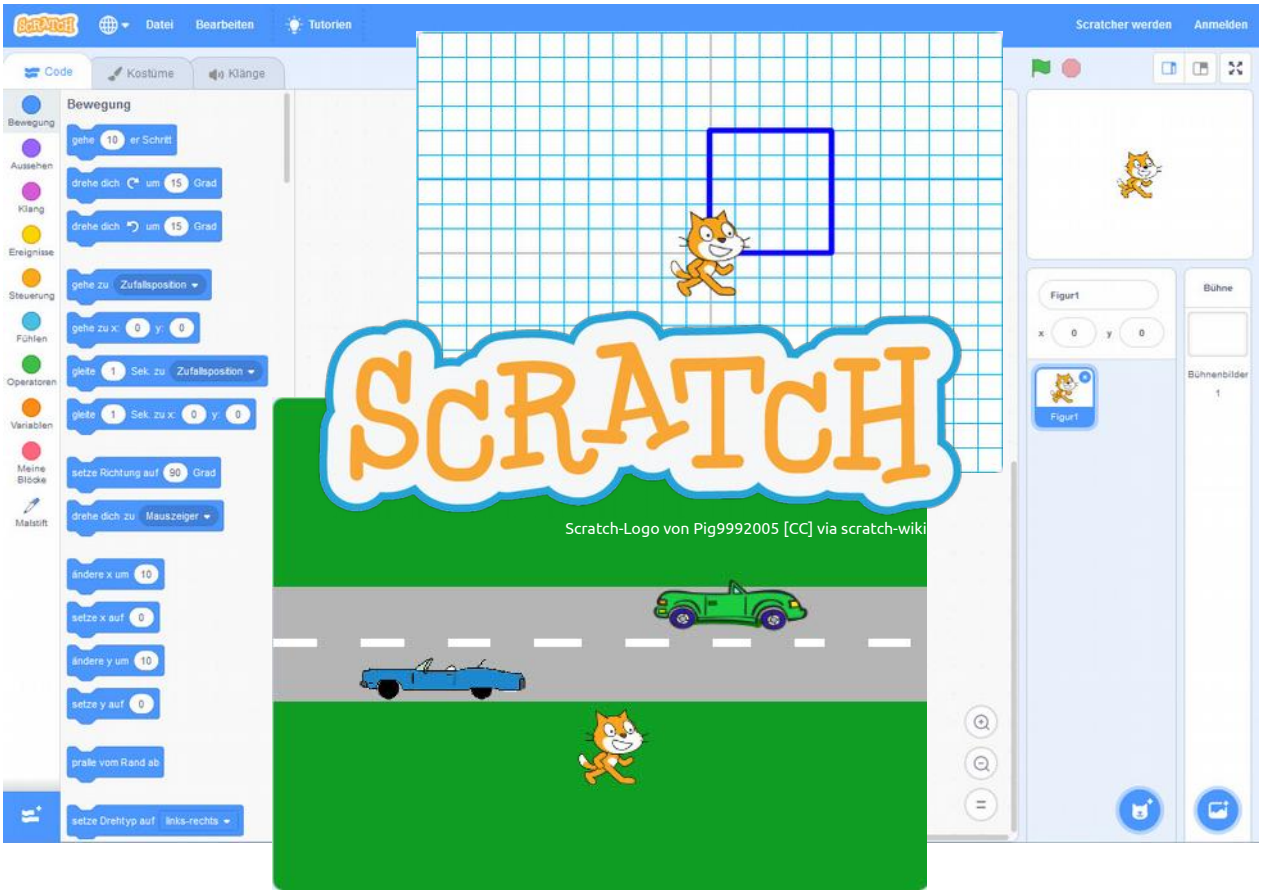
EINFÜHRUNG IN DIE PROGRAMMIERUNG MIT SCRATCH



Scratch ist eine Programmiersprache und Online-Gemeinschaft, in der man seine eigenen interaktiven Geschichten, Spiele und Animationen erstellen und seine Werke mit anderen überall auf der Welt teilen kann. Beim Entwickeln und Programmieren von Scratch-Projekten lernen junge Menschen kreativ zu denken, systematisch vorzugehen und kooperativ mit anderen zusammenzuarbeiten.

Scratch ist ein Projekt der Lifelong-Kindergarten-Gruppe am MIT-Media-Lab. Es ist kostenlos unter *http://scratch.mit.edu* verfügbar.

Monika Eisenmann – E-Mail: eisenmann.schule@email.de. – Januar 2017
Michael Sedding – E-Mail: sedding@csg-tuebingen.de - Juni 2018

Vorbemerkung für Lehrer: Scratch-Versionen und die ihre Zukunft

Die derzeit (06/2018) noch aktuelle Scratch-Version 2.0 basiert auf Adobe Flash. Aufgrund gravierender Sicherheitslücken ist Adobe Flash in den letzten Jahren mehr und mehr in Verruf geraten. Seit etwa 2015 rät die Fachpresse[1] den Flash-Player nicht mehr zu verwenden, Browserhersteller gingen dazu über, das Flash-Plugin standardmäßig zu deaktivieren. Schließlich hat Adobe letztes Jahr angekündigt, die Entwicklung von Flash 2020 einzustellen, keine Updates mehr bereitzustellen und die Software nicht mehr zu verbreiten[2].

Scratch 3.0

Auch die Entwickler von Scratch stellen sich auf das Ende von Flash ein. Derzeit wird an der Version 3.0 gearbeitet, die auf HTML5 basieren wird. Alle Blöcke aus älteren Scratch-Versionen werden verfügbar sein, bestehende Projekte sollen sich importieren lassen. Leider ist bisher nur eine (ausschließlich englische) Beta-Version (*https://preview.scratch.mit.edu/*) zugänglich, bei der u.a. die Community-Funktionen noch nicht implementiert sind. Die Veröffentlichung der finalen Version von Scratch 3.0 soll im August 2018 erfolgen.[3] Eine offline-Version ist für später angekündigt und auch die Website *https://scratch.mit.edu* soll noch dieses Jahr auf die Version 3 migriert werden.

Es ist also davon auszugehen, dass man ab SJ 2018/2019 den Informatikunterricht in Klasse 7 mit einem Flash-losen und sogar auf mobilen Endgeräten nutzbaren Scratch durchführen kann.

Die Alternative: Snap!

Dennoch lohnt ein Blick auf die Alternative. An der Uni Berkeley wird mit Snap! eine weitere Sprache / Entwicklungsumgebung für visuelles Programmieren entwickelt. Die Benutzeroberlfäche von Snap! ähnelt Scratch weitestgehend, was einfach daher rührt, dass Snap! anfangs nur ein modifiziertes Scratch 1.x war. Auch wenn für die aktuelle Version der Code komplett neu geschrieben wurde, merkt man doch die gemeinsamen Wurzeln.

Insgesamt ist Snap! mächtiger als Scratch. So kann man mit Snap! beispielsweise eigene Prozeduren mit Parametern und Rückgabewerten definieren und auf diese Weise rekursive Algorithmen implementieren.

Auch Snap! läuft im Browser (*https://snap.berkeley.edu/run/*), eine offline-Version gibt es nicht.

Abwägung

Die Entscheidung, ob man in Zukunft auf Snap! oder Scratch setzen will, ist eine Abwägung: Einerseits funktioniert unter Snap! derzeit schon einiges, was unter Scratch noch nicht implementiert ist. Zudem läuft es schon jetzt stabil und ohne Flash. Durch den erweiterten Funktionsumfang kann man mit Snap! ggf. auch noch nach Klasse 7 weiterarbeiten. Andererseits hat Scratch im Moment die größere Community und wird, wenn der Umstieg gut gelingt, vermutlich auch das häufiger verwendete Programm bleiben. Auch die Ankündigung eines offline-Editors und die Verwendbarkeit auf Tablets spricht aus technischer Sicht für Scratch.

1 z.B. *https://www.heise.de/ct/ausgabe/2015-5-Editorial-Goodbye-Flash-2536897.html* oder *https://www.heise.de/security/meldung/Flash-Player-deaktivieren-Schon-wieder-Angriffe-auf-ungepatchte-Luecke-2535100.html*

2 *https://www.heise.de/newsticker/meldung/Adobe-verabschiedet-sich-von-Flash-2020-ist-Schluss-3783264.html*

3 https://medium.com/scratchteam-blog/3-things-to-know-about-scratch-3-0-18ee2f564278

Elternbrief

Liebe Eltern,

im Informatikunterricht arbeiten wir in den nächsten Wochen mit Scratch. Scratch ist eine Programmierumgebung, mit der die Schülerinnen und Schüler, ohne sich an komplizierte Syntax halten zu müssen, die Grundstrukturen der Programmierung kennen lernen und einüben.

Es wäre schön, wenn Sie Ihrem Kind zu Hause die Möglichkeit geben könnten regelmäßig mit Scratch zu arbeiten und zu experimentieren.

Sie müssen dafür nichts installieren. Scratch 3.0 läust im Browser und kann mit dem Computer aber auch mit einem Tablets benutzt werden.

Sie finden den Entwicklungsumgebung unter: *https://scratch.mit.edu/*

(Derzeit noch: *https://preview.scratch.mit.edu/*)

Vielen Dank für Ihre Unterstützung!

Mit freundlichen Grüßen

Laufzettel von _____

Ich kann …

… ein Scratchprojekt abspeichern.	A 1, A 3	
… den Namen einer Figur ändern.	A 2	
… die Begriffe Anweisung und Sequenz erklären.	Teil B: 1. Seite	
… beschreiben, was die einzelnen Bewegungs-anweisungen bewirken	B 1	
… die Katze mit dem Stift zeichnen lassen.	Teil B: 2. Seite	
… die Katze ein Rechteck, ein Quadrat und ein gleichseitiges Dreieck zeichnen lassen.	B 2 – 5	
… Tastaturereignisse programmieren, so dass die Katze über die Bühne gesteuert werden kann.	C 2	
… für eine Figur ein neues Kostüm erstellen.	Teil C	
… ein Bühnenbild erstellen und seinen Namen ändern.	D 1 – 3	
… ein neues Objekt anlegen.	Teil E	
… erklären, was eine Schleife ist.	Teil E	
… das Auto mit einer Schleife fahren lassen.	E 1	
… das Auto mit einer Schleife so lange fahren lassen, bis es am Rand angekommen ist.	E 2	
… Programmblöcke mit Schleifen verstehen und zuordnen.	F 1	
… einen Programmcode erweitern, um eine Anzahl Quadrate mehrfach zeichnen zu lassen.	F 2	
… einen Programmcode mit Schleife und einer Verzweigung erklären.	F 3	
… selbständig vorgegebene Figuren entwerfen und zeichnen (Bsp.: Haus des Nikolaus).	F 4 (freiwillig)	

… erklären, was der Steuerungsblock „falls … dann … sonst“ bewirkt.	Teil E mit Hilfekarte	
… erklären, was eine Variable ist.	Teil G	
… eine neue Variable anlegen (deklarieren) und ihr einen ersten Wert zuweisen (sie initialisieren).	Teil G bis G 1	
… Tastaturereignisse so programmieren, dass der Wert einer Variable erhöht und verringert wird.	G 2	
… mit Hilfe einer Variable das Auto mit unterschiedlicher Geschwindigkeit fahren lassen.	G 3 – 6	
… Bedingungen mit „und“ verknüpfen.	G 7	

Teil 0: Vorbereitungen

Die Scratch-Entwicklungsumgebung gibt es derzeit nur in einer online-Variante. Um sie zu verwenden, musst Du kein Programm auf deinem PC installieren. Stattdessen rufst Du die Scratch-Internetseite in einem Browser auf. Dort läuft dann das Programm.

Mach dich arbeitsbereit

Bevor es losgeht, lege bitte einen Arbeitsordner in deinem Home-Verzeichnis an, in dem Du deine Programme / Zwischenstände abspeichern kannst. Starte dazu den Dateimanager und wechsle ins Verzeichnis ~/Home_auf_Server/Info7/ und erstelle dort einen neuen Ordner mit dem Namen Scratch

Öffne dann in einem Browser (z.B. Firefox) die Internetadresse: *https://preview.scratch.mit.edu/*.

Jetzt kann es losgehen!

Ein Wort zum Abspeichern und Öffnen deiner Programme

Weil die Entwicklungsumgebung von Scratch ja „im Internet“ läuft, haben die Menüpunkte zum Speichern und Öffnen etwas eigenwillige Namen.

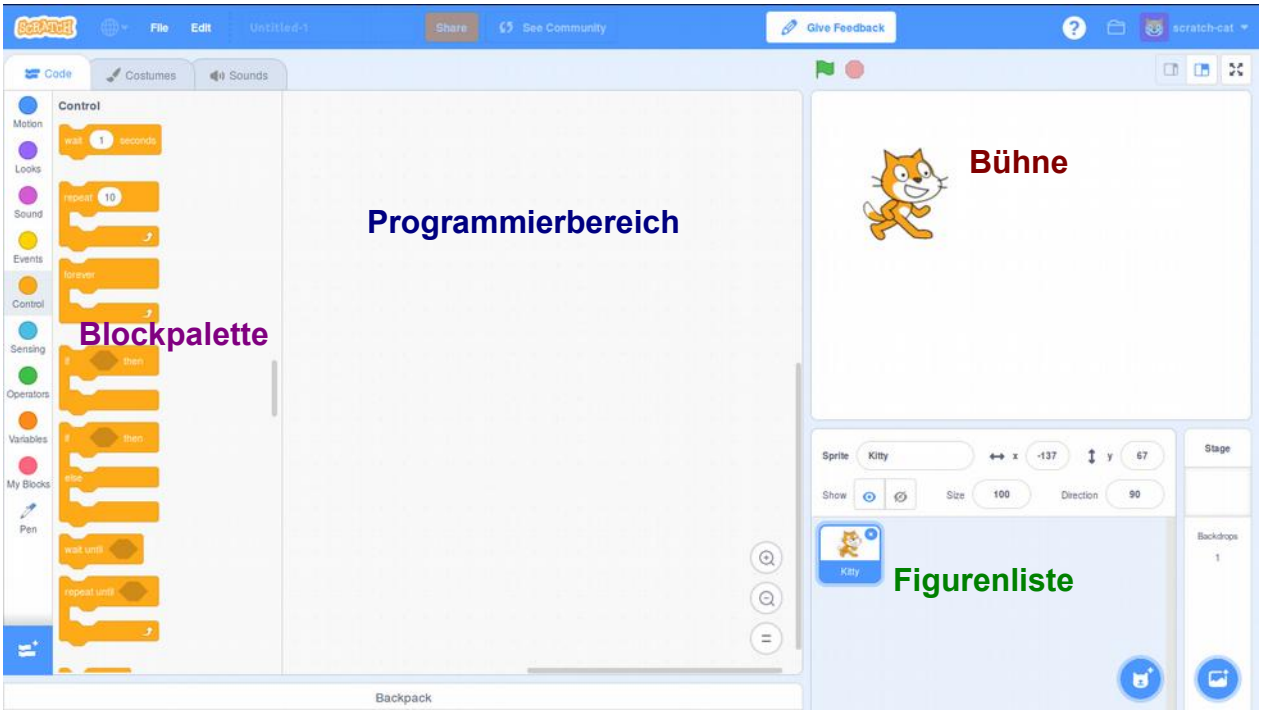
Ein Programm *speicherst* Du *ab*, indem Du im Menü Datei den Menüpunkt: „Herunterladen auf deinen Computer“ klickst.

Ein gespeichertes Programm *öffnest* Du, indem Du im Menü Datei den Menüpunkt: „Hochladen von deinem Computer“ klickst. Diesen Schritt musst du zumindest zu Beginn jeder Unterrichtsstunde einmal machen

Denke daran, Zwischenstände immer wieder unter anderem Namen abzuspeichern. So kannst Du leicht auf vorhergehende Entwicklungszustände zurückspringen.

Möchtest Du zu Hause weiterprogrammieren, lege dir die gespeicherte Datei in die CarloCloud.

Teil A: Erste Begegnung mit Scratch



Die **Scratch-Oberfläche** teilt sich in vier Bereiche auf.

Auf der **Bühne** läuft alles ab, was du programmierst. Dort machen deine Figuren das, was du ihnen im **Programmierbereich** aufträgst.

In der **Figurenliste** findest du alle Figuren, die in deinem Projekt dabei sind. Anfangs ist es nur eine Figur. Du kannst dort auch neue Figuren erzeugen und bestimmte Eigenschaften festlegen. Außerdem findest du dort auf der linken Seite auch die Bühne(n).

In der **Blockpalette** findest du die Blöcke, die du zum Programmieren brauchst. Du kannst sie entweder dort direkt durch einen Mausklick testen, oder mit der Maus in den **Programmierbereich** ziehen und an ein Programm anfügen.

Aufträge:

1. *Du kannst zuerst die Sprache ändern. Klicke dazu auf den Globus links oben neben Datei und wähle die gewünschte Sprache aus.*
2. *Speichere dein Projekt erst einmal ab. Dazu wählst du Datei – Herunterladen auf deinen Computer, wählst den Ordner aus, den du für deine Programme angelegt hast und gibst den Dateinamen* `01katze` *ein. Die Dateiendung .sb3 wird automatisch ergänzt.*
3. *Gib deiner Katze einen Namen. Das kannst Du im Einstellungsbereich oberhalb der Figurenliste tun. Schreibe statt „Figur 1“ den Namen deiner Katze, z.B. Kitty.. Hier kannst Du auch die Größe der Katze verändern.*

Teil B: Die Katze lernt laufen

Die Katze oder auch jede andere Figur, die wir auf die Bühne setzen, kann sich bewegen. Wie sie sich bewegen soll, müssen wir ihr sagen.

In der Blockpalette findest du eine ganze Menge Blöcke, die für die Bewegung der Katze zuständig sind. Du kannst sie durch einen Mausklick ausprobieren. Entweder in der Blockpalette selbst, oder wenn du sie vorher in den Programmierbereich ziehst und dort anklickst. Das hat den Vorteil, dass du auch Blöcke aneinanderhängen kannst und sie dann alle nacheinander ausgeführt werden.

Sequenz

Jeder Auftrag, den du deiner Katze gibst, heißt **Anweisung** (z.B. „gehe 10er-Schritt“). Mehrere Anweisungen nacheinander nennt man **Sequenz**.

Auftrag:

1. *Teste folgende Anweisungen oder Sequenzen und schreibe auf, was passiert.*

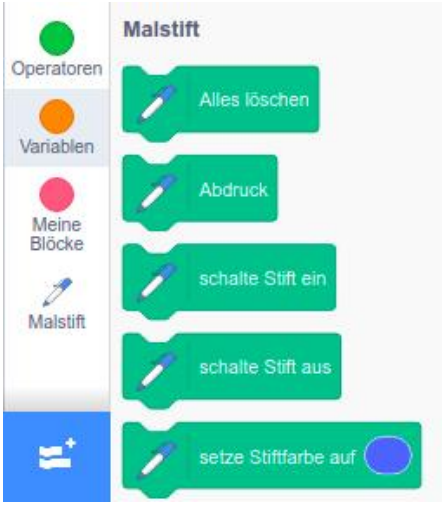
Anweisung / Sequenz	Wirkung
 Zusatzfrage: Was passiert bei negativer Zahl?	
 Zusatzfrage: Wie heißen die Koordinaten der Ecken der Bühne?	
	
	
	
	
 Zusatzauftrag: Test alle vier Richtungen.	

Probiere auch alle Blöcke in der Tabelle, nachdem du jeweils vorher die Anweisung „drehe dich um 30 Grad“ eingefügt hast. Bei welchen ändert sich das Ergebnis?

Bevor du weitermachst, speichere deine Datei unter `02katze_bewegen` ab. („Datei“ – „Herunterladen auf deinen Computer“)

Du kannst auch Blöcke wieder löschen, die du nicht mehr brauchst oder falsch eingefügt hattest. Dazu ziehst du den Block einfach wieder zurück in die Blockpalette, oder du wählst nach einem Rechtsklick auf den entsprechenden Block „Löschen“ aus.

Um besser sehen zu können, was die Katze gemacht hat, können wir ihre Spuren zeichnen lassen. Die nötigen Werkzeuge müssen wir erst aktivieren. Klicke dazu auf das „Erweiterung hinzufügen“-Icon links unten (vgl. Abbildung) und wähle die Erweiterung „Malstift“. Es erscheinen nun weitere Programmierbausteine in einer eigenen Kategorie.



Füge nun jeweils zuerst den Block in deinem Programm ein, der die Anweisung gibt, alle Malspuren wegwischen zu lassen, bevor du etwas Neues probierst, den Block zum Einschalten (Absenken) des Stiftes setzt du vor deine Anweisungen für die Bewegungen, den Block zum Ausschalten (Anheben) danach.

Außerdem machen wir unsere Katze etwas kleiner, dass sie nicht alle Spuren verdeckt. Wie das geht, weißt Du schon von Seite 7.

Weitere Aufträge:

Schreibe jeweils ein kleines Programm, welches...

2. *… zuerst die Katze ins linke obere Eck der Bühne setzt und sie von dort aus dann ein Rechteck um die Bühne zeichnen lässt.*
3. *… zuerst die Katze ins linke untere Eck der Bühne setzt und sie dann eine Treppe mit unterschiedlich großen Stufen nach oben steigen lässt.*
4. *... die Katze ein Quadrat zeichnen lässt.*
5. *… die Katze ein gleichseitiges Dreieck zeichnen lässt.*

Wir können das, was wir im Programmierbereich zusammengesetzt haben, auch mit der grünen Fahne links oberhalb der Bühne starten und mit dem Stoppschild wieder beenden.

Dazu brauchen wir einen Block aus dem Bereich „Ereignisse“. Du siehst sicher schon, welchen Block du brauchst.

6. *Setze den Ereignisblock „Wenn Fahne angeklickt“ an den Anfang deiner Sequenz und teste mit Klick auf die Fahne.*
   (Du kannst hier eine Sequenz einer der Aufgaben nutzen oder einfach alle nacheinander anhängen.)
7. *Speichere dein Projekt noch einmal ab.*

Teil C: Ich bewege die Katze

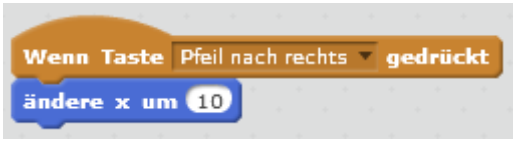
Öffne dein Projekt `01katze` oder starte ein neues Projekt und speichere es unter `03katze` wieder ab.

Verkleinere die Katze wieder – etwa auf Größe 50.

Unser Ziel ist es jetzt, die Katze mit den Pfeiltasten der Tastatur steuern zu können. Du hast schon das erste Ereignis kennen gelernt: „Wenn Fahne angeklickt“

Direkt darunter findest du das Ereignis „Wenn Taste … gedrückt“. Ziehe den Block in den Programmierbereich und wähle die Pfeiltaste nach rechts aus.

Jetzt fehlt nur die Bewegung nach rechts. Wie weit die Katze bei jedem Tastendruck gehen soll, kannst du selbst bestimmen.



Aufträge:

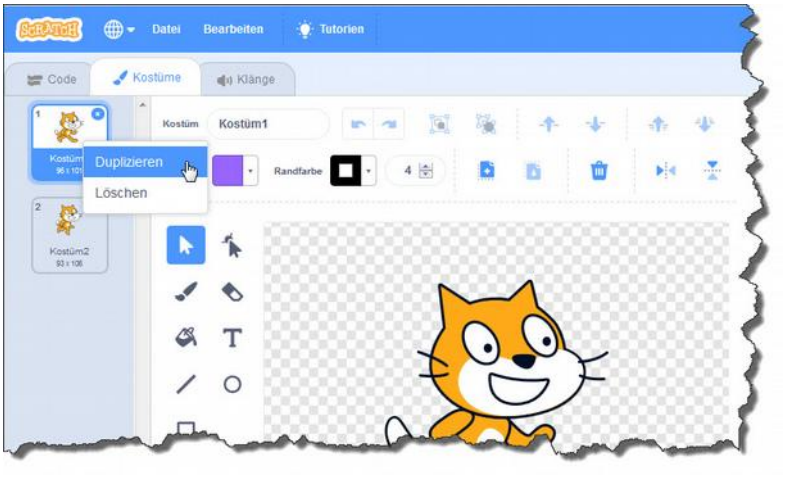
1. *Programmiere das Ereignis „Wenn Fahne angeklickt“ so, dass die Katze dabei an eine Startposition ganz unten auf der Bühne in der Mitte gesetzt wird.*
2. *Programmiere jetzt die Tastaturereignisse für die restlichen Pfeiltasten und probiere alles aus, indem du die Katze ein bisschen über die Bühne spazieren lässt.*

Ein kleiner Tipp:

Du kannst einzelne Blöcke oder auch ganze Blocksequenzen vervielfältigen und dann verändern. Klicke dazu mit der rechten Maustaste auf den ersten zu kopierenden Block und wähle „Duplizieren“ aus.

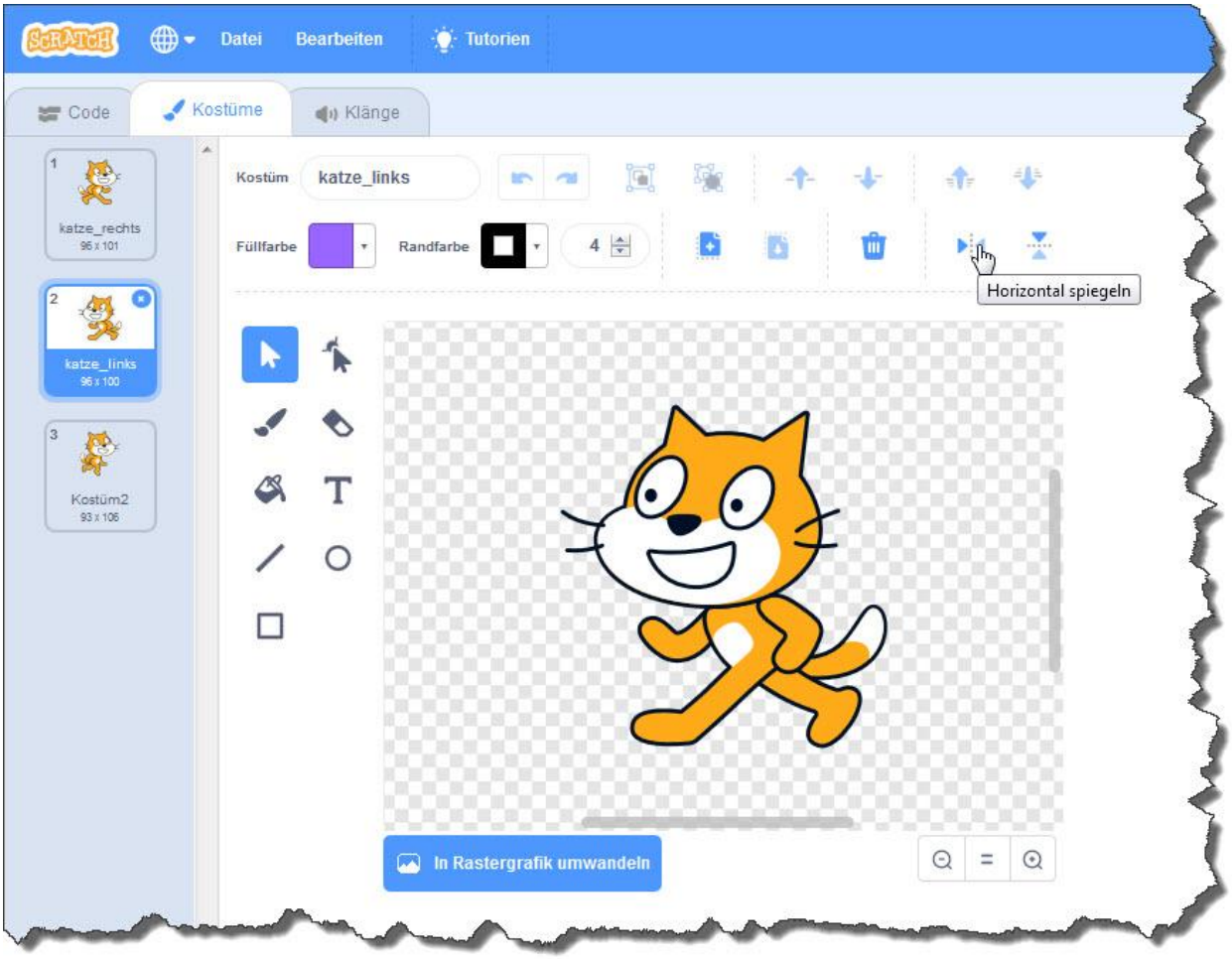
Genauso kannst du auch einzelne Teile löschen, indem du nach dem Klick mit der rechten Maustaste „Löschen“ wählst.

Die Katze läuft jetzt zwar fröhlich über die Bühne, schaut aber immer nach rechts, selbst wenn sie nach links läuft. Um das zu ändern, nutzen wir für unsere Katze ein neues Bild, auf dem sie nach links schaut. Dafür bekommt die Katze ein neues Kostüm.



Öffne das Kostümfenster. Du siehst dort, dass die Katze schon zwei Kostüme hat. Vielleicht brauchen wir das zweite noch irgendwann, wir lassen es also einfach da.

Um ein weiteres Kostüm zu erstellen, haben wir mehrere Möglichkeiten. Im Moment möchten wir aber unsere Katze nur duplizieren und dann spiegeln. Klicke mit der rechten Maustaste auf Kostüm 1 und wähle „Duplizieren“ aus. Klicke auf den Icon für „Links und Rechts vertauschen“ und gib dann dem Kostüm einen sinnvollen Namen.



Verändere auch den Namen für das normale Kostüm in katze_rechts.

Wechsle wieder in den Programmierbereich (Skripte).

Die Blöcke für den Kostümwechsel findest du in der Blockpalette unter „Aussehen“.

Aufträge:

3. *Verändere die Tastaturereignisse so, dass der Katze jeweils vor dem Loslaufen das richtige Kostüm angezogen wird.*
4. *Teste dein Programm und lasse die Katze in alle Richtungen über die Bühne laufen.*
5. *Speichere alles ab.*

Teil D: Die Katze läuft über die Straße

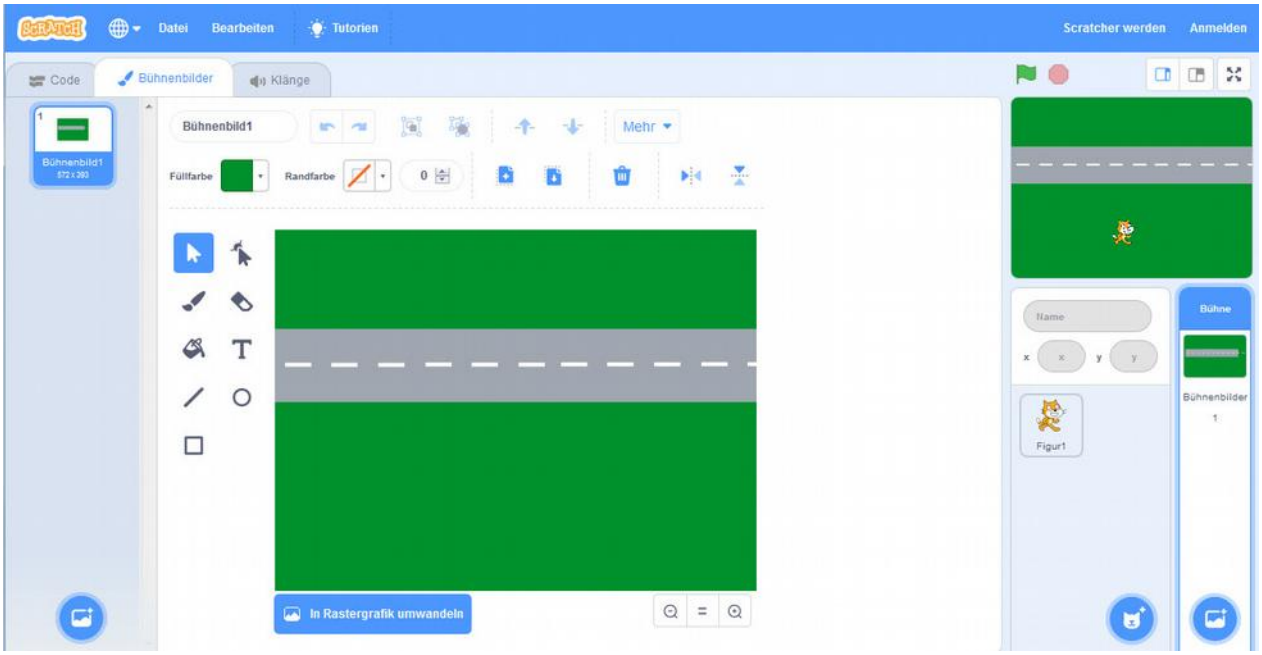
Öffne dein Projekt `03katze` und speichere es unter `04katze` wieder ab.

Nur über eine weiße Fläche zu laufen, ist für die Katze mit der Zeit langweilig. Deshalb verändern wir den Hintergrund – die so genannte Bühne.

Aufträge:

1. *Erkunde zunächst den Zeicheneditor.*
2. *Zeichne ein großes grünes Rechteck auf das Bühnenbild und zeichne darauf eine Straße. Mit dem Füllwerkzeug kannst Du nachträglich die Farbe anpassen.*
3. *Gib dem Bühnenbild einen neuen Namen: Level1.*

Dein Bühnenbild könnte etwa so aussehen:



Als nächstes brauchen wir eine neue Figur (solche Figuren nennt ein Programmierer auch Objekte): das erste Auto, das dann auf der Straße fahren soll.

Findest du heraus, wie man ein neues Objekt anlegen kann? Du kannst dir selbst ein Auto zeichnen oder ein Bild aus einer Bildbibliothek wählen.

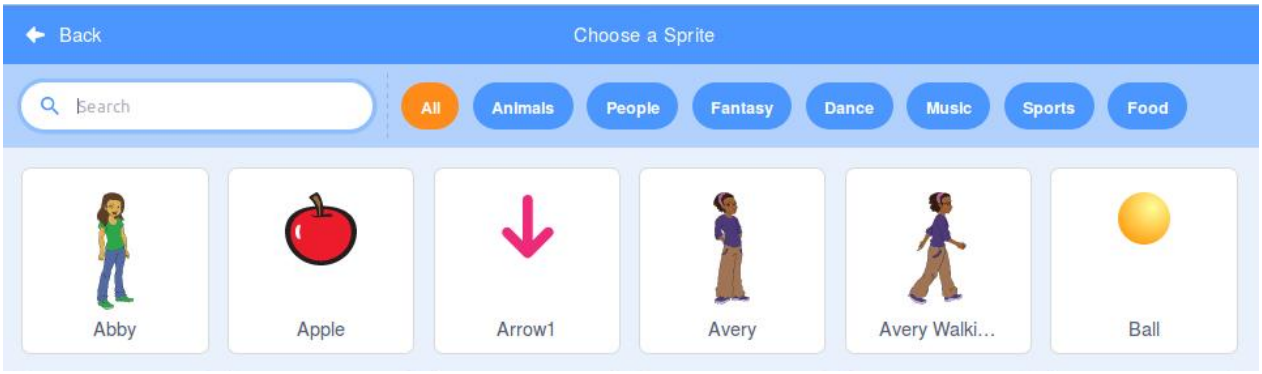
Teil E: Ein Auto fährt vorbei

Hast du schon ein neues Objekt erzeugt? Wenn nicht, bekommst du hier die Anleitung Klicke mal auf den blauen Katzenkopf mit dem Sternchen am rechten Ohr, den Du unten rechts in der Figurenleiste siehst:

Du hast dann – von oben nach unten – vier Möglichkeiten:

1. Eine Figur aus einer Datei laden.
2. Eine zufällig gewählte Figur einfügen.
3. Eine neue Figur selbst zeichnen.
4. Eine Figur aus der Scratch-Bibliothek wählen.

Wähle zunächst (mit der 4. Möglichkeit) ein Auto aus der Bibliothek. Wenn du später magst, kannst du eigene Autos entwerfen.



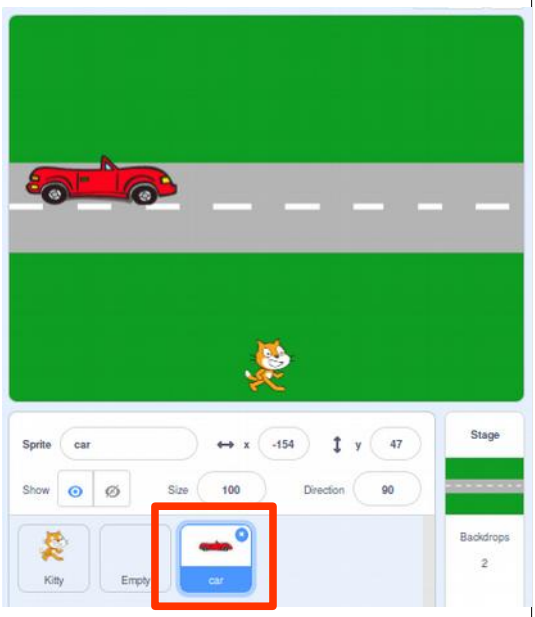
Wähle ein Auto aus und drücke OK. Du kannst die Autofarbe im Kostümfenster und den Namen des Objektes in der Figurenliste ändern. Setze das Auto auf die Straße.

Das Auto hat einen eigenen Programmierbereich.

Wir programmieren das Auto so, dass es beim Klick auf die grüne Fahne losfährt.

Wie sich ein Objekt nach rechts bewegen kann, haben wir schon kennen gelernt. Wir verändern einfach die x-Koordinate um einen bestimmten Wert.

Um das mehrfach nacheinander zu erreichen, findest du in der Blockpalette unter „Steuerung“ verschiedene Blöcke, die **Wiederholungen** auslösen.



Schleifen:

Strukturen, die Wiederholungen von Anweisungsblöcken bewirken, heißen **Schleifen**.

Scratch bietet drei Schleifen an: „wiederhole ... mal“, „wiederhole fortlaufend“ und „wiederhole bis...“

Aufträge

1. *Teste zunächst die Schleife, die eine bestimmte Anzahl an Durchläufen hat. Den Block mit der Anweisung zum Ändern der x-Koordinate ziehst du einfach in den Schleifenblock. Lass das Auto 10-mal nacheinander ein Stückchen nach rechts fahren.*

2. *Teste dann die Schleife, die so lange Anweisungen wiederholt, bis eine Abbruchbedingung erfüllt ist. Das Auto soll bei uns jetzt so lange fahren, bis es am Rand angekommen ist. Du findest dazu in der Blockpalette unter „Fühlen“ die Bedingung „wird … berührt“. Hier kannst du Rand auswählen. (Was passiert, wenn das Auto den Rand links berührt?)*

Um die beiden Schleifenarten noch ein bisschen weiter zu üben, hole dir das **Aufgabenblatt „Die Katze zeichnet“ (Teil F).**

Für unser geplantes Spiel soll das Auto natürlich nicht am Rand stehen bleiben, sondern möglichst ganz aus dem Bild verschwinden und dann irgendwann wieder auf der linken Seite auftauchen. Und das ununterbrochen.

Um sich im Kopf eine Struktur für einen Programmierschritt zu überlegen, versucht man das, was man erreichen möchte, zuerst in Worte zu fassen.

Hier z.B.: „Falls das Auto rechts aus dem Bild gefahren ist, soll es links wieder ins Bild kommen.“ Das Auto soll also fortlaufend abfragen, ob es am rechten Rand rausgefahren ist. Falls ja, dann soll es von links wieder starten, falls nein fährt es weiter nach rechts.

Verzweigungen

Strukturen, bei der ein Teil ausgeführt wird, wenn eine Bedingung erfüllt ist, und der andere, wenn die Bedingung nicht erfüllt ist, heißen **Verzweigungen**.

3. *Schau dir die Blöcke in der Blockpalette unter „Steuerung“ genauer an. Findest du die Blöcke, die du hier nutzen kannst?*
4. *Überlege dir, was es heißt, dass das Auto rechts aus dem Bild gefahren ist. Es genügt uns ja nicht mehr, dass es nur den Rand berührt.*
5. *Überlege, wie du die Blöcke mit Anweisungen füllen musst, dass das Auto so fährt, wie wir es gerade beschrieben haben. Probiere deine Ideen aus. Ein kleiner Tipp: Schau auch mal in der Blockpalette bei den „Operatoren“ nach. Vielleicht kannst du etwas davon brauchen.*

Wenn du Hilfe brauchst, hol dir die passende Hilfekarte.

Wenn du fertig bist, kontrolliere auch mit der Hilfekarte und vergleiche deinen Weg mit dem vorgeschlagenen.

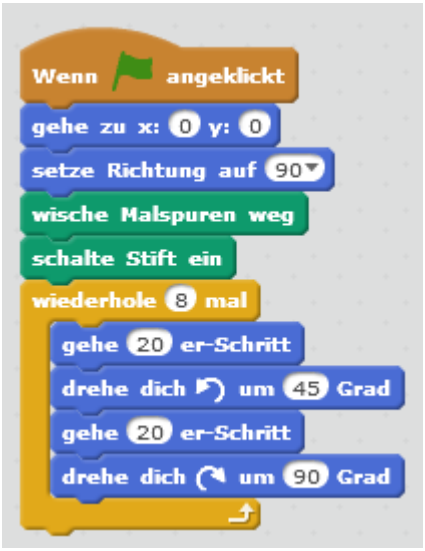
Verändere jetzt den letzten Schleifenblock noch so, dass er beim Start des Programms (beim Klick auf die grüne Fahne) ausgeführt wird und speichere alles ab.

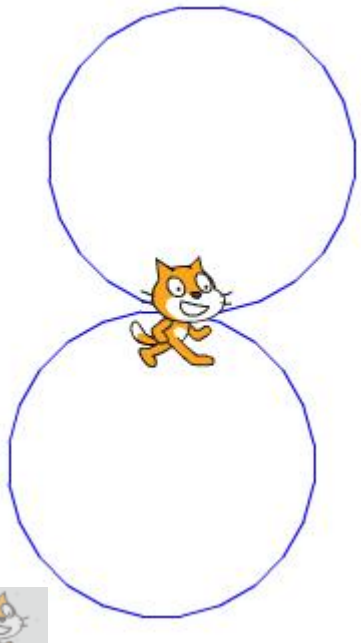
Teil F: Die Katze zeichnet

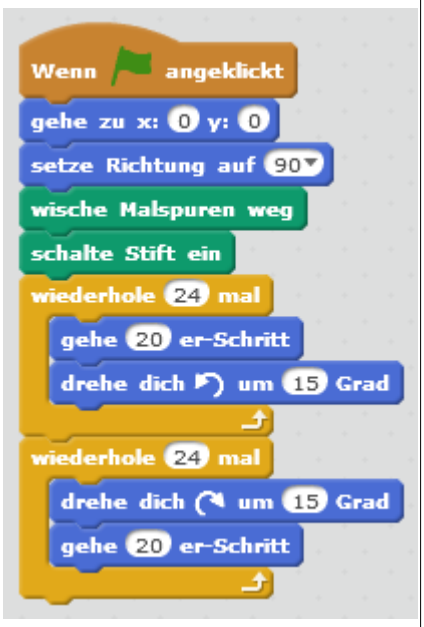
Speichere die folgenden Aufgaben in einer eigenen Datei `05katze_zeichnet`.

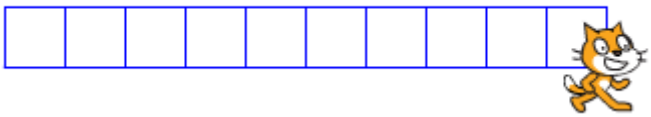
Aufgabe 1:

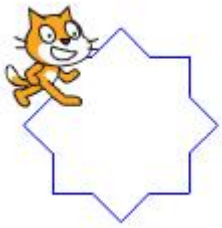
Was passt zusammen? Verbinde und begründe kurz.



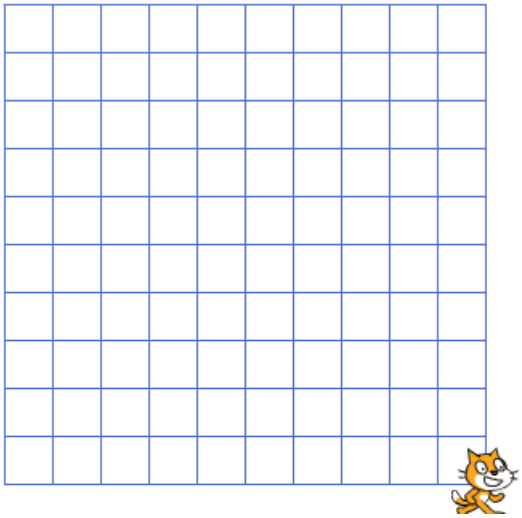








Aufgabe 2:

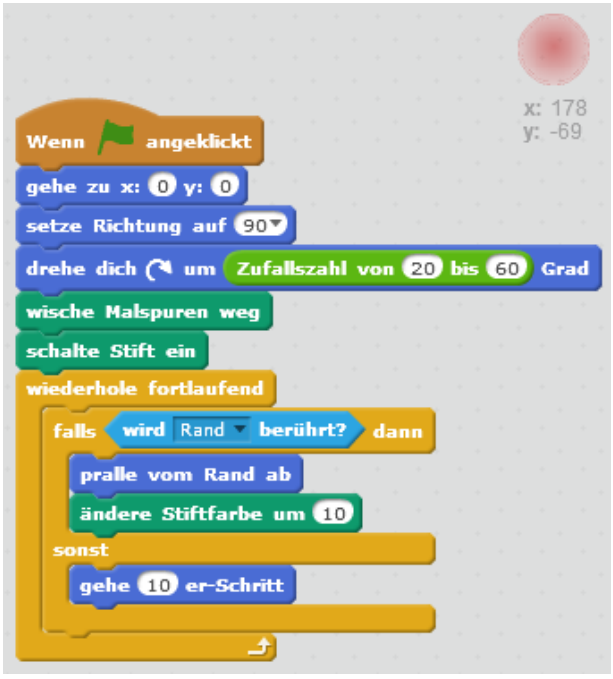


In der ersten Aufgabe gibt es ein Programm, in dem die Katze zehn Quadrate nebeneinander zeichnet. Erweitere das Programm so, dass die Katze zehnmal zehn Quadrate zeichnet.

Aufgabe 3:

Beschreibe, was der folgende Programmcode auslöst.

(Die Katze wurde durch einen Ball ersetzt, was das Abprallen am Rand einfacher macht.)



Zusatzaufgabe 4: **(freiwillig)**

Schreibe ein Programm, das die Katze das „Haus vom Nikolaus“ zeichnen lässt. Verändere es dann so, dass sie fünf Häuser direkt nebeneinander zeichnet.

Teil G: Das Auto ändert die Geschwindigkeit

Öffne die Datei `04katze` und speichere sie unter `06katze`.

Im Moment bewegt sich das Auto bei jedem Schleifendurchlauf um dieselbe Schrittweite vorwärts. Wir werden unser Programm jetzt so abändern, dass wir mit zwei weiteren Tasten das Auto schneller bzw. langsamer machen können.

Dazu geben wir der Schrittweite einen Namen. Da es sich in gewissem Sinne um die Geschwindigkeit des Autos handelt und diese in der Physik die Abkürzung v erhält, nutzen wir hier v1 für die „Geschwindigkeit“ von Auto1. Du könntest auch einen längeren Namen vergeben, z.B. geschwindigkeit_auto1.

Variablen

In der Informatik bezeichnet Wertespeicher als **Variable**. Man benutzt sie, um sich etwas zu merken.

Legt man eine neue Variable an, nennt man das **Deklarieren** der Variable.

Eine Variable kann einen Wert zugewiesen bekommen. Diesen Wert kann man jederzeit verändern. Bekommt eine Variable das erste Mal einen Wert zugewiesen, nennt man das **Initialisierung** der Variable.

Wähle in der Blockpalette „Daten“ und lege eine neue Variable v1 an.

Es werden neue Blöcke erscheinen, die du vermutlich sofort erklären kannst.

Der Wert der Variable wird im linken oberen Eck angezeigt. Im Moment ist das ganz gut, da wir verfolgen können, ob alles so funktioniert, wie wir uns das vorstellen.

Später beim Spiel können wir das Häkchen vor v1 in der Blockpalette entfernen um den Variablenwert nicht mehr auf der Bühne anzuzeigen.

Aufträge:

1. *Initialisiere die Variable v1 beim Start des Programms mit dem bisherigen Wert 10. Überlege dir noch einmal kurz, was die 10 aussagt.*
2. *Wähle dann zwei neue Tastaturereignisse. Die Tasten kannst du selbst wählen, wie du es geschickt findest. Bei der einen Taste erhöhst du den Wert von v1 um einen gewissen Wert, bei der anderen soll etwas subtrahiert werden. (Bei negativer Änderung gibst du der Zahl ein negatives Vorzeichen.)*
3. *Ersetze jetzt an der Stelle, an der x bisher um 10 geändert wurde, die Zahl durch das Symbol der Variable v1.*
4. *Teste dein Programm und speichere es ab.*

Das Auto fährt im Moment noch irgendwann rückwärts oder wird viel zu schnell.

Verbesserungen:

5. *Baue Grenzen für die Variable v1 ein. Ändere v1 nur, wenn die Grenzen noch nicht über- bzw. unterschritten wurden. Nutze dazu die Operatoren (> und <).*
6. *Teste dein Programm, vergleiche es mit der Hilfekarte und speichere wieder ab.*

Wenn mehrere Tastaturereignisse nebeneinander programmiert werden, kann es bei größeren Programmen zu Schwierigkeiten bei der Ausführung kommen.

Wir ändern deshalb den Programmcode für unser Auto. In der Endlosschleife „wiederhole fortlaufend“ befindet sich gerade die Verzweigung, in der die Position des Autos abgefragt und seine Position verändert wird.

Wir wollen in die gleiche Endlosschleife zwei weitere Verzweigungen anhängen.

Falls Taste x gedrückt wird und v1 < 20, dann soll die Geschwindigkeit erhöht werden.

Falls Taste c gedrückt wird und v1 > 5, dann soll die Geschwindigkeit verringert werden.

7. *Probiere, das Gewünschte in deinem Programmcode zu ergänzen.*
   (Mache dich für die gewünschten Blöcke auf die Suche in „Steuerung“, „Fühlen“ und „Operatoren“.)
8. *Speichere das veränderte Programm unter einem neuen Namen* 06katzeV2 *ab und teste hinterher beide Varianten. Merkst du einen Unterschied?*

Du hast für die Bedingung in der Verzweigung ein „und“ verwendet, um zwei Bedingungen zu verknüpfen.

Logische Verknüpfungen

Neben „und“ findest du noch „oder“ und „nicht“ bei den Operatoren. Diese drei nennt man logische Verknüpfungen oder logische Operatoren.

„und“ bedeutet, dass beide Bedingungen gleichzeitig erfüllt sein müssen,

„oder“ bedeutet, dass mindestens eine der beiden Bedingungen erfüllt sein muss.

Teil H: Die Katze muss aufpassen

Jetzt kennst du dich schon so gut aus, dass du erst einmal probieren kannst, wie es weitergeht.

Die Katze möchte ja heil und sicher über die Straße kommen. Wenn das Auto die Katze berührt, soll zunächst einmal das Auto stoppen.

Speichere `06katze` unter `07katze` ab und probiere es aus.

Hast du eine Lösung? Wenn ja, vergleiche sie mit der zugehörigen Hilfekarte.

Jetzt bist du bereit, dein Projekt zu erweitern. Die nächsten Aufgaben sind unabhängig voneinander. Du kannst selbst entscheiden, in welcher Reihenfolge du sie machst. Natürlich kannst du auch eigene Ideen umsetzen, um das Spiel spannender zu machen.

Aufträge zum Spielausbau:

1. *Überlege dir, was passieren soll, wenn die Katze angefahren wird. Setze deine Ideen um.*
2. *Lasse auf der Straße ein weiteres Auto fahren und programmiere alles Nötige für die Kollision mit der Katze.*
3. *Baue einen Zähler ein, der mitzählt, wie oft die Katze heil am oberen Rand angekommen ist.*
4. *Gib der Katze eine bestimmte Anzahl an Leben. (Vielleicht sieben?) Wenn sie diese alle verloren hat, endet das Spiel.*
5. *Erweitere dein Spiel um weitere Level. Diese können immer schwieriger werden. Lege dazu neue Bühnenbilder an und lasse die Katze, wenn sie am oberen Rand angekommen ist, auf die neue Bühne wechseln. Überlege dir auch, was passieren soll, wenn sie erwischt wird. (Start in dem Level? Start in Level1?)*
6. *Füge ein Level mit Fluss ein, auf dem Baumstämme schwimmen, auf die die Katze springen muss, um über den Fluss zu kommen.*

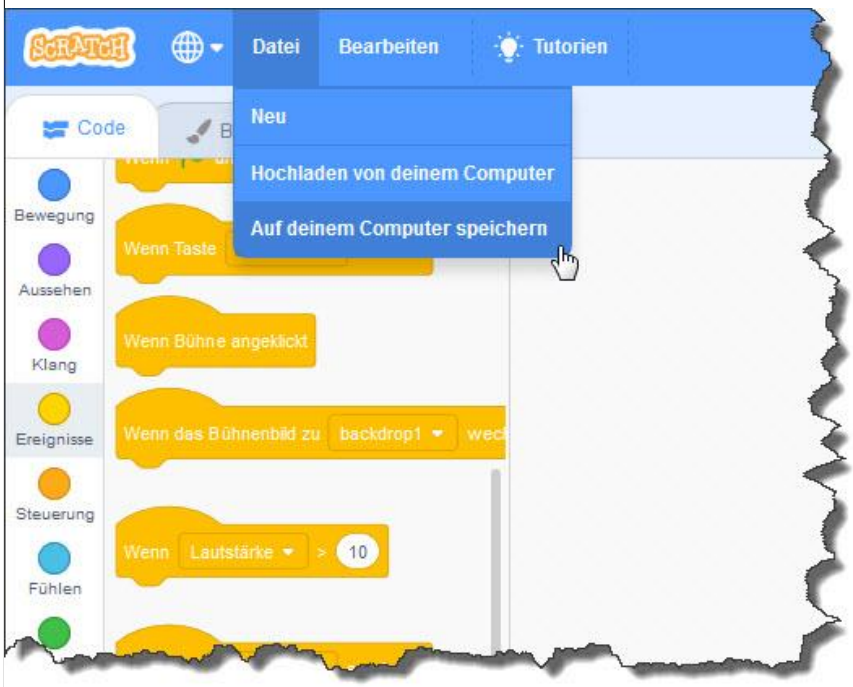
Ein kleiner Tipp:

Objekte können miteinander in Verbindung treten, indem sie sich „Nachrichten“ schicken und auf solche reagieren. Die nötigen Blöcke dazu findest du in der Blockpalette unter „Ereignisse“.

Gib deinen Nachrichten sinnvolle Namen, damit du später noch weißt, wozu sie gut sind.

Hilfekarten zu den Aufgaben – Teil A:

Datei speichern:

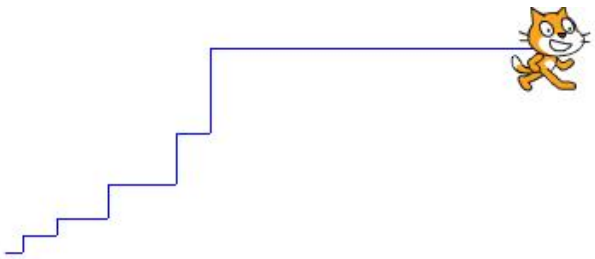


Name des Objektes ändern:

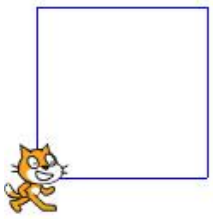
Mögliche Lösungen zu den Aufgaben – Teil B:

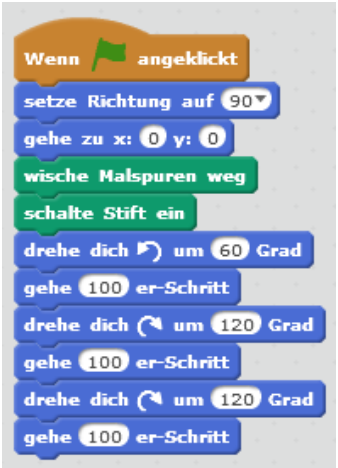
Aufgabe 2:

Aufgabe 3:

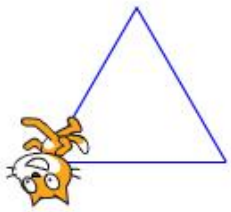


Aufgabe 4:



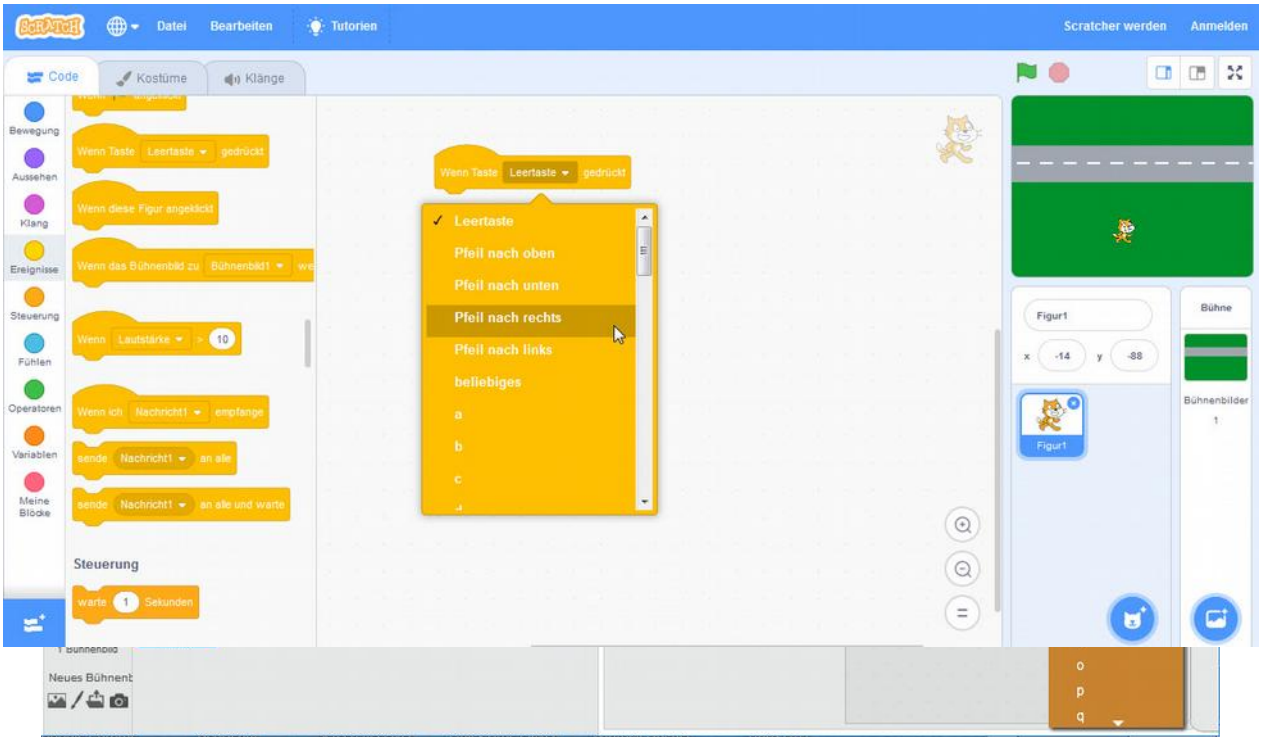


Aufgabe 5:



Hilfekarten zu den Aufgaben – Teil C:

Einstellung der Taste beim Tastaturereignis:



Aufgabe 2:

Teil C:

Aufgabe 3:

Läuft die Katze nach oben oder unten, behält sie das Kostüm bei, das sie gerade anhat.

Hilfekarte – Teil E:

Ergebnis:

Schleifenblock:

Durch „wiederhole fortlaufend“ wird der Inhalt des Schleifenblockes so lange ausgeführt, bis das Programm beendet wird.

Diese Schleife nutzen wir, weil das Auto ja zunächst einmal dauernd fahren soll.

Würden wir nur das „ändere x um 10“ in die Schleife ziehen, würde – wenn Scratch uns da keinen Strich durch die Rechnung machen würde (da bleibt das Auto am Rand hängen) – das Auto immer weiter nach rechts aus der Bühne fahren.

Verzweigungsblock:

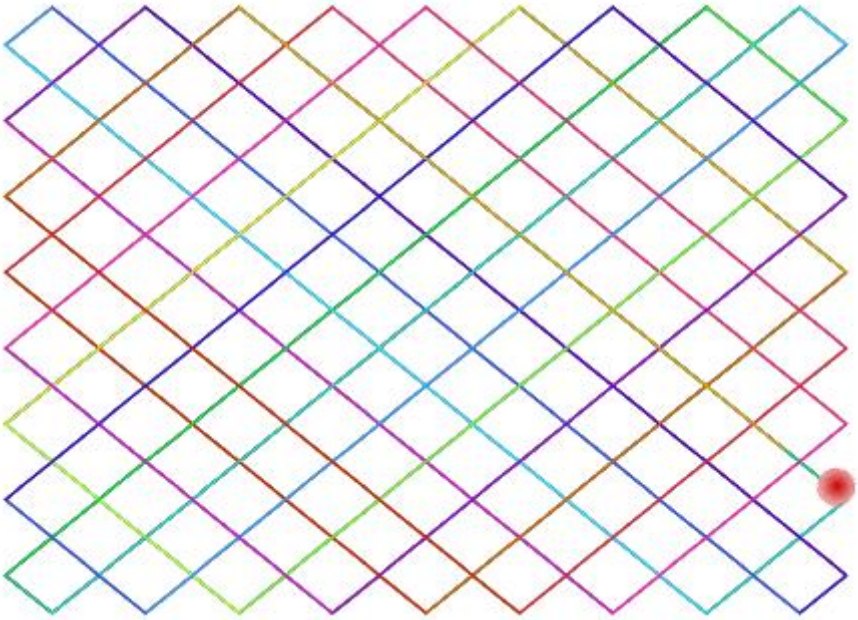
Weil wir wollen, dass das Auto nur fährt, falls es sich noch nicht am rechten Rand befindet, ziehen wir einen Verzweigungsblock in die Schleife.

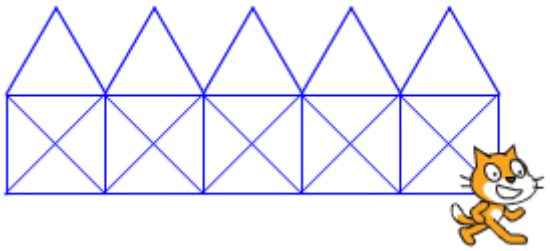
Falls die Bedingung erfüllt ist, wird die Anweisung (oder Sequenz) ausgeführt, die im oberen Teil nach dem „falls“ steht, und ist die Bedingung nicht erfüllt, das was im unteren Teil nach dem „sonst“ steht.

Für solche Verzweigungen müssen wir uns geschickte Bedingungen überlegen. Hier ist es sinnvoll, danach zu schauen, ob die x-Position des Autos kleiner ist als der x-Wert am rechten Rand.

Falls ja, fährt das Auto (ändert seine x-Position um 10); falls nein, wird es links wieder abgesetzt.

Die Katze zeichnet – Lösungsvorschläge – Teil F:

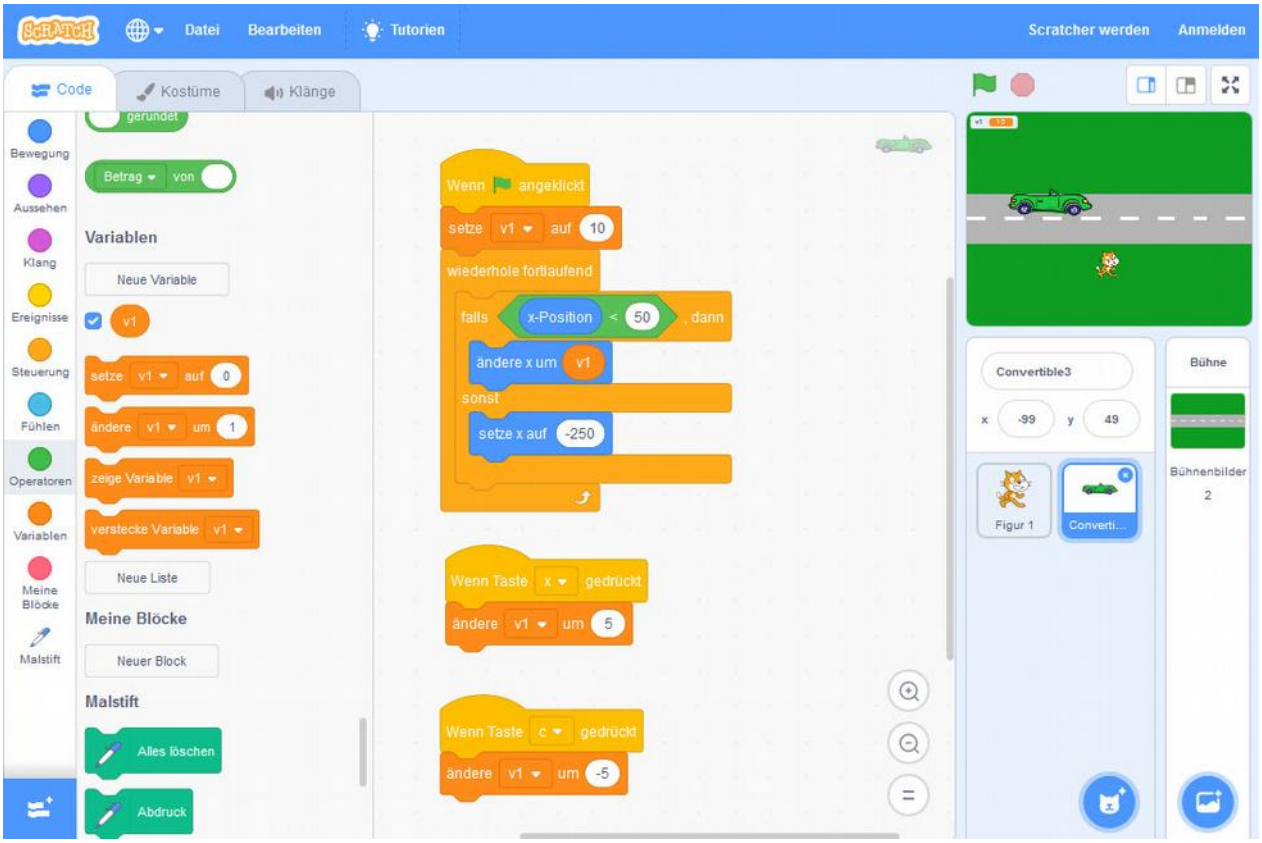





Wenn angeklickt
gehe zu x: -217 y: 0
setze Richtung auf 90
wische Malspuren weg
schalte Stift ein
wiederhole 5 mal
ändere y um 50
drehe dich um 60 Grad
warte 0.5 Sek.
gehe 50 er-Schritt
drehe dich um 120 Grad
warte 0.5 Sek.
gehe 50 er-Schritt
warte 0.5 Sek.
ändere x um -50
setze Richtung auf 90
drehe dich um 45 Grad
warte 0.5 Sek.
gehe 70 er-Schritt
warte 0.5 Sek.
ändere x um -50
drehe dich um 90 Grad
warte 0.5 Sek.
gehe 70 er-Schritt
warte 0.5 Sek.
ändere y um -50
warte 0.3 Sek.
setze Richtung auf 90
Das
ist
das
Haus
vom
Ni...
...ko...
...laus

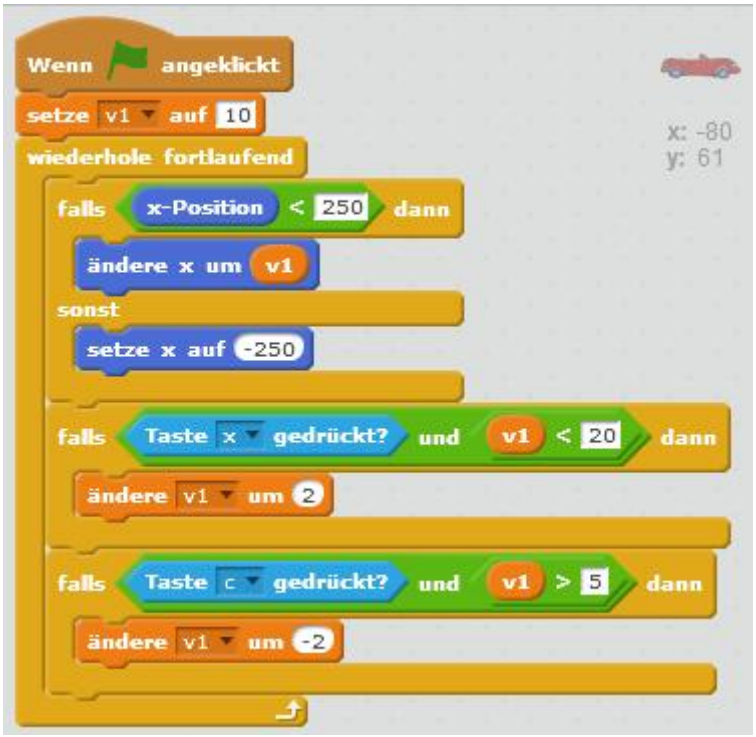
Hilfekarte – Teil G:

Aufgaben 1 – 4:



Aufgabe 5:

Aufgabe 7:


Wenn angeklickt
setze v1 auf 10
wiederhole fortlaufend
falls x-Position < 250 dann
ändere x um v1
sonst
setze x auf -250
falls Taste x gedrückt? und v1 < 20 dann
ändere v1 um 2
falls Taste c gedrückt? und v1 > 5 dann
ändere v1 um -2
x: -80
y: 61

Hilfekarte – Teil H:

Auto stoppt, wenn die Katze berührt wird: