklang abgespielt werden. Klicke auf das rote Stoppschild und speichere dein Programm.

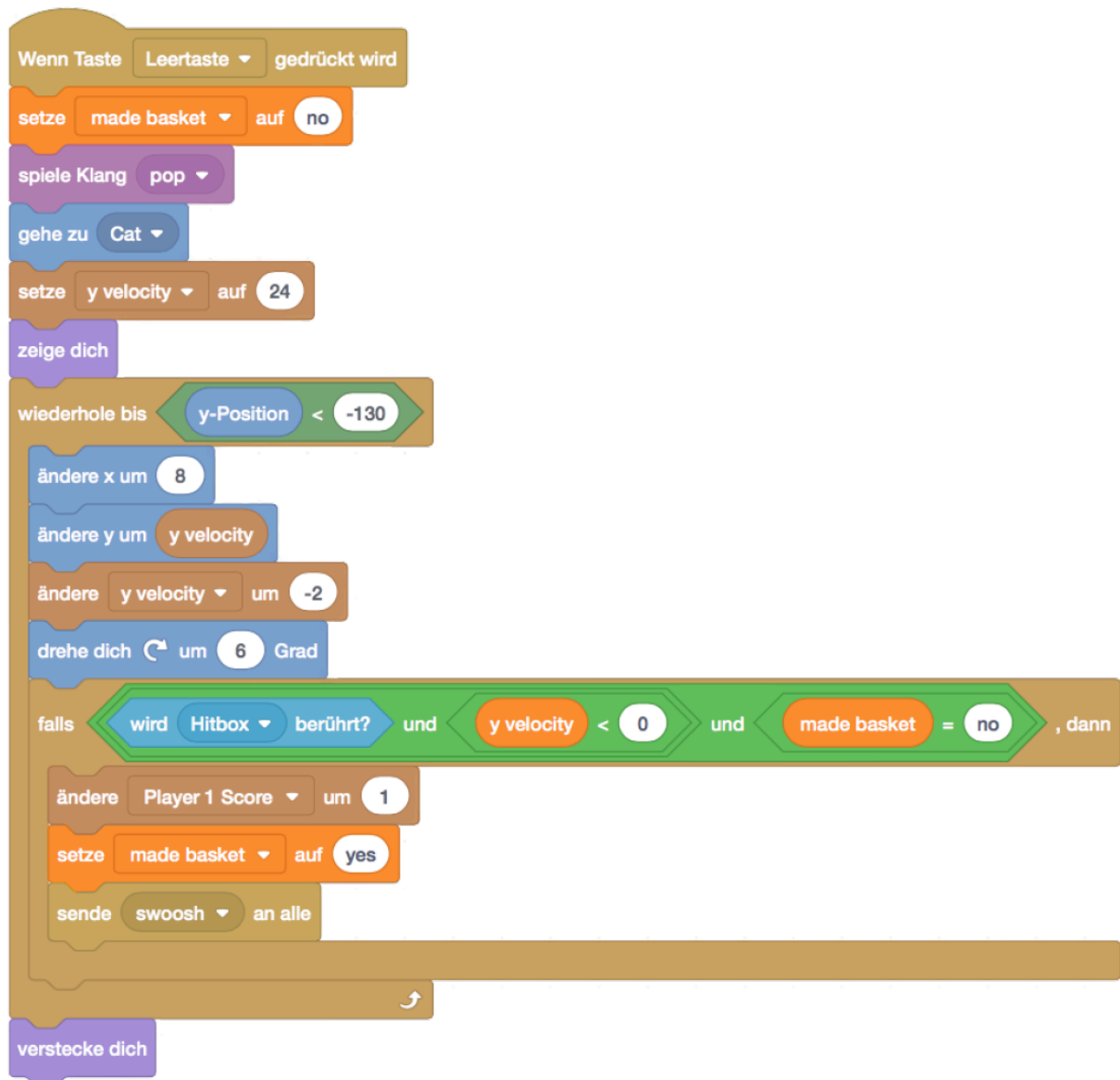
## 10. Den Fehler bei der Punktwertung beheben

Ist dir aufgefallen, dass *Player 1 Score* bei einem einzigen Korb um mehrere Punkte erhöht wird? Das ist ein *Bug*, also ein Problem, das dafür sorgt, dass sich das Programm auf unerwartete Weise verhält. Um herauszufinden, warum das passiert, müssen wir uns den Code noch einmal genau anschauen.

Die Schleife **wiederhole bis** wird ständig durchlaufen, bis der Ball den Grund berührt. Der gesamte Code darin gilt also für einen einzigen Wurf. Dabei prüft der Block aber mehrmals, ob der Ball die Hitbox berührt und dabei nach unten fällt. *Player 1 Score* sollte aber nur beim ersten Eintreten beider Bedingungen erhöht werden.

Um diesen Bug zu beheben, erstellst du eine neue Variable, die sich merkt, ob der Ball während eines Wurfs bereits einmal den Reifen berührt hat. Dadurch kannst du sicherstellen, dass der Spieler immer nur einen Punkt pro Wurf erzielen kann.

Klicke in der Blockpalette auf die orangefarbene Gruppe *Variablen* und dann auf **Neue Variable**. Nenne die Variable *made basket* (»Korb geworfen«) und wähle **Nur für diese Figur**. Ändere dann den Code für den Ball wie im Folgenden gezeigt.

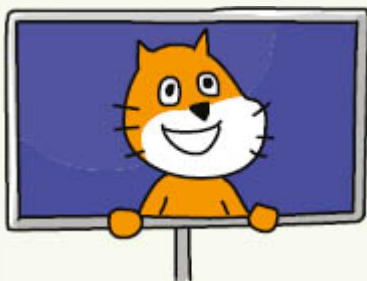
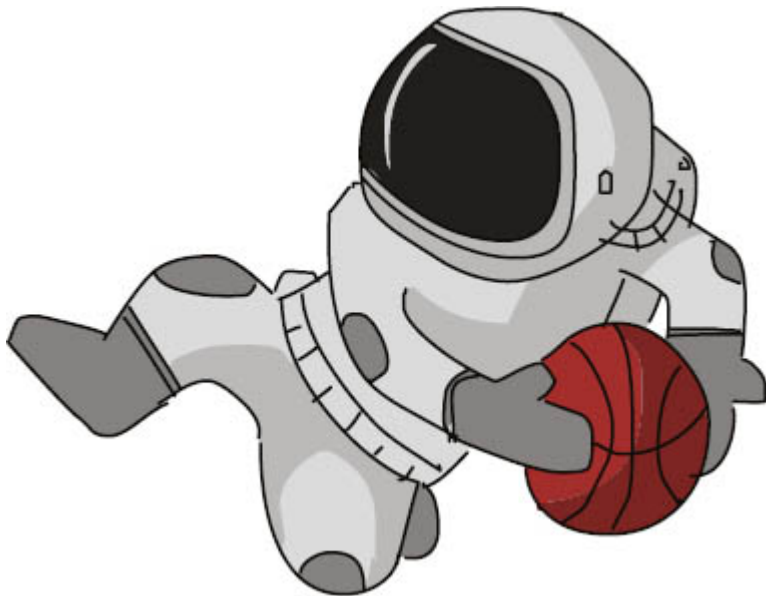


Wenn der Spieler die Leertaste drückt, wird die Variable *made basket* auf *no* gesetzt. Das ist sinnvoll, denn schließlich kann noch gar kein Korb geworfen worden sein, wenn der Spieler gerade erst wirft. Später verwenden wir einen **und**-Block, um eine weitere Bindung hinzuzufügen, die der Code überprüfen muss, um festzustellen, ob ein Korb erzielt wurde. Jetzt wird ein Korb nur dann erkannt und der Code in dem **falls ... dann**-Block nur dann ausgeführt, wenn die folgenden drei Bedingungen alle wahr sind:

1. Die Figur *Basketball* berührt die Figur *Hitbox*.
2. Die Variable *y velocity* ist negativ (der Basketball bewegt sich also abwärts).
3. Die Variable *made basket* hat den Wert *no*.

Wenn der Code zum ersten Mal feststellt, dass ein Korb erzielt wurde, erhöht er *Player 1 Score* um 1 und setzt *made basket* auf *yes*. Bei allen weiteren Prüfungen

während dieses Wurfs ist *made basket* nicht mehr gleich *no*, weshalb kein weiterer Korb mehr erkannt wird. Wenn der Spieler wieder die Leertaste drückt, um den Ball erneut zu werfen, wird *made basket* auf *no* zurückgesetzt.



### Zeit zu speichern!

Klicke auf die grüne Fahne, um den bisherigen Code zu testen. Wirf einige Bälle und vergewissere dich, dass *Player 1 Score* für jeden Korb immer nur um 1 erhöht wird. Klicke auf das rote Stoppschild und speichere dein Programm.

## Das vollständige Programm

Die folgende Abbildung zeigt den fertigen Code. Wenn dein Programm nicht richtig funktioniert, vergleiche deinen Code mit dem in der Abbildung.



Cat

```
Wenn  angeklickt wird
wiederhole fortlaufend
  falls Taste Pfeil nach links gedrückt? , dann
    wechsele zum nächsten Kostüm
    ändere x um -10
  falls Taste Pfeil nach rechts gedrückt? , dann
    wechsele zum nächsten Kostüm
    ändere x um 10
```

```
Wenn  angeklickt wird
setze y velocity auf 0
wiederhole fortlaufend
  ändere y um y velocity
  falls y-Position > -130 , dann
    ändere y velocity um -2
  falls y-Position < -130 , dann
    setze y auf -130
    setze y velocity auf 0
```



Hitbox

```
Wenn Taste Pfeil nach oben gedrückt wird
falls y-Position = -130 , dann
  setze y velocity auf 20
```

```
Wenn  angeklickt wird
setze Effekt Durchsichtigkeit auf 100
wiederhole fortlaufend
  gehe zu Hoop
```

# Index

## Symbole

+ Block. *Siehe* Plus-Block (+)

< Block. *Siehe* kleiner-als-Block (<)

= Block. *Siehe* ist-gleich-Block (=)

> Block. *Siehe* größer-als-Block (>)

- Block. *Siehe* Minus-Block (-)

\* Block. *Siehe* Multiplikations-Block (\*)

/ Block. *Siehe* Divisions-Block

## A

**abrunden von**-Block 154

**Additions**-Block (+). *Siehe* Plus-Block (+)

AI 162

Als lokale Datei speichern 7

**ändere ...-Effekt um**-Block (Aussehen)

Durchsichtigkeit 94

Farbe 93

**ändere Größe um**-Block (Aussehen) 95

**ändere Stiftfarbe um**-Block 25

**ändere ... um**-Block (Variablen) 56

**ändere x/y um**-Block (Bewegung) 33

Animation

Änderung der Durchsichtigkeit 94

Drehen 100

Ein- und Ausblenden 94

Explosion 121

Gehanimation 149

Gehen 60

Helligkeit 101

Kostümwechsel 60, 122

## Animationen

Einzelbilder 151

Fallen 150


Springen 150

Artificial Intelligence 162

Asteroid Breaker 107

Asteroids (Spiel) 105

Atari 105

Aussehen-Blöcke 

ändere ...-Effekt um

Durchsichtigkeit 94

Farbe 93

ändere Größe um 95

komme nach vorn 38

nächstes Kostüm 59

sage 22, 56

setze Größe auf 38

setze Helligkeit-Effekt auf 101

verstecke dich 64

wechsle zu Kostüm 37

Auswahlwerkzeug 157

## B

Basketball 53

Berührungen

An anderer Figur abprallen 81

Andere Figuren 37

Boden 57

Bühnenrand 80

Gelände 133

Hitbox 63, 146

Mehrfach 68

Überstehende Körperteile 146

Umlaufende Bewegung an den Bühnenrändern 110

Bewegung-Blöcke 

ändere x/y um 33

drehe dich gegen den Uhrzeigersinn um ... Grad 9  
drehe dich im Uhrzeigersinn um ... Grad 66, 95, 99  
drehe dich zu 22, 78  
gehe ...er-Schritt 20  
gehe zu 25  
gehe zu x y 47  
pralle vom Rand ab 20, 80  
Richtung 81  
setze Drehtyp auf 79  
setze Richtung auf 21, 82  
x-Position 111, 114, 116  
y-Position 111, 114, 116, 135

## Bibliothek

Bühnenbilder 16

Blöcke (Siehe auch die Bezeichnungen der einzelnen Blöcke und Blocktypen (fett hervorgehoben))

Blockpalette 5

Einzelne Blöcke ausführen 9

Hinzufügen 7

Löschen 8, 24

Stapelblöcke 7

Variablenblock 54

Verschieben 136

Weißer Felder 8

Werteblocke 8

Blockpalette 5, 8

Boolesche Werte 144

Brick-Breaker-Spiel 75

Bugs 68

## Bühne

Am Bühnenrand abprallen 20, 80

Bühnenbild 16

Einführung 4

Koordinaten 32

Ränder 32

Umlaufende Bewegung an den Bühnenrändern 110

Variablen anzeigen 58

Bühnenbilder

Bibliothek 16

Grafischer Hintergrund 36

## C

Cheat-Modus 48,72

Creative-Commons-Lizenz 11

## D

Daumenkino 122

Deckenerkennung 143

Demoscene 13

**drehe dich gegen den Uhrzeigersinn um ... Grad-Block** 9

**drehe dich im Uhrzeigersinn um ... Grad-Block** 66,95,99

**drehe dich zu-Block** 22,78

Drehpunkt 6

Drehtyp 79

Duplizieren

Code 35

Figuren 23

Unterschied zu Klonen 85

Durchsichtigkeit 94

Figuren unsichtbar verschieben 161

## E

Editor

Aufbau 4

Blockpalette 5

Grüne Fahne 5

Offline-Editor 4

Projektname 4


Rotes Stoppschild 5

Skriptbereich 4

Starten 4

Tipps-Fenster 10


Version 1.4.4

Eigene Blöcke   
 Bearbeiten 144  
 Boolesches Feld 144  
 Eingaben 138  
 Einstellungen 144  
 Wiederholung von Code vermeiden 153

Eingaben 138

Energiespirale 126

Entfernung von 93

Ereignisse-Blöcke 

sende ... an alle 40

Wenn grüne Fahne angeklickt 9

Wenn ich ... empfangen 41

**erzeuge Klon von mir selbst**-Block 83

Explosion 121

## F

**falls ... dann**-Block 34

**falls ... dann ... sonst**-Block 97

Farbwähler 6

Farbwechsel 25

Figuren

Am Bühnenrand abprallen 20, 80

An anderer Figur abprallen 81

Animation durch Kostümwechsel 60

Anstoßen 109

Auf dem Boden landen 57

Berührungen 37

Beschleunigen 56

Bewegungen an steilen Wänden blockieren 138

Bewegungen zufällig auswählen 162


Blinken 93

Code hinzufügen 20

Drehpunkt 6

Duplizieren 23

Einführung 4

Ein- und Ausblenden 94  
Figur aus einer Datei laden 36  
Figurenliste 5  
Gehanimation 149  
Gelände 132  
Größe ändern 38  
Hitbox 62  
Katze löschen 16  
Klettern 139  
Klone 83  
Kostüme 36  
Löschen 16  
Mausbewegungen folgen 78  
Mit Tasten steuern 33  
Nach rechts und links gehen 135  
Neue Figur zeichnen 6  
Programmieren 4  
Richtung Mauszeiger bewegen 114  
Schwerkraft 53  
Skripte 5  
Springen 58  
Sprunghöhe 141  
Spur hinter sich herziehen 98  
stoppe andere Skripte der Figur 100  
Teilen 117  
Umlaufende Bewegung an den Bühnenrändern 110  
Unsichtbar verschieben 161  
Verstecken, bis eine Bedingung eintritt 89  
Flappy Bird (Spiel) 168  
Forum 11  
Fühlen-Blöcke   
Entfernung von 93  
Taste ... gedrückt? 34  
wird ... berührt? 37, 82  
Füllwerkzeug 43

Für alle Figuren 55

## G

Gehanimation 149

**gehe ...er-Schritt**-Block 20

**gehe zu**-Block 25

**gehe zu x y**-Block 47

Gelände

Berühren 133

Berührungen 146

Berührungen mit überstehenden Körperteilen 146

Bühnenbild 156

Decken 143

Figur 132

Hintergrund entfernen 156

Hitbox 156

Klettern 139

Landen 133

Steile Wände 138

Geschosse

Abfeuern 113

Begrenzter Munitionsvorrat 124

Energiespirale 126

Löschen 119

Reichweite 114

Zielen 114

Gradangaben 21

griffpatch 149

**größer-als**-Block (>) 57, 125, 145

Grüne Fahne 5, 9

## H

Helligkeit 101

Hilfe 10

Hintergrund 92

Hitbox 62, 146

Gelände 156

Kostüm 147

## I

**ist-gleich**-Block (=) 69, 140

iterative Entwicklung 15, 43

## J

JavaScript 169

Jonasson, Martin 91

Jump-&-Run-Spiele

    Gegner 162

    Gelände 132

    Levels 156


    Seitenansicht 51

    Spielprinzip 129

    Sprunghöhe 141

## K

KI 162

Klang-Blöcke 

    spiele Klang 61, 97, 99

    spiele Klang ganz 93

Klänge

    Musik 93

    spiele Klang 61

**kleiner-als**-Block (<) 57, 67, 87, 93, 125, 142

Klone

    Figuren teilen 117

    Klonen von Klonen verhindern 113

    Klone von Klonen 118

    Löschen 98

    Richtung Mauszeiger bewegen 114

    Sichtbar machen 84

    Spur aus Klonen hinter Figuren herziehen 98

    Unterschied zu Duplikaten 85

    Viele Exemplare einer Figur 83

Wiederholt erstellen 85

Kollisionserkennung. *Siehe* Berührungen

**komme nach vorn**-Block 38

Koordinaten 32

Kostüme

Animation 60

Einzelbilder für Animationen 151

Explosionsanimation 122

Gehanimation 149

Hitbox 147

Kostüm aus einer Datei laden 150

Levels 36

Registerkarte 36

Speichern 7

Künstliche Intelligenz 162

## L

Levels

Gelände 156

Kostüme 36

Lifelong Kindergarten 2

Linienbreite 6

**lösche diesen Klon**-Block 98

Löschen rückgängig 9

## M

Malstift-Blöcke 

ändere Stiftfarbe um 25

schalte Stift ein 25

wische Malspuren weg 25

Maus

Mausbewegungen folgen 78

Zielen 112

Minecraft (Spiel) 169

**Minus**-Block (-) 82

MIT Media Lab 2

**mod**-Block 154

Modulo 154

**Multiplikations**-Block (\*) 140

Musik 93

## N

Nachrichten

    Neue Nachricht 40

    sende ... an alle 40

    Wenn ich ... empfangen 41

**nächstes Kostüm**-Block 59

Neue Figur zeichnen 6

Neues Bühnenbild 16

Neue Variable 53


Neue Variable (Variablen-Blöcke) 53

**nicht**-Block 81, 133, 137

## O

**oder**-Block 139, 164

Offline-Editor 4

Operatoren-Blöcke 

    abrunden von 154

    mod 154

    nicht 81, 133, 137

    oder 139

    und 48, 68

    Zufallszahl von ... bis ...-Block 20, 21, 62, 72

## P

Pac-Man (Spiel) 168

Pfeiltasten 33

Pinselwerkzeug 18

Platformer 129

**Plus**-Block (+) 153, 154

**pralle vom Rand ab**-Block 20, 80

Professionelles Aussehen 76

Programme

Anhalten 5  
Ausführen 9  
Demoscene 13  
Lizenz 11  
Professionell aussehende Spiele 76  
Remix 11  
Schau hinein 11  
Skizzieren 14  
Speichern 4  
Starten 5  
Veröffentlichen 10  
Projektname 4  
Punkttestand  
    Auf 0 zurücksetzen 84  
    Variablen 65  
Purho, Purhoetri 91  
Python 169

## R

Radiergummi 157  
Rechteckwerkzeug 63  
Regenbogenlinien 23  
Remix 11  
Rest einer Division 154  
**Richtung**-Block 81  
Richtungen  
    Drehtyp 79  
    Figuren in eine bestimmte Richtung stoßen 109  
    Gradangaben 21  
    Koordinaten ändern 33  
    setze Richtung auf 82  
Rotes Stoppschild 5  
Rückgängig  
    Löschen rückgängig 9  
    Zeichenbereich 6

## S

**sage**-Block 22, 56

**schalte Stift ein**-Block 25

Schau hinein 11

Schleifen

wiederhole fortlaufend 9

Schwerkraft

Auf dem Boden landen 57

Beschleunigung 56

Gelände 133

Springen 58

Sprunghöhe 141

Variablen 53

Scratch

Editor. *Siehe* Editor

Einführung 1

Forum 11, 169

Hilfe 10

Konto 3

Programme veröffentlichen 10

Quellen 169

Schau hinein 11

Version 1.4 4

Website 2

Scratcher 2

**sende ... an alle**-Block 40

**setze ... auf**-Block 55, 84, 114, 116, 133

**setze Drehtyp auf**-Block 79, 133

**setze Durchsichtigkeit-Effekt auf**-Block 161

**setze Größe auf ... %**-Block 38, 84, 116

**setze Helligkeit-Effekt auf**-Block 101

**setze Richtung auf**-Block 21, 82

Skelettdateien 30

Skriptbereich 4

## Skripte

- Blöcke hinzufügen 7
- Einführung 5
- Einzelne Skripte ausführen 9
- Registerkarte 20
- Skriptbereich 5
- stoppe alles 100
- Stoppen 100

Sonic the Hedgehog (Spiel) 51


Speichern 4

**spiele Klang**-Block 61

**spiele Klang ganz**-Block 93

sprites. *Siehe* Figuren

Stapelblöcke 7

Steuerung-Blöcke 

- erzeuge Klon von mir selbst 83
- falls ... dann 34
- falls ... dann ... sonst 97
- lösche diesen Klon 98
- warte bis 89
- Wenn ich als Klon entstehe 83
- wiederhole bis 66
- wiederhole fortlaufend 9, 34, 62

**stoppe alles**-Block 88, 100

stoppe andere Skripte der Figur 100

**Subtraktions**-Block (-). *Siehe* Minus-Block (-)

Super Mario Bros. (Spiel) 51, 127, 129

Super Meat Boy (Spiel) 127

## T

**Taste ... gedrückt?**-Block 34

Tasten

- Pfeiltasten 33
- Sprunghöhe 141
- WASD 45

Testen

Schrittweise xxii

Vorübergehende Codeänderungen 90

Textwerkzeug 87

Timer 166

Tipps-Fenster 10

Turbo-Modus 27

## U

Umlaufende Bewegung an den Bühnenrändern 110

**und**-Block 48, 68

Ursprung 32

## V

Variablen

Einführung 53

Für alle Figuren 55

Löschen 55

Neue Variable 53

Nur für diese Figur 54

Punkttestand 65

Unsichtbar 58

Variablenblock 54

Werte speichern 55

Variablen-Blöcke 54

ändere ... um 56

setze ... auf 55, 84, 114, 116, 133

Veröffentlichen 10

**verstecke dich**-Block 64, 84


## W

Wahr/falsch-Wert 144

**warte bis**-Block 89

WASD-Tasten 45

**wechsle zu Kostüm**-Block 37

Weitere Blöcke. *Siehe auch* Eigene Blöcke 

**Wenn grüne Fahne angeklickt**-Block 9

**Wenn ich als Klon entstehe**-Block 83

**Wenn ich ... empfangen**-Block 41

Werteblocke 8

Wiederherstellen 6

**wiederhole bis**-Block 66

**wiederhole fortlaufend**-Block 9, 34, 62

**wird ... berührt?**-Block 37, 82

**wische Malspuren weg**-Block 25

## X

**x-Position**-Block 111, 114, 116

## Y

**y-Position**-Block 57–58, 111, 114, 116, 135

## Z

Zeichenbereich

Aufbau 6

Auswahlwerkzeug 157

Drehpunkt 6

Fadenkreuz 6

Farbwähler 6

Füllwerkzeug 43

Hintergrund entfernen 157

Linienbreite 6

Pinselwerkzeug 18

Radiergummi 157

Rechteckwerkzeug 63

Rückgängig 6

Textwerkzeug 87

Wiederherstellen 6

Zeichenwerkzeuge 6

Zoom-Schaltflächen 6, 18

ZIP-Datei xxiii

Zoom-Schaltflächen 6, 18

Zufallsbewegungen 111

**Zufallszahl von ... bis ...**-Block 20, 21, 62, 72

Zusatzmaterial xxiii

Zwei-Spieler-Modus 43