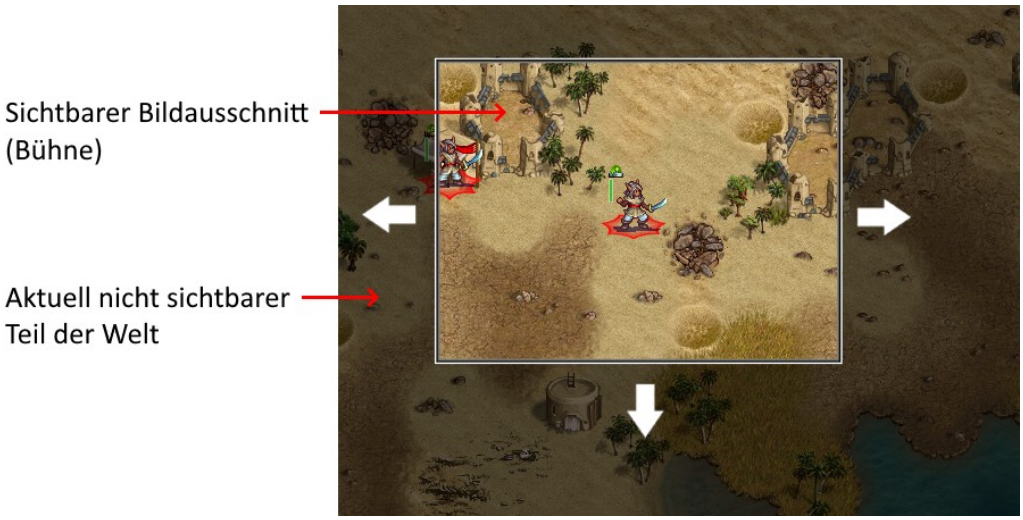
Sichtbarer Bildausschnitt

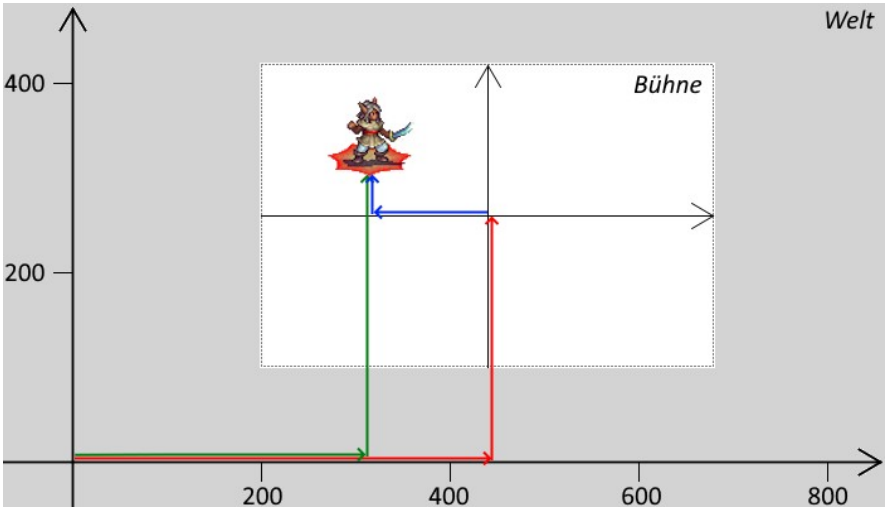
In vielen Computerspielen ist die **Spielwelt**, in der sich die Spieler bewegen, viel größer als der sichtbare **Bildausschnitt** (in Scratch: die **Bühne**). Es ist immer nur ein Teil der Spielwelt sichtbar. Wenn der Spieler sich in der Spielwelt bewegt, verschiebt sich dabei auch der sichtbare Bildausschnitt, so dass man zu verschiedenen Zeitpunkten unterschiedliche Teile der Spielwelt sieht. Dieses Verschieben des sichtbaren Bildausschnitts nennt man „**Scrolling**“.



Zwei Koordinatensysteme

Wenn man mit einem Bildausschnitt arbeitet, der in einer größeren Spielwelt „scrollt“, hat man es mit zwei Koordinatensystemen (KOS) zu tun:
Das **KOS der Welt** und das **KOS der Bühne** (also des sichtbaren Ausschnitts).

In der folgenden Abbildung ist eine Figur dargestellt, die auf der Bühne sichtbar ist. Sie hat Koordinaten in der Welt (grün). Der Ursprung des KOS der Bühne hat ebenfalls Koordinaten in der Welt (rot). Die Koordinaten der Figur auf der Bühne sind in blau dargestellt.



Da die Figuren im KOS der Bühne gezeichnet werden, muss man die Koordinaten jeder Figur auf der Bühne ausrechnen – was auf eine simple Subtraktion hinausläuft:

$$X_{\text{Figur auf der Bühne}} = X_{\text{Figur in der Welt}} - X_{\text{Bühne in der Welt}}$$
$$Y_{\text{Figur auf der Bühne}} = Y_{\text{Figur in der Welt}} - Y_{\text{Bühne in der Welt}}$$

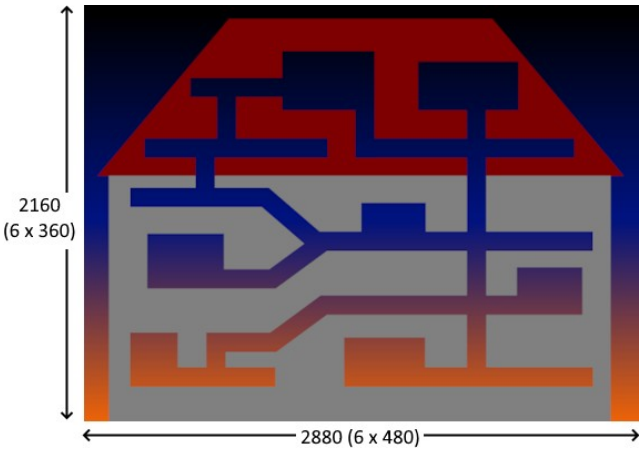
Im Beispiel auf Seite 1 hat die Figur in der Welt die Koordinaten X = 300, Y = 320.
Der Ursprung der Bühne hat in der Welt die Koordinaten X = 440, Y = 280.

Um die Figur auf der Bühne an der richtigen Stelle zu zeichnen, berechnet man daraus die Koordinaten auf der Bühne: X = 300 – 440 = –140, Y = 320 – 280 = 40.

Also sind die Bühnenkoordinaten der Figur X = –140, Y = 40

Große Bilder in Scratch einfügen

Als Beispiel entwickeln wir ein Spiel, in dem eine Fledermaus durch ein großes Haus fliegt und dabei Schätze aufsammeln und Gespenstern ausweichen muss.

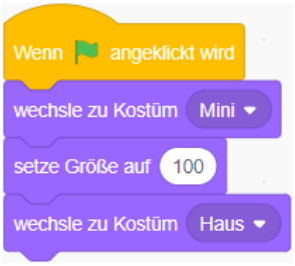


Das Haus mit seinem Gängen ist größer als die Bühne und wird daher gescrollt – in diesem Beispiel ist das Bild für das Haus in x- und y-Richtung sechs mal so groß wie die Bühne (es hat also die 36-fache Fläche der Bühne).

Leider verkleinert Scratch Bilder im JPG- oder PNG-Format automatisch auf eine Größe von 480 x 360 Pixeln, so dass sie in die Bühne passen, statt (was eigentlich sinnvoll wäre) sie in Originalgröße zu belassen. Außerdem blockiert Scratch die Möglichkeit, solche Bilder einfach wieder zu vergrößern. Man kann sich aber mit folgendem Trick behelfen:

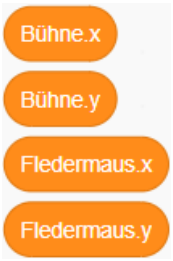
1. Man speichert das Bild als **Vektorgrafik**. Dazu öffnet man das Bild mit einem Zeichenprogramm, z.B. mit dem kostenlosen **Inkscape** und speichert es im SVG-Format (scalable vector graphics)[1]. Anschließend importiert man die SVG-Datei in Scratch.

2. Scratch wird die Grafik dann verkleinern, so dass sie in die Bühne passt. Um sie wieder auf 100 % zu vergrößern, erhält die Figur (in diesem Fall das Haus) ein **zusätzliches, kleines Kostüm**. Zu Beginn wechselt man zu diesem Mini-Kostüm, vergrößert die Figur auf 100% und wechselt dann zum eigentlichen Kostüm. Das hat dann die Originalgröße, in diesem Beispiel 2880 x 2160 Pixel.



1 Das Bild wird dabei nicht wirklich in eine Vektorgrafik umgewandelt, sondern nur als Bitmap „eingebettet“. Es ist die gleiche Bildinformation enthalten, nur mit einer anderen Dateiendung – doch das genügt in diesem Fall.

Variablen



Um mit den zwei Koordinatensystemen zu hantieren, braucht es einige Variablen:

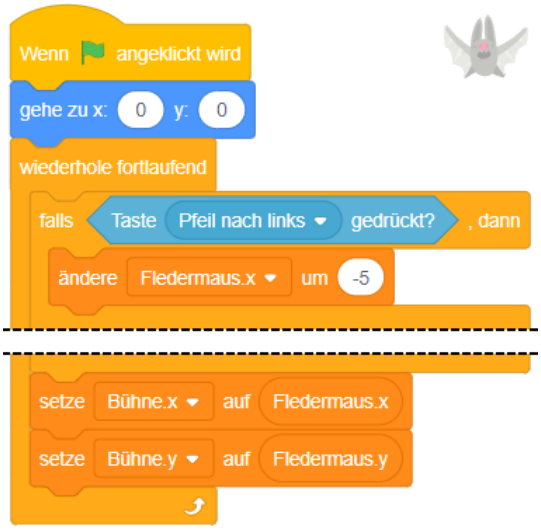
- Jede **Figur** hat ein Paar (x / y) ihrer Koordinaten in der Welt.
- Die **Bühne** benötigt ebenfalls Variablen für ihre Koordinaten in der Welt.

Die Koordinaten jeder Figur auf der Bühne werden aus diesen Variablen berechnet, entsprechend der Formel auf Seite 2.

Bei Spielen mit mehreren Figuren verliert man bei den vielen Variablen schnell die **Übersicht**. Von den genannten Variablen müssen nur die Koordinaten der Bühne für alle Figuren sichtbar sein, da sie für die Formel gebraucht werden. Die x-/y-Position einer Figur wird nur von ihr selbst gebraucht, daher sollten diese Variablen nur für die jeweilige Figur sichtbar sein.

Außerdem sollte man die Variablen nicht einfach „x“ oder „y“ nennen, denn davon gibt es ja jetzt mehrere. Nützlich ist hier die **Punkt-Schreibweise**: Bühne.x, Fledermaus.x usw. Diese Schreibweise kann man für alle Variablen anwenden und so jeweils angeben, zu welcher Figur bzw. zu welchem Kontext sie gehören.

Bewegung von Hauptfigur und Bühne



Es gibt verschiedene Möglichkeiten, den sichtbaren Bildausschnitt in der Welt, sprich die Bühne, zu bewegen. Die einfachste ist, die Bühne synchron mit der vom Spieler bewegten Figur (hier: mit der Fledermaus) zu halten. D.h. die Fledermaus ist immer genau in der Mitte der Bühne, und wenn sie sich bewegt, scrollt die Bühne in der Welt immer genau dorthin, wo sich die Fledermaus gerade befindet.

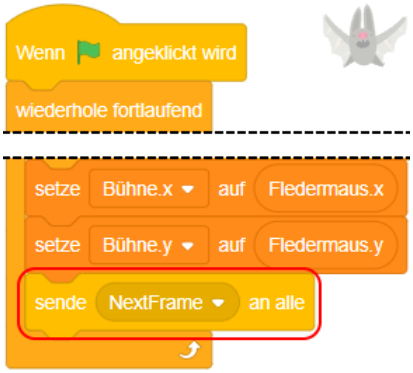
Im abgebildeten Skript wird die Fledermaus mit den Pfeiltasten nach links / rechts / oben / unten bewegt. Die Bewegung drückt sich allein durch eine Änderung der Variablen aus: Die Pfeiltasten ändern die Position (x / y) der Fledermaus, und am Ende des Skripts wird die Bühne auf die gleiche Position (x / y) gesetzt.

Da die Fledermaus immer in der Mitte der Bühne zu sehen ist, brauch man sie nur einmal auf die Bühnenposition (0 / 0) zu setzen und diese nicht mehr zu ändern. Die anderen Figuren, insbesondere die „Welt“, also das Haus mit seinen Gängen, werden dann entsprechend der Variablen Bühne.x und Bühne.y gezeichnet.

Figuren zeitgleich bewegen

In den bisherigen Scratch-Projekten hatte jede Figur eigene Wiederholschleifen, in denen Bewegung, Animation usw. abliefen. Wenn man beim Scrolling aber Figuren mithilfe von Variablen bewegt und anhand dieser Variablen die Position auf der Bühne berechnet, funktionieren diese parallel ablaufenden Wiederholschleifen nicht mehr gut, denn es ist nicht klar, wann die Schritte der verschiedenen Schleifen genau ablaufen.

Eine bessere Alternative bietet der Einsatz von Nachrichten: Es gibt nur eine einzige Wiederholschleife, die im Skript der Hauptfigur abläuft. Diese berechnet in jeder Wiederholung zunächst die aktuellen Bühnenkoordinaten und sendet dann eine Nachricht an alle Figuren, dass ein neuer „Frame“ ansteht, also ein neues Zeitfensterchen für einen Bewegungsschritt.



Alle Figuren reagieren auf diese Nachricht, indem sie einen Bewegungsschritt (und ggf. einen Animationsschritt) machen, und dann ihre Position auf der Bühne ausrechnen. So kann man davon ausgehen, dass alle Figuren mit den gleichen Bühnenkoordinaten gezeichnet werden.

Bewegung der Welt

Die Figur der Welt (also das Haus mit den Gängen) reagiert die Nachricht, die die Schleife der Hauptfigur in jeder Wiederholung sendet: sie wird an die Position gescrollt, zu der sich die Hauptfigur bewegt hat.

Im Koordinatensystem der Welt befindet sich die Welt selbst an Position $x = 0$, $y = 0$.
Um ihre Bühnenkoordinaten auszurechnen, wendet man die Formel auf Seite 2 an:

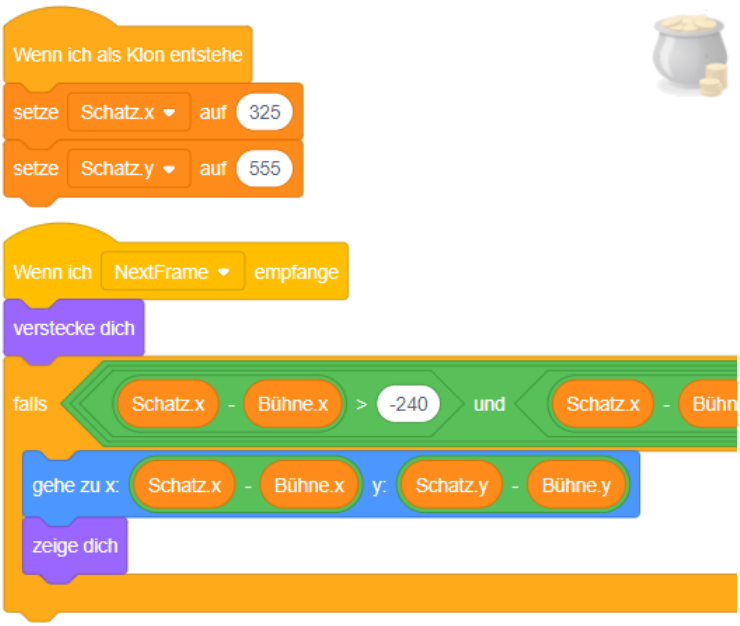
$$X_{\text{Figur auf der Bühne}} = X_{\text{Figur in der Welt}} - X_{\text{Bühne in der Welt}} = 0 - X_{\text{Bühne in der Welt}}$$
$$Y_{\text{Figur auf der Bühne}} = Y_{\text{Figur in der Welt}} - Y_{\text{Bühne in der Welt}} = 0 - Y_{\text{Bühne in der Welt}}$$

Die Figur der Welt muss also nur jedesmal, wenn sich die Bühnenkoordinaten ändern, ihre Position auf der Bühne ändern, die einfach den negativen Bühnenkoordinaten entspricht.

Scrollen unbewegter Figuren

Im Spiel gibt es Töpfe voll Gold, die die Fledermaus aufsammeln kann. Diese werden zu Beginn des Spiels als Klone erzeugt und in den Gängen des Hauses platziert. Sie bleiben an Ort und Stelle, solange bis die Fledermaus sie gefunden hat.

Nebenstehend siehst du ein Beispiel: Einer der Goldtöpfe erhält als Position in der Welt x = 325, y = 555.



Man kann die Bühnenkoordinaten mit der Formel berechnen:

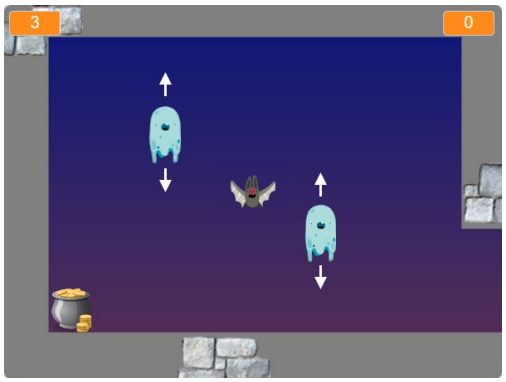
x = Schatz.x – Bühne.x und
y = Schatz.y – Bühne.y.

Ein Problem ist aber, dass Scratch die Koordinaten von Figuren, die außerhalb der Bühne liegen, automatisch korrigiert, so dass sie wieder innerhalb der Bühne sichtbar sind. Der Goldtopf soll aber nur dann gezeichnet werden, wenn die Fledermaus in seiner Nähe ist, und nicht, wenn sie sich in einem ganz anderen Teil des Hauses befindet, in dem man den Topf nicht sehen kann.

Die Lösung ist, den Topf zu verstecken, wenn er außerhalb des sichtbaren Bereichs liegt. Er ist sichtbar, wenn seine Bühnenkoordinaten im Bereich $-240 < x < 240$ und $-180 < y < 180$ liegen. Man prüft also, ob die Bühnenkoordinaten innerhalb dieses Bereichs liegen, und zeigt den Topf nur dann an. Ansonsten wird er versteckt.

Scrollen bewegter Figuren

Im Haus soll es Geister geben, die in Räumen oder Gängen auf und ab schweben und die die Fledermaus nicht berühren darf.



Da sich die Geister im KOS der Welt bewegen, und die Bühne unabhängig von ihnen durch den Spieler bewegt wird, kann man die bisher verwendeten Blöcke **gehe ( ) er Schritt**, **drehe dich um ( ) Grad** usw. nicht benutzen, denn sie verändern die Bühnenkoordinaten, nicht die Koordinaten der Geister in der Spielwelt.

Stattdessen müssen die Welt-Koordinaten der Figur, also die entsprechenden Variablen verändert werden, und zusammen mit den Koordinaten der Bühne in der Welt wird dann die Position auf der Bühne ausgerechnet.

Auch den Block **wird (Haus) berührt?** kann man nicht für Kollisionen mit dem Haus verwenden, um z.B. die Richtung zu ändern. Der Test auf Berührung funktioniert zwar, wenn der Geist vollständig auf der Bühne zu sehen ist. Wenn er aber durch Scrolling nur teilweise oder gar nicht zu sehen ist, prüft Scratch diese Berührung nicht mehr korrekt, und kann insofern nicht feststellen, ob der Geist die Richtung ändern muss. Er würde dann einfach durch die Decken bzw. Fußböden hindurch schweben (was im Fall von Geistern vielleicht vorstellbar ist, aber allgemein braucht man schon eine Möglichkeit, eine Kollision auch außerhalb der sichtbaren Bühne festzustellen).

Eine Möglichkeit ist, sich hier auch wieder mit **Variablen** zu behelfen. Die Bewegung ist rechts simpel: Geister bewegen sich zwischen Boden und Decke auf und ab. Sie ändern also nur ihre y-Koordinate.

Ihre **Bewegungsrichtung** kann man durch eine Variable ausdrücken: dy[2]= 5 bedeutet, dass sich die y-Koordinate in jedem Schritt um 5 vergrößert, dy = –5 entsprechend, dass die y-Koordinate sich jeweils um 5 verringert.

Auch die **Kollision** mit Boden bzw. Decke kann mithilfe von zwei Variablen geprüft werden: min-y drückt den kleinsten y-Wert aus, den der Geist haben darf – wenn seine y-Koordinate also kleiner ist, muss er seine Richtung ändern. Entsprechend drückt max-y den größten erlaubten Wert aus, wenn y noch größer ist, wird die Richtung geändert.

Im abgebildeten Skript sieht man das Zusammenspiel dieser Variablen.

Auch hier gilt: Achte darauf, dass die Variablen nur für diese Figur sichtbar sind, und dass sie mit „Geist . Variable“ benannt sind, um den Überblick zu behalten.

Zu Beginn, also wenn die Klone der Geister entstehen, muss man all diese Variablen mit Anfangswerten versehen, was bei vielen Geistern schon recht aufwendig ist – aber das Ergebnis kann sich dann auch sehen lassen!

Die Umrechnung der Welt-Koordinaten in Bühnenkoordinaten funktioniert dann genauso wie bei den unbewegten Figuren, einschließlich der Prüfung, ob die Figur gerade sichtbar ist.

2 In der Mathematik nutzt man den Buchstaben d (für delta), um Änderungen auszudrücken. dy = 5 bedeutet also soviel wie „ändere y in jedem Schritt um 5“.

Aufgabe

Verwende die ausgeteilte Vorlage „Spukhaus.sb3“. Sie enthält fast alle Figuren, die benötigten Variablen und ein paar grundlegende Skripte für das auf den vorigen Seiten beschriebene Spiel.

a) In der Vorlage ist das Haus enthalten, es fehlt aber der **Hintergrund**.
Er hat die gleiche Größe und soll genau wie das Haus gescrollt werden.

   i. Um den Hintergrund als übergroße Grafik in Scratch zu importieren, musst du die Bilddatei erst in eine SVG-Datei umwandeln (siehe S. 2).

   ii. Anschließend kannst du die Grafik als Figur importieren. Zusätzlich benötigt die Figur ein Mini-Kostüm, damit du die Grafik auf 100% vergrößern kannst.

   iii. Im Skript sollte der Hintergrund auf die hinterste Ebene geschoben werden, damit sie hinter dem Haus und den Figuren gezeichnet wird.

b) Programmiere zuerst die **Bewegung der Fledermaus** und der Bühne.
Die Fledermaus wird mit den Pfeiltasten x- und y-Richtung bewegt.
Lasse Kollision mit und anderen Figuren zunächst weg.
Ziel ist, dass sich die Fledermaus völlig frei bewegen kann und Haus und Hintergrund korrekt scrollen. Wenn du die Variablen für ihre x- und y-Koordinaten auf der Bühne anzeigst, kannst du die Fledermaus benutzen, um die Koordinaten für neue Klone der Geister und der Schätze zu ermitteln.

c) Programmiere die **unbewegten Schätze**. Falls sie beim Scrollen sichtbar sind, sollen an der richtigen Stelle gezeichnet werden.
Ergänze den Code für weitere Klone: die Schätze sollten am Ende von Gängen oder in Ecken von Räumen stehen, wo sie von Geistern bewacht werden.

d) Programmiere die **Geister**. Der Code für die Erzeugung von zwei Klonen (im Raum unterhalb des Startpunktes der Fledermaus) ist vorgegeben.
Entsprechend der Anleitung auf S. 6 bewegen sie sich zwischen Boden und Decke hin und her.
Ergänze den Code für weitere Klone in anderen Räumen und Gängen.

e) Zuletzt programmiere die **Kollisionen**: Wenn die Fledermaus das Haus oder die Geister berührt, verliert der Spieler ein Leben, und muss wieder am Ausgangspunkt starten.
Wenn sie einen der Schätze berührt, erhält der Spieler Punkte, und der Schatz verschwindet.