

IT4Kids Modulhandbuch
© 2015 Enactus Aachen e. V.
1. Vorabversion

Informatik und Programmieren für Kinder

**Modulhandbuch für die Verwendung in
Informatikkursen der Klassen drei bis sechs**

Enactus Aachen e. V. IT4Kids

20. Januar 2016

Zusammengetragen und editiert von Steffen
Schneider, Lucas Mosig, Dimitri Rusin und
Leonhard Hetz

1. Vorabversion vom 20. Januar 2016

Dieses Material steht unter der Creative-Commons-Lizenz Namensnennung-Nicht kommerziell 4.0 International. Um eine Kopie dieser Lizenz zu sehen, besuchen Sie <http://creativecommons.org/licenses/by-nc/4.0/>.



”I think everyone in this country should learn to program a computer. Everyone should learn a computer language because it teaches you how to think. I think of computer science as a liberal art.“ - *Steve Jobs*

Vorwort

Informatik und Programmierung ist ein spannendes Fachgebiet, das eine hohe Bedeutung in unserem Alltag hat. Informatik und Programmierung ist überall gegenwärtig und wird ständig genutzt - dennoch ist das Angebot für informatische Bildung an Schulen in Deutschland oftmals kaum oder gar nicht vorhanden.

An vielen Schulen fehlen die Kapazitäten, sei es personell, sei es finanziell - dabei ist Informatik und Programmierung als spannendes Ergänzungsfach schon ab der dritten Klasse denkbar.

Ziel dieses Buchs soll sein, eine Basis für die Durchführung von Informatik-Kursen für Kinder im Grundschulalter und in der frühen Sekundarstufe I zu bieten. Auf diese Weise können Sie als Lehrperson auch ohne tiefgreifende Informatikkenntnisse schnell Anregungen finden, wie Schülerinnen und Schüler an dieses spannende Feld herangeführt werden können.

Die Struktur des Buches ist bewusst offen gehalten und bietet eine Basis für einen sehr individuellen, auf Schülerinteressen angepassten Unterricht. Inhalt sind Projekte unterschiedlicher Themengebiete, die den Schülerinnen und Schülern auf vielfältige Weise die Grundkonzepte des Programmierens näher bringen. Mit steigender Lernkurve kann auch die Komplexität der Projekte schrittweise erhöht werden.

Das vorliegende Konzept wurde in einem einjährigen Modellversuch an einer Grundschule in Aachen im Jahr 2013/14 getestet und im Schuljahr 2014/15 an über zehn weiteren Schulen erfolgreich erprobt.

Unser Ziel ist es, Informatik flächendeckend an jeder Schule zu einem festen Bestandteil im Kursangebot zu machen. Die gewählten Projekte erfordern minimalen Einarbeitungsaufwand für Sie als Betreuerin oder Betreuer, werden aber zugleich alle denkbaren Schülerinteressen abdecken können. Überdies sind wir stets bemüht, die Inhalte des Modulhandbuchs weiter auszubauen.

In diesem Sinne hoffe ich, dass dieses Werk Ihnen als Lehrperson eine Hilfe ist, einen spannenden Kurs vorzubereiten und den Schülerrinnen und Schülern an Ihrer Schule die Möglichkeit zu bieten, einen erfolgreichen Einstieg in die Welt der Informatik zu finden.

Mit freundlichen Grüßen

Steffen Schneider
Aachen, den 20. Januar 2016

Inhaltsverzeichnis

1 Vorbereitung

1.1 Zu den Projekten

Die Projekte gliedern sich in verschiedene Kategorien. Ziel ist es nicht, eine starre Unterrichtsstruktur vorzugeben. Aus bisheriger Erfahrung lässt sich sagen, dass die Auswahl der Materialien und Themen für den Kurs sehr entscheidend von der Kurszusammensetzung abhängt und dass eine korrekte Auswahl unabdingbar für das Gelingen des Kurses sein kann.

Aus diesem Grunde verfolgen wir in diesem Handbuch ein modulares Konzept: Die Projekte sind nicht allgemein nach Niveau (Anfänger, Fortgeschritten etc.) gegliedert, sondern in verschiedene Kompetenzbereiche unterteilt. Es gibt verschiedene Möglichkeiten, alle Bereiche abzudecken.

Trotz dieser bewusst offen gehaltenen Grundstruktur sollen an dieser Stelle einige beispielhafte Abläufe genannt werden, um die Kursplanung zu vereinfachen und einen schnellen Start zu ermöglichen.

1.1.1 Kursvorschlag: Grundschule

- Einstieg: Labyrinth im Klassenraum
- Erstes Projekt: Maus zum Käse
- Geschichten
- Klänge
- Aquarium

1.1.2 Kursvorschlag: Weiterführende Schule

- Einstieg: Labyrinth im Klassenraum
- Erstes Projekt: Labyrinth

- Einführungsprojekte
 - Zeichnen und Farben
 - Klänge
 - Botschaften (zB Tanzstunde oder Bananen)
- Wettrennen
- Pong

1.2 Notwendige Tools

Für die Durchführung der hier vorgestellten Projekte im Kurs kommen verschiedene Werkzeuge infrage, die im Folgenden vorgestellt werden sollen.

1.2.1 Scratch Editor

Scratch ist eine am MIT entwickelte, quelloffene Programmierumgebung. Scratch ist sowohl als Online- als auch als Offline-Version frei erhältlich.

es sowohl online als auch offline. Scratch ist die Standardumgebung, auf die für die Projekte in diesem Konzept zurückgegriffen wurde. Auf Scratch kann online unter scratch.mit.edu zugegriffen werden.

Alle hier genannten Vorlagen mitsamt Beschreibungstext und Musterlösung sind online unter it-for-kids.org/projects abrufbar. Zudem liegen die aktuellen Projektstände auf der mitgelieferten CD bereit.

Das Laden von Projekten in Scratch erfolgt durch Druck auf "Entwickeln" auf der Startseite. Wir empfehlen, im Kursbetrieb diese Seite als Lesezeichen oder Verknüpfung auf den Schülerrechnern zu hinterlegen.

1.2.2 IT4Kids Online Editor

Als Alternative zu Scratch sind wir bemüht eine eigene, auf die konkreten Bedürfnisse des Kursbetriebes zugeschnittene Entwicklungs-umgebung anzubieten. Die Umgebung befindet sich derzeit in der aktiven Entwicklung und ist eine veränderte Version der ebenfalls quelloffenen Software *Snap!*. Auf den IT4Kids Online Editor kann unter code.it-for-kids.org zugegriffen werden.

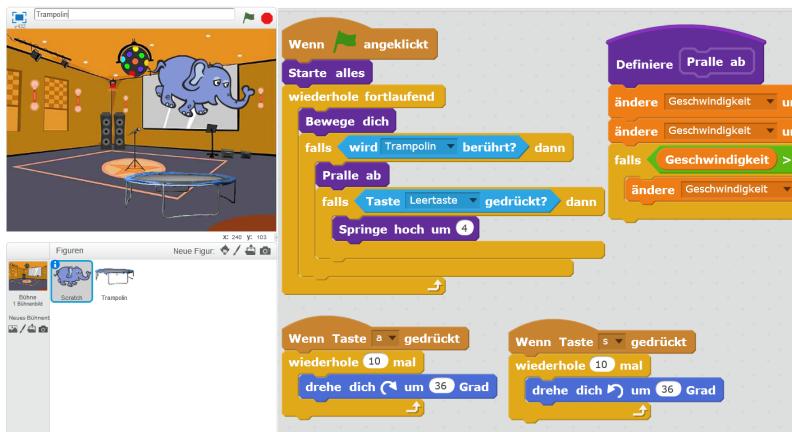


Abbildung 1.1: Scratch Online Editor

1.2.3 IT4Kids Offline Editor

Ebenfalls in der aktiven Entwicklung befindet sich der IT4Kids Offline Editor. Der aktuelle Entwicklungsstand kann auf unserer Homepage eingesehen werden.

1.3 Vor Kursbeginn

Vor Beginn eines IT4Kids Kurses empfehlen sich einige Vorkehrungen, die in diesem Abschnitt besprochen werden.

1.3.1 Verfügbarkeit eines Beamers

Bei vielen Projekten bietet es sich an, die fertigen Programme von den Schülerinnen und Schülern präsentieren zu lassen. Außerdem eignet sich ein Beamer sehr gut, um zu Beginn des Kurses das Stundenziele sowie das fertige Projekt zu zeigen. Sollte kein Beamer zur Verfügung stehen können Projekte an einem gut einsehbaren Rechner gezeigt werden. Manche Konzepte lassen sich auch gut an einer Tafel oder einem OHP demonstrieren.

1.3.2 Infrastruktur zum Abspeichern von Projekten

Zum Abspeichern von Projekten eignen sich grundsätzlich zwei Ansätze: Scratch unterstützt das manuelle Hoch- und Herunterladen von Projekten. Sofern an der Schule ein Netzwerkordner vorhanden ist, kann dieser für die Verteilung von Vorlagen und auch zum Abspeichern von Projekten verwendet werden.

Der zweite Ansatz ist eine Anmeldung auf der Scratch Homepage. Hier kann sich jede Schülerin und jeder Schüler einen eigenen Account zulegen. Einmal angemeldet, wird das jeweils aktuelle Projekt jeweils automatisch in der Cloud gespeichert.

In jedem Fall sollte im Vorfeld mit der Schule die Verfügbarkeit eines Netzwerkordners abgestimmt werden.

Auch die Verwendung von Schüler-USB-Sticks ist denkbar, sodass jeder Schüler seine Fortschritte für zuhause speichern kann. Es empfiehlt sich jedoch, die Projekte auch noch an einem anderen Ort zu speichern, da die Schüler ihre Sticks u.U. zuhause vergessen.

1.3.3 Vor Stundenbeginn

Als Kursbetreuer sollte vor dem Kurs das Projekt ausgewählt und verstanden werden. Dazu reicht es in der Regel sich das Lehrkonzept durchzulesen und die Programmierung in der Musterlösung nachzuvollziehen. Bei komplexeren Projekten kann es hilfreich sein sich bestimmte Programmteile zu notieren oder auf ein Cheat-sheet zurückzugreifen. Erfahrungsgemäß ist es als Kursbetreuer ratsam sich deutlich vor Stundenbeginn im Kursraum einzufinden. So können die bisweilen etwas langsameren Schulrechner in aller Ruhe hochgefahren werden und auch Projekte und Scratchumgebung geöffnet werden. Dies kann andernfalls eine nicht unbedeutliche Zeit in Anspruch nehmen, was unter den Schülern zu Unruhe oder Unmut führt.

2 Informatik mit Stift und Papier

In den ersten ein bis zwei Unterrichtsstunden sollte den Schülern, insbesondere wenn diese noch keinerlei Zugang zur Informatik hatten, kurz dargelegt werden, was Informatik eigentlich ist und mit welchen Mitteln im Unterricht gearbeitet wird. Dazu kann die Lehrperson kreativ werden oder aber auch eine Einführung aus diesem Buch verwenden.

2.1 Zauberwürfel

Materialien: Ein Zauberwürfel (“Rubicks Cube”)

Schwierigkeit: Eher für ältere Schüler

Der Zauberwürfel sollte bis auf einige verbleibende Drehungen gelöst sein, auf jeden Fall muss die Lösung relativ offensichtlich sein. Einem Schüler werden die Augen verbunden, er erhält anschließend den Würfel, den er vor sich auf den Tisch legt.

Anschließend haben die Kinder die Möglichkeit, nacheinander dem Probanden Anweisungen zu geben. Dazu kann das vorliegende Spektrum an “Befehlen” genutzt werden:

- Drehe nach oben: Der Schüler dreht den Würfel so, dass die von ihm weggerichtete Seite nun in Richtung Decke zeigt
- Drehe nach rechts: Der Schüler dreht den Würfel so, dass die von ihm nach links ausgerichtete Seite nun in Richtung Decke zeigt
- Verändere: Der Schüler dreht die obere “Scheibe” des Würfels im Uhrzeigersinn

Die Befehle sind bewusst auf ein Minimum reduziert, gerade so, dass das Problem lösbar ist. Während dem Schüler die Befehle mitgeteilt werden, schreibt die Lehrperson oder ein weiterer Schüler die Befehle an die Tafel. Nach einiger Zeit sollte das Problem gelöst sein, wenn vielleicht auch nicht optimal. Dann wird der Würfel an einen weiteren Schüler gegeben (wieder entsprechend präpariert), der die Augenbinde erhält. Ein weiterer Schüler liest das Programm vor, das anschließend umgesetzt wird.

Optional ist es möglich, die Schüler mit der Frage zu konfrontieren, wie der erste Zustand wiederhergestellt werden kann und wie sich das Problem noch effizienter lösen lässt.

2.2 Labyrinth

Wie Zauberwürfel, nur mit einem Labyrinth mit Stift und Papier. Mögliches Szenario: Zuerst sieht man das Labyrinth auf eine Übersichtskarte, anschließend allerdings nur noch einen schwarzen Hintergrund, trotzdem muss man den Ausgang finden

2.3 Labyrinth im Klassenraum

Ein Schüler erhält eine Augenbinde, im Klassenraum wird ein kleiner Parcours aufgebaut. Die anderen Schüler müssen durch Anweisungen aus dem untenstehenden Befehlssatz versuchen, den Schüler sicher und ohne anzuecken durch den Klassenraum zu geleiten.

- Schritt vor
- Drehe nach rechts (Drehung um 90 Grad nach rechts)
- Drehe nach links (Drehung um 90 Grad nach links)

Dieses Experiment kann sehr gut als Einstieg genutzt werden, um die Unterschiede in der “Genauigkeit” von realer Welt und Computer zu verdeutlichen. Das Programm wird hier beim zweiten Schüler vermutlich aufgrund anderer Schrittweiten o. ä. nur noch ansatzweise funktionieren. Anschließend kann das Experiment “Programmieren mit Stift und Papier” gewissermaßen als Kontrast verwendet werden.

2.4 Labyrinth mit Folie und Papier

Es ist teilweise in den Klassenzimmern nicht möglich die Tische zu verrücken um ein Labyrinth zu bauen. Der Grund dafür ist entweder, dass die Tische am Boden fest montiert sind oder Teilweise ist auch die Gruppengröße für das Klassenraumprojekt nicht angemessen. In diesem Fall gibt es die Alternative für eine Übung in Kleingruppen am Arbeitsplatz. Man dasselbe Experiment auch mit einem a gedruckten Labyrinth in kleinen Gruppen durchgeführt werde. Man bildet kleine Gruppen bis zu einer Gruppengröße von bestehnfalls 3 bis maximal 5 Personen. Jede Gruppe erhält eine Folie, ein Papier mit einem Raster und ein Papier, auf dem das Labyrinth abgedruckt ist. Eine Person in der Gruppe darf das Labyrinth sehen und der

2 Informatik mit Stift und Papier

Zeichnenden Person Anweisungen geben. Ein Gruppenmitglied ist der Zeichner. Dieses Gruppenmitglied hat die Aufgabe auf der Folie, die auf dem Rasterpapier liegt, die Anweisungen der Person aufzuzeichnen, die das Labyrinth lösen muss. Eine dritte Person schreibt die Befehle untereinander auf, die die Anweisende Person gibt. So entsteht ein Programm.

Der Befehlssatz besteht hier aus:

- Ein Kästchen vor
- Drehe links (um 90 Grad)
- Drehe rechts (um 90 Grad)

Im Anschluss an diese Übung kann dasselbe Labyrinth mit Scratch gelöst werden. Hier kann im Anlehnung an das in der Gruppe erstellte Programm eine Maus durch das Labyrinth zum Käse geführt werden. Dieses Projekt vereint das Projekt "Maus zum Käse" und "Labyrinth". Für Schüler, die bereits sehr schnell das erste Programm in Scratch fertiggestellt haben, liegt im Anhang ein anspruchsvoller Labyrinth vor, um das Erlernte zu vertiefen.

Die Parallelen zur allgemeinen Programmierung können mit diesem Projekt sehr schön bildhaft dargestellt werden. Die Kinder sollen verstehen, wie der Programmierer mit dem Computer und der Entwicklungsumgebung arbeitet. Dafür werden zu Beginn drei Rollen verteilt:

- *Programmierer* : Anweisende Person, dieser erhält das Labyrinth auf Folie. Der Programmierer kennt damit das zu lösende Problem, kann es aber nur mithilfe des Computers lösen
- *Computer*: Der "Computer" erhält ein leeres Blatt Papier mit einem vorgezeichneten Raster. Auf dieses wird Schritt für Schritt der Weg eingezeichnet, den der Programmierer vorsagt. Der Computer hat also keine Kenntnis über das Problem und befolgt lediglich die Anweisungen des Programmierers.
- *Entwicklungsumgebung/Sekretär*: Damit der Programmierer das Problem lösen kann, ohne bei jedem Durchlauf die Befehle neu zu überlegen, werden alle Anweisungen auf einem leeren Blatt Papier festgehalten. Wenn das Problem dann einmal fertig gelöst ist, kann mit diesen Anweisungen das Labyrinth immer wieder durchlaufen werden.

2.4.1 Erweiterungen

Wenn das Problem mit einfachen Anweisungen gelöst ist, kann als neuer Befehl die “Wiederhole”-Struktur eingeführt werden. Mit dem Befehl

Wiederhole (N Mal)

wird der darauffolgenden Befehl entsprechend wiederholt ausgeführt. Die Schülerinnen und Schüler können dann mit diesem neuen Befehl ihr Programm entsprechend vereinfachen.

Mit leichten Abstrichen lässt sich diese Vorgehensweise, wenn nötig, auch auf einer Tafel durchführen.

2.4.2 Überleitung

Nach erfolgreichem Abschluss des Projekts kann die Arbeit am PC beginnen. Empfohlenes Projekt für den Anschluss ist wahlweise “Maus zum Käse” vor allem für jüngere Schülerinnen und Schüler aus der Grundschule oder auch “Labyrinth”.

3 Projekte zum Einstieg

In diesem Kapitel werden Projekte für den einfachen Einstieg in Scratch und damit die Programmierung im Allgemeinen behandelt. Sie sind in der Regel, je nach Gruppe, im Rahmen einer Unterrichtsstunde umsetzbar. Die meisten Einstiegsprojekte sind so konzipiert, dass sie eine relativ spezifische Fähigkeit vermitteln sollen, weshalb sie teilweise auch später noch hilfreich sein können.

3.1 Bananen hergezaubert

Erforderliche Kompetenzen:

Bewegung

Gewonnene Kompetenzen:

Klone, Senden und Empfangen, Zufall

3.1.1 Beschreibung

Der Zauberer oben links wird mehrmals angeklickt, woraufhin er aus einer Banane sehr, sehr viele herzaubert. Dieses Projekt bietet eine Möglichkeit Zufall und Botschaften (in diesem Fall als “Zauber”) anschaulich zu machen

3.1.2 Durchführung

Phase 1: Planung

In der Planungsphase werden mit den Schülern die folgenden Dinge besprochen:

- Wie klont man eine Figur? Wie steuert man den Klon? Was bedeuten die Klonbefehle unter “Steuerung”?
- Wie bewegt sich der Klon? Wie wählt er sein Ziel aus?
- Was ist der Trigger für das Klonen? Welche Rolle spielt dabei der Zauberer?

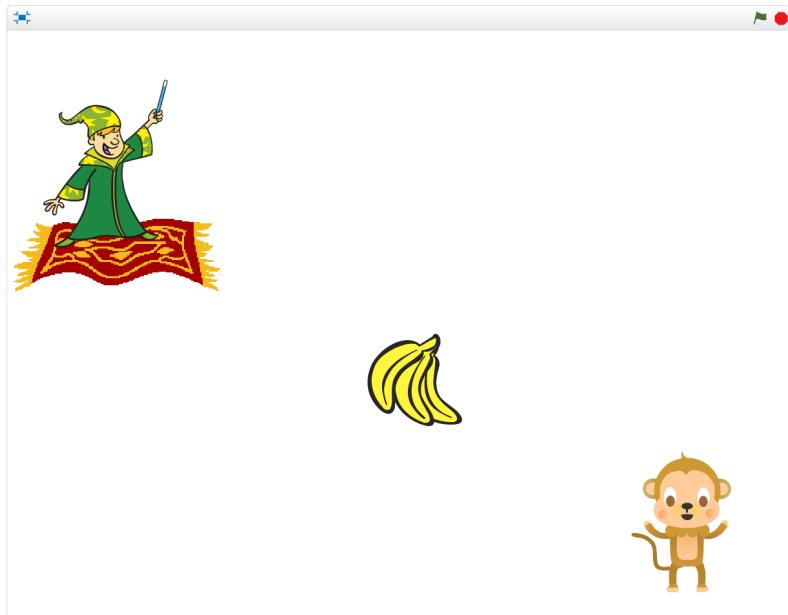


Abbildung 3.1: Bananen hergezaubert

Phase 2: Vorbereitungen

In der Vorbereitungsphase suchen sich die Schüler die Figuren aus: Zauberer, Zauberteppich, Banane und Affe. Selbstverständlich können auch diverse andere Kombinationen aus den verfügbaren Figuren der Scratch-Spritzbibliothek verwendet oder Schüler können sich selbst etwas malen.

Phase 3: Programmierung

1. Die Banane klont sich selbst
2. Der Klon gleitet an eine zufällig gewählte Spielfeldposition
3. Die Banane klont sich selbst genau dann, wenn der Zauberer angeklickt wird
4. Optional können verschiedene Geräusche und Farbeffekte an den passenden Stellen eingebaut werden

Detaillierte Programmbeschreibung

Die Musterlösung funktioniert folgendermaßen: Bananen erzeugt einen Klon von sich selbst, wenn sie die Nachricht “Teilt euch!” erhält. Im “When I start as a clone”-Block gleitet Bananen in zufällig bestimmt 1 bis 10 Sekunden zu einer ebenfalls zufällig gewählten Spielfeldposition. Zauberer versendet die Nachricht “Teilt euch!” in dem Moment, das sie angeklickt wird.

Phase 4: Testen und vorstellen

In der Vorstellungsphase kann ein beliebiger Schüler sein Spiel vorstellen, falls er es möchte. Dabei kann man explizit auf die einzelnen Teilprobleme hinweisen, welche der Schüler gelöst hat. Aber man kann die anderen auch fragen, was er verbessern könnte, wenn er an diesem Projekt nächste Stunde noch arbeiten will.

3.1.3 Erweiterungen

Es wäre denkbar, dass der Affe mit den Pfeiltasten bewegt werden kann und dabei die berührten Bananen gegessen werden.

3.2 Maus zum Käse

Erforderliche Kompetenzen:

keine

Gewonnene Kompetenzen:

BEWEGUNGEN

3.2.1 Beschreibung

Dieses Projekt kann relativ gut skaliert werden, von einem recht einfachen bis zu einem recht umfangreichen Schwierigkeitsgrad. Primäres Ziel ist die Einführung von *Bewegungen* und erste Erfahrung mit den Funktionen aus der Kategorie *Fühlen*. Optional kann auch mit den grundlegenden Funktionen aus der Kategorie *Steuerung* gearbeitet werden.

Das Projekt entstand als Anlehnung an die manuell durchgeführten Labyrinth Versuche. Als Spielfiguren kommen eine Maus und ein Stück Käse zum Einsatz. Die Maus befindet sich auf einer Insel, die

3 Projekte zum Einstieg



Abbildung 3.2: Maus zum Käse

über einen Holzsteg mit einer anderen Insel verbunden ist. Die Schülerinnen und Schüler sollen nun versuchen, mithilfe von hintereinander ausgeführten Bewegungen die Maus bis zum Käse zu bringen. Im ersten Anlauf können die einzelnen Befehle dabei manuell angeklickt werden, wenn alles funktioniert, wird alles zu einem Programm zusammengefasst.

3.2.2 Erweiterung

Schnelle Schüler können versuchen, die Maus auf Tastendruck reagieren zu lassen und sie somit zu steuern. Zudem können mehrere "Level" gespielt werden, indem weitere Hintergrundbilder angelegt werden.

3.3 Labyrinth

Erforderliche Kompetenzen:

keine

Gewonnene Kompetenzen:

Scratch-Umgebung kennen lernen

Ereignisse {Startbedingung (Wenn angeklickt)}

Bewegung {Koordinaten (gehe, gehe in zu), Winkel (drehe dich)}

Steuerung {Wiederholungen (wiederhole)}

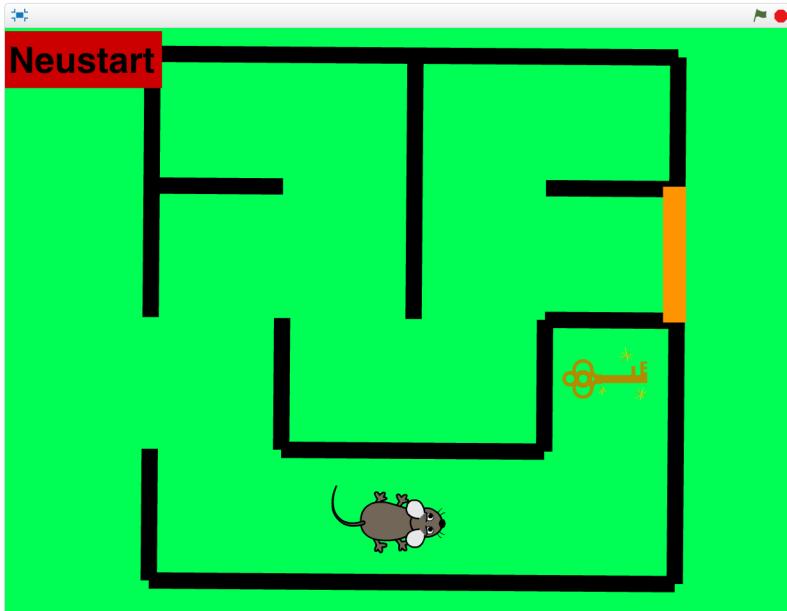


Abbildung 3.3: Labyrinth

3.3.1 Beschreibung

Mit dem Start des Programms soll die Maus ohne weitere Eingabe das Labyrinth bis zum Ziel durchlaufen.

Bei der Schlüsselvariante muss die Maus vorher auf den Schlüssel navigieren, damit sich das Tor öffnet.

3.3.2 Durchführung

Phase 1: Planung

In der Planungsphase werden mit den Schülern die folgenden Punkte besprochen:

- Wie und wodurch wird ein Programmteil gestartet? Welchen Einfluss hat dies auf andere Programmteile?
- Wie Programmiere ich eine spezielle Figur und nicht versehentlich eine andere?
- Was sind Koordinaten und Winkel? Wie kann ich diese ablesen? (Dieser Punkt ist optional, gerade in der Grundschule kann das Verständnis von Gradzahlen/Winkeln und Koordinaten auf einen späteren Zeitpunkt mit einem spezifischen Projekt verschoben werden.)

Phase 2: Vorbereitungen

Hierbei ist eine Einarbeitung vorgesehen: Wo bearbeitet man die Figuren, Info der Figuren, Koordinaten und Winkel ablesen, Wo sind die Skripte gehe, drehe dich, ... zu finden.

Phase 3: Programmierung

Die verwendeten Befehle sind:

- wiederhole ... mal
- gehe ... er-Schritt
- drehe dich um ... Grad
- Diese Schritte müssen so oft wiederholt werden, bis die Maus das Ziel erreicht hat.

Phase 4: Testen und vorstellen

In der Vorstellungsphase können die Schüler ihre Lösung vorstellen, wodurch die unterschiedlichen Arten der Lösung erläutert werden können. Dabei sollte jedoch der spielerische Spaß im Vordergrund stehen und nicht abschreckend wirken.

3.3.3 Erweiterungen

Als Erweiterung zum Labyrinth steht noch Labyrinth mit Schlüssel zur Verfügung, in diesem Können sie sich weiter mit der Problematik auseinander setzen und eine andere Variation der Programmiermöglichkeiten aneignen.

3.4 Buchstaben

Erforderliche Kompetenzen: Scratch-Umgebung kennen

Gewonnene Kompetenzen:

Figuren auswählen/bearbeiten, Ereignisse, Bewegung (gehe zu, drehe dich). Steuerung {Wiederholungen (wiederhole)}

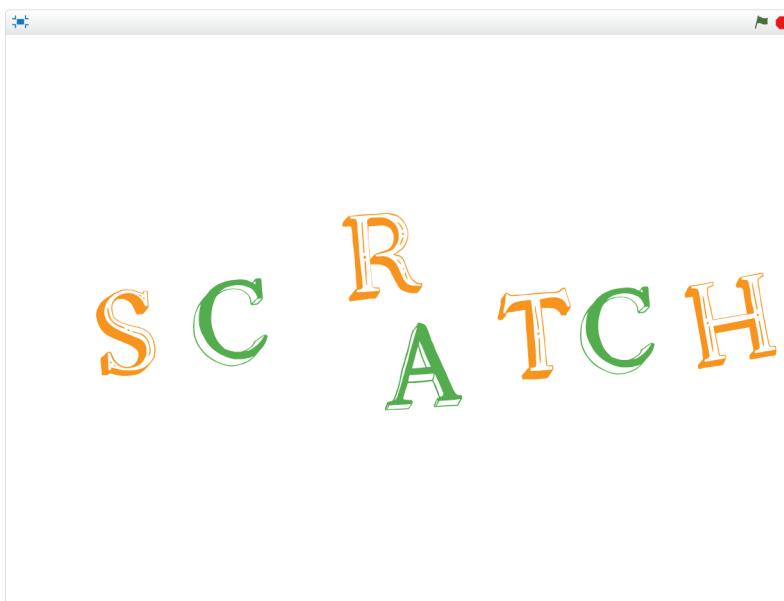


Abbildung 3.4: Buchstaben

3.4.1 Beschreibung

Es soll ein Programm erstellt werden, in welchen beliebige Figuren (beispielsweise Buchstaben) wackelnd, drehend, größer oder kleiner werdend, ... zu einer sinnvollen Endposition (einem Wort) gleiten.

3.4.2 Durchführung

In der Planungsphase werden mit den Schülern die folgenden Punkte besprochen:

Phase 1: Planung

- Welche Möglichkeiten stehen mit bei den Figuren zur Verfügung? Wie wähle ich diese aus?
- Wie Programmiere ich eine spezielle Figur und nicht versehentlich eine andere?
- Was sind Koordinaten und Winkel? Wie kann ich diese ablesen?

Phase 2: Vorbereitungen

In der Vorbereitungsphase suchen sich die Schüler die Figuren aus. Diese können sie auch noch beliebig bearbeiten und ihrer Kreativität freien Lauf lassen indem sie z.B. auch weitere Figuren selber malen.

Phase 3: Programmierung

Die nötigen Schritte sind:

- Beim Start Figuren zum Chaospunkt Position springen lassen.
- Figuren drehend, ... zum geordneten Position bewegen.

Phase 4: Testen und vorstellen

In der Vorstellungsphase können die Schüler ihre eigenständig Programmierten Spiele vorstellen, um den anderen Ihre Kreativität zu zeigen und den anderen dadurch weitere Anregungen für die nächste Stunde zu geben.

3.4.3 Erweiterungen

Als Erweiterung zu Buchstaben besteht die Möglichkeit den Schülern das Projekt mit Musik hinterlegen zu lassen, oder die Figuren mit kleinen Änderungen mehrmals einfügen, um diese dann geschickt wechselnd eine Bewegung simulieren zu lassen oder einfach nur die Farbe ändernd.

3.5 Farben

Erforderliche Kompetenzen: keine

Erworbene Kompetenzen: Fühlen, Bedingungen

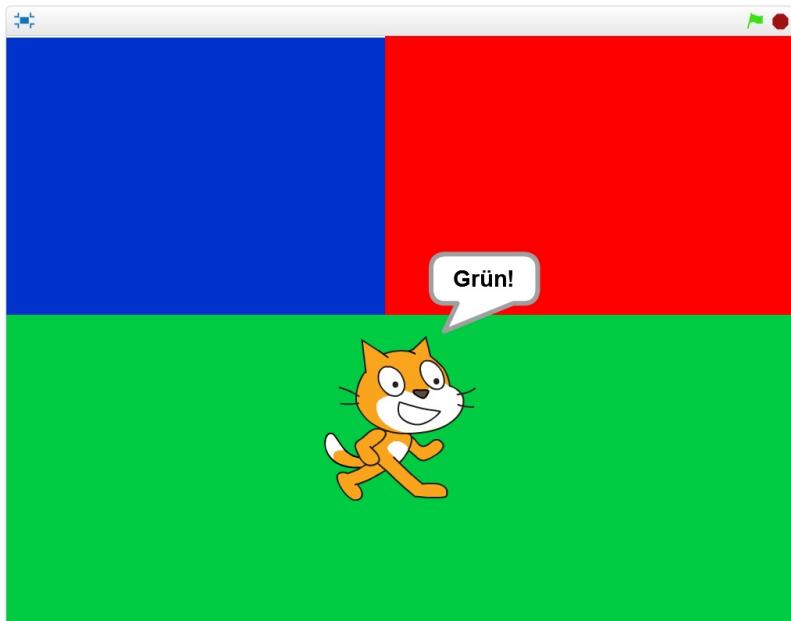


Abbildung 3.5: Farben

3.5.1 Beschreibung

Bei diesem Projekt geht es darum, die verschiedenen Funktionen aus der Klasse *Fühlen* zu testen. Grundsätzlich bekommen die Schülerinnen und Schüler einen zwei- oder mehrfarbigen Hintergrund. Auf Basis der Hintergrundfarbe soll sich das Aussehen der Figur verändern. Dabei kann die Figur zwischen den einzelnen Bereichen verschoben werden.

In einer Endlosschleife soll immer die aktuelle Hintergrundfarbe abgefragt werden. Dies geschieht über die entsprechende Funktion in der Befehlsgruppe “Fühlen”. Über eine Falls-Bedingung kann abgefragt werden, ob die Hintergrundfarbe der gewählten entspricht. Ist dies der Fall, so soll eine bestimmte Aktion ausgeführt werden. Obligatorisch wäre eine Ausgabe in einer Sprechblase, hier kann den Schülern aber grundsätzlich Freiraum für eigene Experimente gelassen werden.

3.5.2 Vorbereitung

Sehr wichtig für die Durchführung dieses Projekts ist die Einführung von Kontrollstrukturen. Das Programm später wird auf die “Falls … Dann” Abfrage zurückgreifen, diese sollte also in der Vorbereitung vorab angesprochen werden.

Der entsprechende Block findet sich in der Kategorie “Steuerung”. Als Argument muss in den Block noch ein weiterer Befehl eingesetzt werden. Im Falle dieses Projekts handelt es sich dabei um einen Block aus der Kategorie “Fühlen”. Hier stehen zahlreiche Funktionen bereit, so wie die Kollisionserkennung mit anderen Objekten, dem Mauszeiger oder dem Rand auch das Abfragen der Hintergrundfarbe, was wir uns in diesem Projekt zunutze machen wollen.

In der Vorbereitung sollte den Schülerinnen und Schülern nun vermittelt werden, dass mithilfe dieser Kontrollstrukturen eine Abfrage der Hintergrundfarbe mit anschließendem Ausführen einer Aktion möglich ist.

3.5.3 Programmierung

Die erste Aufgabe ist es, die Hintergrundfarbe als Text auf dem Bildschirm auszugeben. Dazu kann der entsprechende Befehl aus der Klasse “Aussehen” verwendet werden.

In der Vorlage steht ein Hintergrund mit verschiedenen Farben bereit.

Die Programmabfolge soll nun so aussehen, dass die Figur mit der Maus auf eine Farbe gezogen wird. Sobald die Figur angeklickt wird, soll die aktuelle Farbe ausgegeben werden.

Zur Lösung ist die Verwendung einer Kontrollstruktur pro Farbe notwenig, gekoppelt an die jeweilige Ausgabe der Farbe.

3.5.4 Erweiterungen

Für dieses Projekt sind einige Änderungen denkbar: Schülerinnen und Schüler zeichnen den Hintergrund selbst, statt die Vorlage zu verwenden.

Nach Fühlen einer anderen Farbe ist eine Kostümänderung anstelle einer Ausgabe in einer Sprechblase denkbar.

3.6 Tanzstunde

Erforderliche Kompetenzen: Scratch-Umgebung kennen

Gewonnene Kompetenzen: Startbedingungen, Nachrichten, Wiederholungen, Kostüme, Klänge/Instrumente

3.6.1 Beschreibung

Wird die oberste Figur, der Tanzlehrer, angeklickt, sendet er eine Nachricht an alle Tänzer, das sind die anderen drei Figuren unter ihm. Beim empfangen dieser Nachricht beginnen die Tänzer, zu tanzen, und Musik zu machen. Die Tänzer tanzen, indem sie ihre Kostüme wechseln.

3.6.2 Durchführung

Phase 1: Planung

In der Planungsphase werden mit den Schülern die folgenden Punkte besprochen:

- Wie wird eine Nachricht (“Message”) versandt und empfangen
- Wie werden einer Figur mehrere Kostüme zugewiesen und diese im Programm geändert

3 Projekte zum Einstieg



Abbildung 3.6: Tanzstunde

- Welche Möglichkeiten gibt es, Klänge zu erzeugen

Phase 2: Vorbereitung

In der Vorbereitungsphase erstellen die Schüler die nötigen Figuren. Diese können auch noch beliebig bearbeitet und kreativ verändert werden, in z.Bsp.: weitere Kostüme hinzugefügt oder Figuren selber gemalt werden.

Eine oder mehrere Bühnenbilder sind auch denkbar, welche dann zusätzlich mit der Musik variieren können.

Phase 3: Programmierung

- sende / empfange Message
- nächstes Kostüm
- warte ... Sek.

- diese letzten beiden Schritte werden fortlaufend wiederholt
- zwischen den Wiederholungen kann noch mit Klang gearbeitet werden

Phase 4: Testen und vorstellen

In der Vorstellungsphase können die Schüler ihre eigenständig individuellen Kreationen vorstellen. Dabei haben die Schüler die Möglichkeit sich gegenseitig zu inspirieren.

3.6.3 Erweiterungen

Als Erweiterung dazu können die Schüler noch unter Daten eine Variable hinzufügen, die die Anzahl der Klicks vom Tänzer zählt.

3.7 Zeichnen

Erforderliche Kompetenzen: Bewegen

Gewonne Kompetenzen: Zeichnen, Bedingungen

Ziel des Projekts “Zeichnen” ist das tiefere Verständnis des Koordinatensystems sowie der Drehwinkel. Außerdem werden Möglichkeiten aufgezeigt, wie Formen auf den Bildschirm gezeichnet werden können.

3.7.1 Vorbereitung

Das Projekt sollte mit einer Einführung und einem anschaulichen Beispiel an der Tafel begonnen werden. Dabei wird eine Schülerin oder ein Schüler zur Tafel gebeten, um ein Quadrat zu zeichnen.

Dabei darf die Person an der Tafel nur den Anweisungen seiner Mitschülerinnen und Mitschüler folgen. Aufgabe ist es, mittels der Befehle

- Vorwärts
- Drehe dich (wie weit?)

verschiedene Formen zu zeichnen. Für den Beginn eignen sich Quadrate, Dreiecke, Sechsecke und später auch ein Kreis.

3 Projekte zum Einstieg

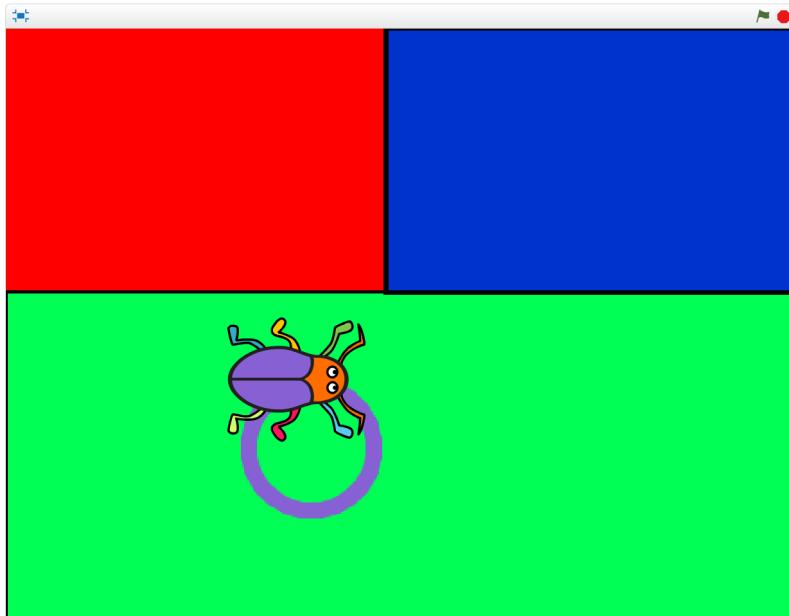


Abbildung 3.7: Zeichnen

Ziel des Projekts ist es, dass die Schülerinnen und Schüler mit den verschiedenen Drehwinkeln vertraut gemacht werden. Mögliche Anweisungen könnten dann etwa so aussehen:

- Quadrat: "Vorwärts, Drehen (90 Grad), Vorwärts, Drehen (90 Grad), Vorwärts, Drehen (90 Grad), Vorwärts, Drehen (90 Grad)"
- Sechseck: Vorwärts, Drehen (60 Grad), Vorwärts, ...
- Kreis: ein Stück vorwärts, Drehen, ein Stück vorwärts, ...

Diese Anweisungen können natürlich auch hier wie bei "Informatik mit Stift und Papier" durch entsprechende *Wiederhole-Anweisungen* stark abgekürzt werden.

Welche Formen genau eingeführt werden, bleibt generell dem Kursbetreuer überlassen. Der Kreis ist als schwierigste Form anzusehen und kann auch optional als "Bonusaufgabe" beim nun folgenden Programmieren für besonders schnelle Schüler herangezogen werden.

3.7.2 Zum Projekt

Nach der Einführung sollten alle Schülerinnen und Schüler die Grundidee des Projekts erfasst haben. Ziel ist es nun, die zuvor an der Tafel gezeichneten Formen im Programm umzusetzen. Die Tafel mit dem entsprechenden Anschrieb kann dabei je nach Stärke der Gruppe verdeckt/zugeklappt werden oder aber auch offen bleiben und als Vorlage dienen.

Die Aufgabe am PC ist, mit der Programmierumgebung mindestens drei verschiedene Formen zu zeichnen.

Blöcke in der Gruppe “Zeichnen”

In der Kategorie “Zeichnen” sind in der Programmierung alle wichtigen Befehle zusammengefasst, die den sog. Malstift steuern. Dieser virtuelle Stift greift im Mittelpunkt der jeweiligen Figur an und kann abgesetzt und aufgenommen werden. Wenn der Stift abgesetzt ist, zieht die Figur bei ihren Bewegungen eine Linie hinter sich her. Diese Linie soll hier genutzt werden, um die Formen zu zeichnen.

Blöcke in den Gruppen “Steuerung” und “Bewegung”

Mit den Blöcken “gehe (10)er Schritt” und “drehe dich um (15) Grad” werden die Bewegungen realisiert. Zu Beginn können mehrere dieser Blöcke aneinandergereiht werden. Wenn die Schüler schnell zurecht kommen, kann das Programm gekürzt werden, indem mehrere dieser Blöcke über die Steuerungsblöcke “wiederhole (10) mal” zusammengefasst werden.

Die Geschwindigkeit der Bewegung kann über die Variation von Wiederholungen und Schritten eingestellt werden.

Beispielsweise führen folgende Möglichkeiten zu der gleichen Strecke, bei der ersten Möglichkeit bewegt sich die Figur aber schneller:

- wiederhole 10 Mal: gehe 10er Schritt
- wiederhole 100 Mal: gehe 1er Schritt

3.7.3 Erweiterungen

- *Farben*: eine Schülerin, welche drei oder mehr geometrische Figuren programmiert hat, könnte nun ein Hintergrundbild zeichnen, wo es drei oder mehr farbige Bereiche gibt. Der Charakter

3 Projekte zum Einstieg

soll dann je nachdem auf welcher Farbe er sich gerade befindet, eine bestimmte geometrische Figur zeichnen. Hier könnte der Befehl “Wenn ich angeklickt werde” zum Einsatz kommen. Dies eignet sich zur Einführung von Bedingungen.

4 Spieleprogrammierung

In diesem Kapitel geht es um einige Projekte, bei der kleine Spiele entwickelt werden. Es handelt sich dabei um Spiele für einen Spieler gegen den Computer als auch für zwei Spieler, die per Tastatur etc. gegeneinander antreten können. Generell weisen diese Projekte einen höheren Schwierigkeitsgrad auf, deshalb sollte beachtet werden, dass zunächst alle nötigen Kompetenzen aus den vorigen Kapiteln erworben wurden. Alternativ kann die Vorlage auch angepasst werden, sodass bestimmte Elemente nicht von den Schülern erstellt werden müssen.

Neben der hier beschriebenen Programmierung gibt es besonders bei den Spielen eine Vielzahl an Erweiterungsmöglichkeiten, die teilweise hier aufgeführt werden, teilweise aber auch der Vorstellungskraft und Experimentierfreudigkeit von Schülerinnen und Schülern überlassen sind.

4.1 Finde das Tier

Erforderliche Kompetenzen: Steuerung(Bedingungen, Wiederholungen); Bewegung(Gehe zu); Variablen; Klänge; Nachrichten

Gewonnene Kompetenzen: Ereignis: Wenn ich angeklickt werde; Zufallszahlen

4.1.1 Beschreibung

Finde das Tier ist ein Reaktionsspiel, bei dem der Spieler innerhalb einer bestimmten Zeit eine bestimmte, durch Zufall ausgewählte Figur anklicken muss, um einen Punkt zu erlangen und einen Punkt verliert, wenn er die falsche Figur anklickt. Zusätzlich bewegen sich die Figuren zufällig im Spielfeld, um das Spiel zu erschweren.

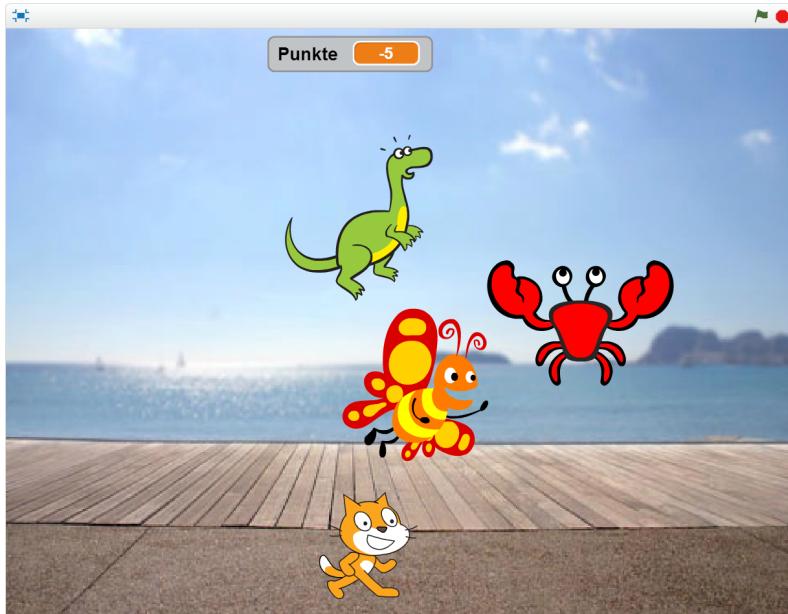


Abbildung 4.1: Finde das Tier

4.1.2 Durchführung

Phase 1: Planung

Zunächst soll das Projekt den AG-Teilnehmern vorgestellt und erläutert werden. Anschließend werden unbekannte Blöcke wie das Ereignis “Wenn ich angeklickt werde” und die Zufallszahlen erklärt. Dies kann mithilfe von kleinen Beispielprogrammen geschehen. Insbesondere soll gezeigt werden, wie sich die Figuren im Raum zufällig bewegen können mithilfe von “gehe zu.” und Zufallszahlen.

Phase 2: Vorbereitung

Zunächst müssen die benötigten Figuren hinzugefügt werden. Die Auswahl der Figuren ist beliebig.

Phase 3: Programmierung

Den Schiedsrichter programmieren: Zuallererst soll der Schiedsrichter (in der Vorlage die Katze) also die Figur, die das zu fangende Tier bestimmt, programmiert werden. Dazu soll zunächst die Variable “Gesuchtes Tier” angelegt werden.

Das Skript des Schiedsrichters soll nun aus einer “Endlosschleife” (wiederhole fortlaufend) bestehen, in der in jedem Durchgang die Variable “Gesuchtes Tier” auf einen zufälligen Wert zwischen eins und drei gesetzt wird.

Anschließend soll nun der Schiedsrichter dem Spieler noch mitteilen welches Tier gesucht wird. Dies kann mit einer “falls ... dann” Bedingung realisiert werden. (Wichtig: Hier ist der Block “sage xxx für 1 Sekunde” nötig, damit die Schleife nicht zu schnell läuft.)

Schiedsrichter mit Spielfiguren verbinden: Nun sollen drei Spielfiguren erstellt werden, die die zu fangenden Tiere darstellen. Bei jedem Durchlauf der Schleife des Schiedsrichters, sollen die Tieren eine zufällige Position im Raum einnehmen. Dazu soll der Schiedsrichter eine Nachricht (z.Bsp. “Position”) an alle Tiere aussenden, bei deren Empfang eine zufällige x-und y-Position eingenommen wird. (Am besten geschieht der Positionswechsel vor dem Setzen der Variable “Gesuchtes Tier”.)

Den Punktstand programmieren: Als nächstes muss die Variable “Punkte” angelegt werden. Sie soll zu Beginn jedes Spiels auf null gesetzt werden. Die Änderung des Punktstandes erfolgt im Skript der Spielfiguren, nicht des Schiedsrichters. Hier wird nun das neue Ereignis “Wenn ich angeklickt werde” verwendet, um den Klick auf dieses Tier zu registrieren. Schließlich soll überprüft werden, ob das richtige Tier angeklickt worden ist. Ist dies der Fall so soll der Punkttestand um eins erhöht werden, wenn nicht, soll er um eins verringert werden.

Das Spielende programmieren: Zu guter Letzt soll nun einprogrammiert werden, dass das Spiel endet sobald eine bestimmte Punktzahl erreicht worden ist. Dies geschieht innerhalb der “wiederhole fortlaufend”-Anweisung des Schiedsrichters, wo die Variable “Punkte” abgefragt wird. Falls das Spiel beendet werden soll, teilt der Schiedsrichter dem Spieler mit, dass er gewonnen hat und stoppt alle Skripte.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, können alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben! Es kann anschließend eine weitere Programmierphase angestoßen werden, bei der jedem die Chance gegeben wird, die vorhandenen Mängel noch zu beheben.

4.1.3 Erweiterungen

4.2 Klänge

Erforderliche Kompetenzen: Gewonnene Kompetenzen: KLÄNGE

4.2.1 Beschreibung

Ziel dieses Projekts ist die Einführung in die Kategorie "Klänge". Ziel ist es, eine Bühne aus Figur und einigen Musikinstrumenten aufzubauen. Passend zu einer Hintergrundmusik kann die Figur durch Tastendrücke, Klicks auf den Bildschirm etc. dazu veranlasst, die verschiedenen Musikinstrumente zu spielen. Das Projekt kann sehr gut ausgebaut werden und eignet sich vor allem für kreativ veranlagte Schülerinnen und Schüler, da das Maß an Gestaltungsmöglichkeiten sehr groß ist.

Unbedingt erforderlich zur Durchführung des Projekts ist es, dass PCs mit Lautsprechern oder Kopfhörern vorhanden sind. Der Projektumfang beträgt etwa eine Doppelstunde an der Grundschule.

4.2.2 Durchführung

Phase 1: Planung

Zum Beginn sollte das fertige Projekt gezeigt und vorgestellt werden. Unter Umständen kann es Sinn machen, auch einzelne Schülerinnen



Abbildung 4.2: Klänge

und Schüler die fertige Lösung am Lehrerrechner ausprobieren zu lassen. Anschließend sollten die wichtigsten Befehle erwähnt werden. Dazu gehören

- **Wie gelange ich zur Auswahl der Klänge?** Die geschieht durch die Auswahl einer Figur und Wahl der Registerkarte “Klänge”. Anschließend können zur aktuellen Figur neue Klänge hinzugefügt werden.
- **Wie spiele ich im Programm einen Klang ab?** Das Abspielen erfolgt mit dem Block “spiele Klang (ganz)” in der Kategorie “Klang” der vorher ausgewählten Figur.

Sofern es zuvor noch nicht angesprochen wurde, sollten die Schülerinnen und Schüler zuvor zusätzlich noch damit vertraut gemacht werden, wie auf Tastendrücke oder Mausaktionen hin mit Blöcken der Kategorie Ereignisse auf Nutzereingaben reagiert werden kann.

Phase 2: Vorbereitung

. Die Schülerinnen und Schüler sollten nun ausreichend Zeit zur Verfügung gestellt bekommen, um sich einen Hintergrund sowie Figuren und Instrumente auszusuchen bzw. selbst zu malen. Diese Phase darf ruhig mehr Zeit in Anspruch nehmen. Es ist darauf zu achten, dass viele individuelle Fragen zum Zeichnen, Einbinden von Figuren und vor allem auch dem Einbinden von Klängen auftreten werden.

Zum Ende dieser Phase sollte jeder Teilnehmer über ein Bühnenbild mit einer oder mehreren Figuren verfügen. Darüber hinaus sollten in der “Hauptfigur” ein langer Ton für das Abspielen im Hintergrund und einige kurze Töne (z. B. Schlagzeug) ausgewählt sein.

Phase 3: Programmierung

Die Programmierung kann bei diesem Projekt beliebig komplex werden. Die Reihenfolge der verschiedenen Funktionen kann dabei sehr stark von der Lernmotivation und den Interessen der Schülerinnen und Schülern abhängen. Generell empfiehlt es sich, bei Anmerkungen folgenden Ablauf im Hinterkopf zu haben:

- Abspielen eines Klanges im Hintergrund
- Abspielen weiterer Klänge basierend auf Interaktion (Klicken auf Figur, Drücken von Tasten, Berühren anderer Figuren, ...)
- Einbau von Animationen, z. B. Tanzbewegungen

Es sollte bei diesem Projekt stark auf die individuellen Ideen der Schüler geachtet werden.

Phase 4: Testen und vorstellen

Am Ende sollten einige Minuten für die Vorstellung des Projekts eingeplant werden. Nach Möglichkeit sollte jeder sein Projekt kurz selbst präsentieren (und steuern) dürfen, sofern der Lehrerrechner entsprechend mit Lautsprechern ausgestattet ist.

4.2.3 Erweiterungen

Für das Projekt ist eine ganze Reihe von Erweiterungen denkbar. So kann die Bühne etwa mit Vorhängen versehen werden, die sich

zu Beginn öffnen und am Ende der Vorstellung wieder geschlossen werden. Dies kann beispielsweise durch zwei "Figuren" geschehen, die sich auf Befehl vor den Hintergrund und die anderen Figuren schieben.

4.3 Was soll ich anziehen?

Erforderliche Kompetenzen:

**Was soll ich nur anziehen ?
Klicke die Dinge an, die ich anziehen soll !**



Abbildung 4.3: Was soll ich anziehen?

4.3.1 Beschreibung

Dies ist ein Spaßprojekt: Auf der Bühne gibt es eine Figur und eine Menge von Kleidungsstücken. Die Figur soll vom Spieler angezogen werden, indem dieser auf die verschiedenen Kleidungsstücke klickt.

4.3.2 Durchführung

Phase 1: Planung

- Präsentation eines komplett fertigen Projekts
- Überlegungen und Ideensammlung für Hintergrund, Figur, etc.
- Erläuterung des “Gehe-zu”-Befehls
- Wie müssen sich die Figur einerseits und die Kleidungsstücke andererseits verhalten?

Phase 2: Objekte und Hintergrund zeichnen

Zeichnen von Hintergründen, Figuren etc. Man kann, wie es im Beispiel gezeigt ist, auch den Hintergrund weiß lassen, aber besonders spaßig ist es nicht.

Phase 3: Programmieren

1. Das Erscheinen von Figur und Kleidungsstücken
2. Die korrekte Bewegung des jeweiligen Kleidungsstücks zur Figur auf Klick

Phase 4: Testen und vorstellen

Wie immer.

4.4 Pong

Erforderliche Kompetenzen: BEWEGUNGEN, FÜHLEN, BEDINGUNGEN, VARIABLEN

Erworбene Kompetenzen:

4.4.1 Beschreibung

Dieses Projekt gehört zu den eher fortgeschrittenen Projekten. Wie auch beim Projekt *Pacman* geht es um ein komplettes Spiel.

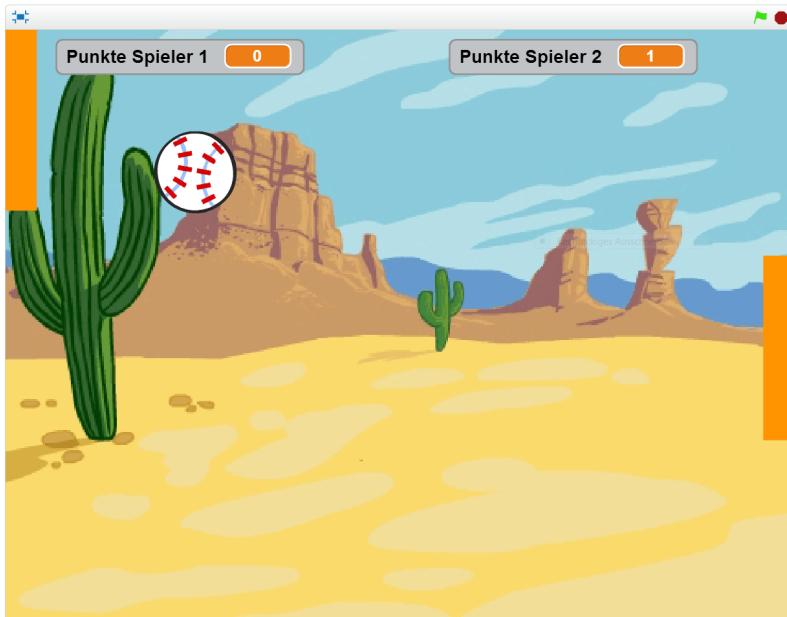


Abbildung 4.4: Pong

4.4.2 Durchführung

Phase 1: Planung

Während der Planungsphase sollten mit den Schülerinnen und Schülern die folgenden Schritte vollzuogen werden:

- Überlegungen und Ideensammlung für Hintergrund, Figuren, etc.
- Überlegungen zur Programmierung und der nötigen Aufgaben:
 - Bewegungen der Schläger (hoch und herunter)
 - Bewegung des Balls (bei jüngeren Schülern kann diese später vorgegeben werden, einige Überlegungen sollten in der Planungsphase aber dennoch angestellt werden)
 - Zählen der Punktzahl
 - Verhalten des Balls an Rändern und am Schläger

- Sammeln aller Ideen und Erstellung der TODO Liste

Anschließend kann mit der Umsetzung des Projekts begonnen werden.

Phase 2: Vorbereitungen

Zunächst sollten die Figuren und der Hintergrund gezeichnet werden. Für die Figuren bieten sich einfache Striche an, die auf und ab bewegt werden können, die Gestaltung des Hintergrunds ist den Schülerinnen und Schülern überlassen.

Phase 3: Programmierung

Im Anschluss an die vorbereitetenden Maßnahmen kann mit der Programmierung des Spiels begonnen werden. Als Vorschlag sei die folgende Reihenfolge genannt, in der Schritt für Schritt leicht validierbare Teile des Spiels entstehen:

1. *Steuerung der Figuren:* Beide Spielfiguren müssen durch entsprechende Tasten auf der Tastatur nach oben bzw. unten bewegt werden können. Da das Spiel mit zwei Spielern gespielt werden soll, bieten sich die Tasten Q und A für Spieler 1 und Pfeil hoch/Pfeil herunter für Spieler 2 an.
2. *Bewegung des Balls:* Die Bewegung des Balls erfordert deutlich mehr Aufwand. Je nach Kenntnisstand der Schüler sollte diese vorgegeben werden. Bei sehr engagierten und älteren Schülerinnen und Schülern können auch größere Teile hiervon selbst entwickelt werden, dies wird aber im Vorfeld eine intensivere Besprechung erfordern. Generell soll der Ball realistisch von den Wänden abprallen können. Am einfachsten zerlegt man hierzu die Bewegung des Balls in eine Bewegung in Richtung der horizontalen x-Achse und der vertikalen y-Achse. Beim Abprallen an der oberen und unteren Wand wird das Vorzeichen in y-Richtung umgekehrt, bei Zusammenprall mit einem Schläger wird das Vorzeichen in x-Richtung umgekehrt.
3. *Zählen der Punkte:* Einen Punkt erhält der Spieler dann, wenn der Ball am gegnerischen Schläger vorbei an die hintere Wand anstößt. Den einfachsten Punktezähler erhält man, wenn man

einen farbigen Strich ans Ende des Spielfeldes setzt und den Ball auf einen Zusammenstoß mit ebendiesem überprüft. Wird ein solcher Zusammenstoß detektiert, so erhält der entsprechende Spieler einen Punkt und der Ball wird wieder zurück an die Startposition gesetzt.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

Es kann anschließend eine weitere Programmierphase angestoßen werden, bei der jedem die Chance gegeben wird, die vorhandenen Mängel noch zu beheben.

4.4.3 Erweiterungen

- *Erschwerte Bedingungen:* Mit jedem Punkt Differenz erhöht sich die Schwierigkeit für den führenden Spieler. Dies könnte z. B. passieren, indem seine Schlagfläche verkleinert oder die Bewegungsgeschwindigkeit des Schlägers verringert wird.
- ...

4.5 Zitterlabyrinth

Erforderliche Kompetenzen: Fühlen, Schleifen

Gewonnene Kompetenzen: Variablen (Zeit Stoppen), Bewegung (Mauszeiger verfolgen)

4.5.1 Beschreibung

Bei diesem Konzentrationsspiel soll mit der Maus eine Figur durch ein enges Labyrinth bewegt werden, dessen Wände nicht berührt wer-

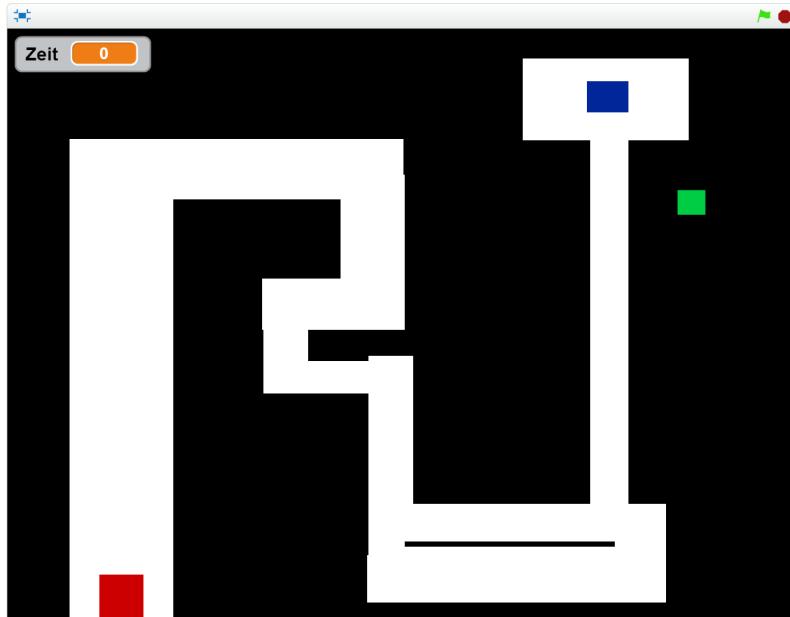


Abbildung 4.5: Zitterlabyrinth

den dürfen. Zusätzlich soll die benötigte Zeit zum durchqueren des Labyrinths gemessen werden.

4.5.2 Durchführung

Phase 1: Planung

Während der Planungsphase soll mit den Schülern folgende Schritte vollzogen werden:

- Erläuterung des Blocks “Gehe zu Mauszeiger”
- Ideensammlung zu den Funktionen des Programms: Bewegung der Figur, Erkennung der Wandberührung
- Überlegung anstellen, wie man den Mauszeiger von Startfählenchen zum Beginn des Labyrinths bekommen kann, ohne bereits die Wände zu berühren.

Phase 2: Vorbereitungen

Zunächst sollte der Hintergrund, also das Labyrinth mit farblich gekennzeichneten Start- und Endpunkt gezeichnet werden. Anschließend soll die Figur gezeichnet werden. Dabei ist darauf zu achten, dass das Kostüm sich zentriert im Bild befindet, damit sie sich nicht verschoben zum Mauszeiger bewegt.

Phase 3: Programmierung

1. Bewegung der Figur: Die Bewegung der Figur kann mit einer simplen Wiederholungsschleife, die den Block “Gehe zu Mauszeiger” enthält, programmiert werden.

2. Kontrolle der Wandberührung: Bewegung der Figur und Wandberührung können als parallele Skripte programmiert werden.

Die Kontrolle der Wandberührung darf jedoch erst einsetzen, sobald die Figur den Startpunkt berührt hat. Möglich ist dies mit dem “Warte bis”-Block im Bereich Steuerung.

3. Zeitmessung: Die Zeitmessung soll ebenfalls mit der Berührung des Startpunkts beginnen und mit dem Berühren des Endpunktes enden. Dazu wird eine Variable “Zeit” angelegt, die den Startwert Null besitzt und bis zur Berührung des Endpunkts die verstrichene Zeit zählt, indem jede 0.1 Sekunden die Variable um 0.1 erhöht wird.

Detaillierte Programmbeschreibung

Das Programm besteht aus einem Bühnenbild und einer Figur. Das Bühnenbild enthält das Labyrinth, durch das die Figur bewegt werden soll. Zudem sind der Start- und der Endpunkt des Labyrinths mit Farbpunkten versehen, damit die Zeitmessung installiert werden kann und damit die Figur vom „grünen Fähnchen“ zum Startpunkt bewegt werden kann, ohne mit den Wänden des Labyrinths zu kollidieren.

Zunächst muss die Bewegung der Figur programmiert werden. Da die Figur immer dem Cursor folgen soll, wird im Skript der Figur in einer Wiederholungsschleife der Block “gehe zu Mauszeiger” ausgeführt.

Anschließend muss also von der Figur das Erreichen des Startpunktes (in der Musterlösung rot) registriert werden. Dazu wird der “warte

bis wird Farbe rot berührt"-Block verwendet. Anschließend wird mit dem "warte bis wird Farbe schwarz berührt"-Block die Kollision mit der Labyrinthwand detektiert.

Die Zeitmessung wird parallel zur Wanddetektion im Skript der Figur implementiert, und zwar mit einer Variable namens Zeit, die bei Erreichen des Startpunktes auf Null gesetzt wird und anschließend in einer Schleife mit der Abbruchbedingung "wird Farbe blau berührt" hochgezählt (blau ist Farbe des Endpunktes).

Außerdem wird die Größe der Figur bei Programmbeginn auf 100 gesetzt, da sich bei Kollision mit der Wand die Figur aufbläht, um die Wandberührung zu verdeutlichen.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

4.5.3 Erweiterungen

- *Mehrere Levels:* eine besonders schnelle Schülerin könnte ein oder zwei zusätzliche Hintergrundbilder (also Labyrinthe) zeichnen und das Spiel auf mehrere Levels skalieren: Man beginnt mit Level 1 und wenn man gewinnt, wechselt das Bühnenbild und man kommt zu Level 2 etc. Hier könnte man den Befehl "Wechsle zu Bühnenbild ()" verwenden.

- ...

4.6 Helikopter

Erforderliche Kompetenzen:Bewegung, Bedingungen, Schleifen Gewonnene Kompetenzen: Kostüme Aussehen, Zufall

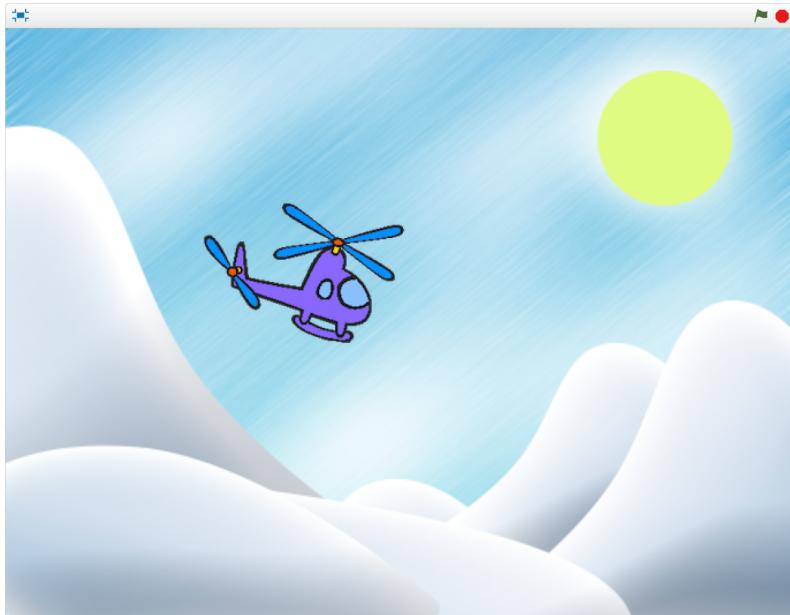


Abbildung 4.6: Helikopter

4.6.1 Beschreibung

Der Spieler steuert einen Helikopter durch einen Regen von Hindernissen, welchen er ausweichen soll. Der Helikopter kann mit Pfeiltasten in alle Richtungen auf dem ganzen Spielfeld bewegt werden. Die Hindernisse sind bunt und ändern kontinuierlich die Farbe. Stößt der Helikopter mit einem der Hindernisse zusammen, bricht er auseinander und das Spiel ist vorbei.

4.6.2 Durchführung

Phase 1: Planung

In der Planungsphase werden mit den Schülern die folgenden Dinge besprochen:

- Wie fügt man die gewünschte Figur aus der Scratch-Bibliothek in das Spiel hinzu. Wie fügt man ein zusätzliches Zerstörungs-

kostüm hinzu? Wo kann man in der Scratch-Umgebung ein Quadrat (als Hindernis) zeichnen?

- Wie programmiert man das Bewegen des Helikopters und das Spielende durch Zusammenstoß mit dem Hindernis?
- Und die wichtigste Frage: Wie programmiert man die Bewegung des Quadrats (als Hindernis) und seine Vermehrung?

Phase 2: Vorbereitungen

In der Vorbereitungsphase zeichnen die Schüler das Hindernis (ein Quadrat oder Ähnliches) oder wählen eines aus der Scratch-Bibliothek aus. Außerdem fügen sie dem Helikopter (oder einer anderen gewählten Figur wie z.B. ein Flugzeug) ein Zerstörungskostüm hinzu.

Phase 3: Programmierung

1. Bewegung des Helikopters mit den Pfeiltasten
2. Helikopter fühlt den Zusammenstoß mit dem Hindernis, worauf das Spiel endet
3. Ein Quadrat nach dem anderen taucht rechts irgendwo zufällig auf und fliegt nach links
4. Jedes Quadrat ändert kontinuierlich seine Farbe.

Detailierte Programmbeschreibung

Das Zerstörungskostüm für den Helikopter kann man erstellen, indem man in der Scratchumgebung den Helikopter aus der Scratch-Bibliothek lädt, beim Helikopter auf Kostüme klickt, das aktuelle Kostüm per Rechtsklick dupliziert und das neu erzeugte Kostüm modifiziert. Man verändert es, indem man mit dem Auswahl-Tool einige Teile des Helikopters auswählt und diese auseinander zieht.

Wenn die Flagge angeklickt wird erzeugt die Figur "Hindernis" mit Hilfe eines "wiederhole fortlaufend"-Blocks jede Sekunde einen Klon von sich selbst. Unter dem "Wenn ich als Klon entsteh"-Block wird die x-Koordinate fest auf 300 (hinter des rechten Rands) und die y-Koordinate auf eine zufällige Zahl zwischen -180 und 180 gesetzt. Dann tritt der Klon in eine Schleife, in welcher er sich nach links

bewegt und den Effekt seiner Farbe ändert. Die Schleife bricht ab, sobald die x-Position kleiner als -200 ist. Danach löscht sich der Klon selber mit dem Befehl “lösche diesen Klon”.

Helikopter detektiert außerdem die Kollision mit dem Hindernis in einer Endlosschleife, woraufhin der Helikopter zum Zerstörungskostüm wechselt. Nach einer Sekunde wird das Spiel mit “stop all” beendet.

Phase 4: Testen und vorstellen

In der Vorstellungsphase kann ein beliebiger Schüler sein Spiel vorstellen, falls er es möchte. Dabei kann man explizit auf die einzelnen Teilprobleme hinweisen, welche der Schüler gelöst hat. Aber man kann die anderen auch fragen, was er verbessern könnte, wenn er an diesem Projekt nächste Stunde noch arbeiten will.

4.7 Pacman

Erforderliche Kompetenzen: BEWEGUNGEN, FÜHLEN, BEDINGUNGEN

Erworbene Kompetenzen: Fortgeschritten

4.7.1 Beschreibung

Dieses Projekt ist deutlich fortgeschritten. Es geht um die Programmierung des bekannten Spiels “Pacman”. Die Schülerinnen und Schüler haben hier die Möglichkeit, von der Planungsphase des Projekts über das Zeichnen von Hintergrund und Gestaltung der Figur alle wesentlichen Abläufe der Spieleprogrammierung kennenzulernen.

Es ist besonders bei diesem Projekt sehr wichtig, dass die Schülerinnen und Schüler zu jedem Zeitpunkt sehr klare Vorstellungen über ihre nächste Aufgabe haben und nicht tatenlos am PC sitzen. Daher gibt es einige Erweiterungen für besonders schnelle Schülerinnen und Schüler.

4.7.2 Durchführung

Im Folgenden ist die Durchführung des Projekts geschildert. Sollte es triftige Gründe geben, kann die Lehrperson gerne auch ein alternatives Konzept heranziehen.

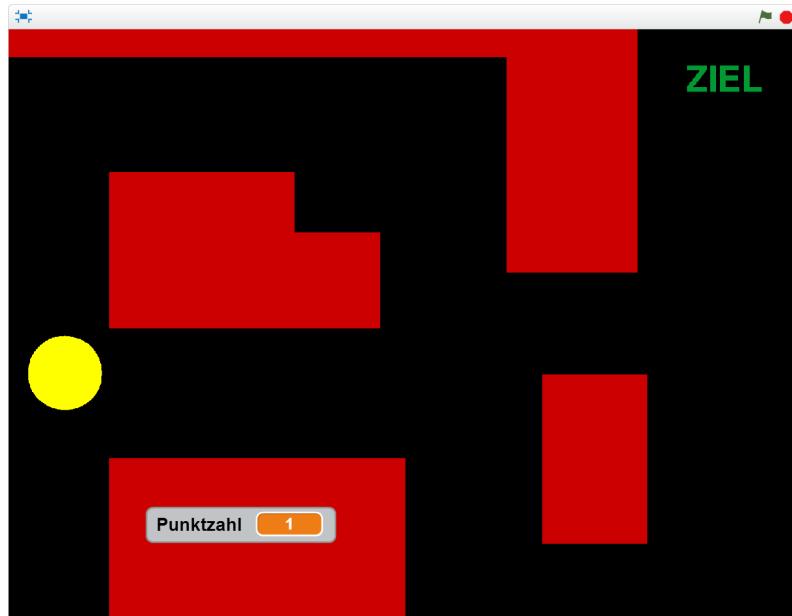


Abbildung 4.7: Pacman

Phase 1: Planung

Die Planungsphase wird in den ganzen Gruppe vollzogen. Die Lehrperson erläutert das Projekt und die Ziele der aktuellen Stunde werden geklärt. Beim Projekt Pacman müssen konkret geplant werden:

- Präsentation eines komplett fertigen Spiels
- Überlegungen und Ideensammlung für Hintergrund, Figur, etc.
- Überlegungen zur Programmierung und der nötigen Aufgaben:
 - Bewegungen der Spielfigur auf Tastendruck: Umsetzung durch entsprechende Ereignisse
 - Anstoßen der Spielfigur an die Ränder des Spielfelds
 - Anstoßen der Spielfigur an die Mauern
 - “Essen” der Spielfigur

- Punktezähler und dessen Funktionsweise
- Nach dem Sammeln der Ideen bestimmen einer sinnvollen Reihenfolge, diese TODO Liste könnte z.B. an die Tafel geschrieben werden.

Mit der nun entstandenen Aufgabenliste können die Schüler mit der eigentlichen Umsetzung des Projekts beginnen.

Phase 2: Objekte und Hintergrund zeichnen

Es bietet sich auch aus Zeitgründen an, im Anschluss an die Planung mit dem Zeichnen von Hintergründen, Figuren etc. die erste Stunde abzuschließen, da dies relativ leicht umsetzbare Aufgaben sind. Besonders schnelle Schülerinnen und Schüler können anschließend direkt mit der Programmierung beginnen.

Phase 3: Programmieren

Hier werden die zuvor besprochenen Aufgaben in Programmcode umgesetzt. Es macht Sinn, dabei die folgende Reihenfolge einzuhalten:

1. *Steuerung der Figur:* Zuerst programmieren die Schülerinnen und Schüler die Bewegung der Spielfigur. Die Figur soll sich dabei konstant in Vorförwärtsrichtung bewegen (in einer Endlosschleife, dem *Game Loop*). Auf Tastendruck (wird über Steuerungsböcke abgegriffen) findet eine Drehung statt. Es können die Tasten A, S, D, F oder das Steuerkreuz verwendet werden.
2. *Anstoßen der Spielfigur an den Rand:* Beim Anstoßen an den Spielfeldrand soll sich die Figur nicht weiterbewegen
3. *Anstoßen der Spielfigur an die Wände:* Wenn die Spielfigur die Wände des Spielfelds (gezeichnet in einer möglichst einmalig vorhandenen Farbe) berührt, soll das Spiel als verloren gezählt werden.
4. *Essen ...*
5. *Punktezähler:* Der Punktezähler wird über eine Variable realisiert. Die Variable muss angelegt und benannt werden und anschließend im Modus *Groß* auf dem Spielfeld angezeigt werden. Bei Neustart des Spiels wird die Variable auf null gesetzt, bei dem Essen eines Elements wird sie entsprechend erhöht.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

4.8 Trampolin

Erforderliche Kompetenzen: Bewegung, Schleifen Gewonnene Kompetenzen: Tastendrücke, Kostüme



Abbildung 4.8: Trampolin

4.8.1 Beschreibung

In diesem Programm springt Scratch (die Katze) auf einem Trampolin und kann in der Luft Figuren wie Drehungen und Salti ausführen.

Durch Gedrückthalten der Leertaste springt Scratch höher. Auf die Tasten „a“ und „s“ reagiert Scratch mit Salti und auf die Taste „d“ mit einer Drehung.

4.8.2 Durchführung

Phase 1: Planung

Während der Planungsphase soll mit den Schülern folgende Schritte vollzogen werden:

- Testen des fertigen Programmes durch die Schüler
- Erläuterung des Blocks “wird Taste * gedrückt”
- Ideensammlung zu den Funktionen des Programms: Drehung und Salto der Katze; auch weitere Funktionen möglich !

Phase 2: Vorbereitungen

Auswählen des Hintergrundes.

Phase 3: Programmierung

1. Aktionen der Figur wie z.B. Salti
2. Ggf. erstellen weiterer Kostüme für Kunststücke

Detaillierte Programmbeschreibung

Zunächst wird die Physik des Spiels (das Fallen und Abprallen am Trampolin) programmiert und in den unteren Teil des Skripts verschoben, und soll vom Schüler ignoriert werden. Dies ist in der Vorlage enthalten.

Zum Programmieren der Physik ist die Variable Geschwindigkeit nötig. Um ihren Wert wird die y-Position von Scratch kontinuierlich geändert. Ist Geschwindigkeit größer Null so steigt Scratch nach oben, ist die Geschwindigkeit kleiner als Null, so fällt Scratch. Wenn nun die Geschwindigkeit ebenfalls kontinuierlich verringert wird, so

entsteht der Eindruck eines freien Falles. Dazu wird die Variable Geschwindigkeit, bei jedem Durchlauf der Hauptschleife im Block “Beuge dich” verringert und die y Position von Scratch um den Wert der Geschwindigkeit verändert.

Zusätzlich wird in der Hauptschleife abgefragt, ob das Trampolin berührt wird. Ist dies der Fall so muss die Geschwindigkeit ihr Vorzeichen ändern, was mit der Addition der Geschwindigkeit um minus zweimal der Geschwindigkeit geschieht.

$$(\text{Geschwindigkeit} = \text{Geschwindigkeit} - 2 * \text{Geschwindigkeit})$$

Wird zum Zeitpunkt der Berührung des Trampolins auch die Leertaste gedrückt, so wird mithilfe des Blocks “Springe hoch um 5” die nun positive Geschwindigkeit um fünf erhöht.

Außerdem wird bei jeder Berührung des Trampolins der Geschwindigkeitsbetrag um eins verringert, sodass Scratch irgendwann nicht mehr springt, was den Eindruck von Luftreibung hervorruft. An der Stelle kann man die Schüler auch fragen, was Luftreibung bedeutet.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

4.8.3 Erweiterungen

- *Mehrspielermodus:* Eine schnelle Schülerin könnte noch eine zweite Figur ins Spiel einbauen, welche mit anderen Tasten gesteuert werden kann. Dann können zwei Spielerinnen in demselben Spiel sich darum konkurrieren, wer die besseren Tricks ausführen kann.

4.9 Wettrennen

Erforderliche Kompetenzen: Bewegungen, Fühlen, Bedinungen

Erworbane Kompetenzen: Spieleprogrammierung, Variablen

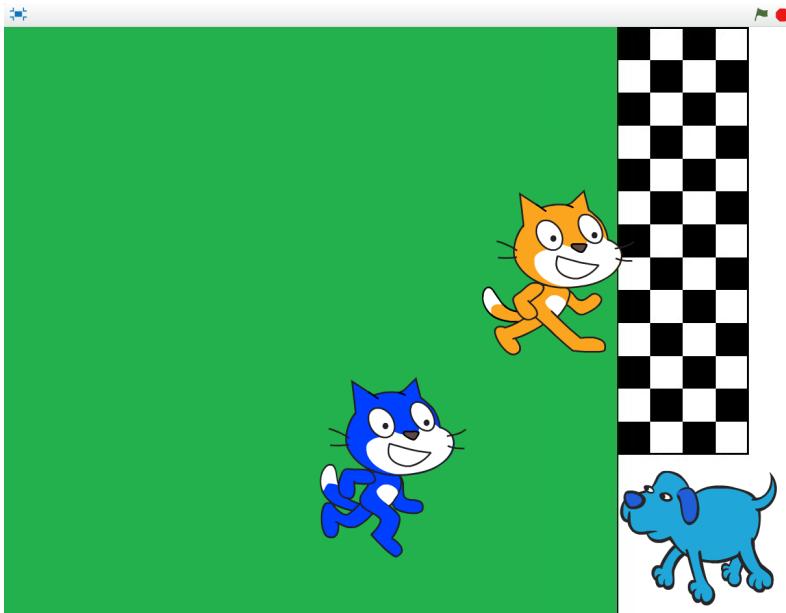


Abbildung 4.9: Wettrennen

4.9.1 Beschreibung

Dieses Projekt eignet sich sehr gut zum Einstieg in die Spieleentwicklung, denn es handelt sich um ein einfaches dass zwei Spieler gegeneinander auf einer Tastatur spielen können. Bei dem Projekt sollen zwei Spielfiguren vom linken zum rechten Bildrand "um die Wette laufen". Die Steuerung des Spiels gestaltet sich dabei einfach: Jeder Spieler muss möglichst schnell abwechselnd zwei ihm zugeordnete Tasten drücken. Bei jedem Tastendruck bewegt sich die Figur einen Schritt in Richtung Ziel am rechten Bildrand.

4.9.2 Durchführung

Phase 1: Planung

Während der Planungsphase sollten mit den Schülerinnen und Schülern die folgenden Schritte vollzogen werden:

- Erläuterung des Projektes
- Überlegungen und Ideensammlung für Hintergrund, Figuren, etc.
- Überlegungen zur Programmierung und der nötigen Aufgaben:
 - Bewegung der Spieler/Steuerung
 - Erkennen des Zieldurchlaufs
- Sammeln aller Ideen und Erstellung der TODO Liste

Anschließend kann mit der Umsetzung des Projekts begonnen werden.

Phase 2: Vorbereitungen

Zunächst sollten die Figuren und der Hintergrund gezeichnet werden. Für die Figuren bieten sich einfache Striche an, die auf und ab bewegt werden können, die Gestaltung des Hintergrunds ist den Schülerinnen und Schülern überlassen.

Phase 3: Programmierung

Im Anschluss an die vorbereitenden Maßnahmen kann mit der Programmierung des Spiels begonnen werden. Als Vorschlag sei die folgende Reihenfolge genannt, in der Schritt für Schritt leicht validierbare Teile des Spiels entstehen:

1. *Steuerung der Figuren:* Die Figuren sollen jeweils auf zwei Tasten mit einem 10er-Schritt reagieren. Allerdings sollen die Tasten abwechselnd betätigt werden müssen.
2. *Erkennung des Ziels:* Ob eine Figur das Ziel erreicht hat, kann über die Bedingung realisiert werden, ob die Farben des Zielbanners berührt werden.

3. *Ende des Spiels:* Sobald eine Figur das Ziel erreicht hat, soll das Spiel beendet werden. Dies ist einfach mit dem Block “Stoppe alles” zu realisieren.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

Es kann anschließend eine weitere Programmierphase angestoßen werden, bei der jedem die Chance gegeben wird, die vorhandenen Mängel noch zu beheben.

4.9.3 Erweiterungen

Das Spiel kann um einen Schiedsrichter erweitert werden, der einen Countdown herunterzählt und anschließend eine Nachricht verschickt, die die Bewegung der Figuren ermöglicht.

4.10 Aquarium

Erforderliche Kompetenzen: Bewegungen, Fühlen

Erworbene Kompetenzen: Programmieren mehrerer Figuren, Operatoren

4.10.1 Beschreibung

Dieses Projekt eignet sich sehr gut zum Einstieg in die Spieleanthropologie. Ziel des Spiels ist es, eine steuerbare Figur an anderen computergesteuerten Figuren vorbeizubewegen, ohne diese zu berühren.

In dem Aquariums befinden sich mehrere Fische, die es fortlaufend durchqueren. Gleichzeitig befindet sich eine vom Spieler steuerbare Figur in dem Aquarium, die zum oberen Rand des Spielfeldes bewegt

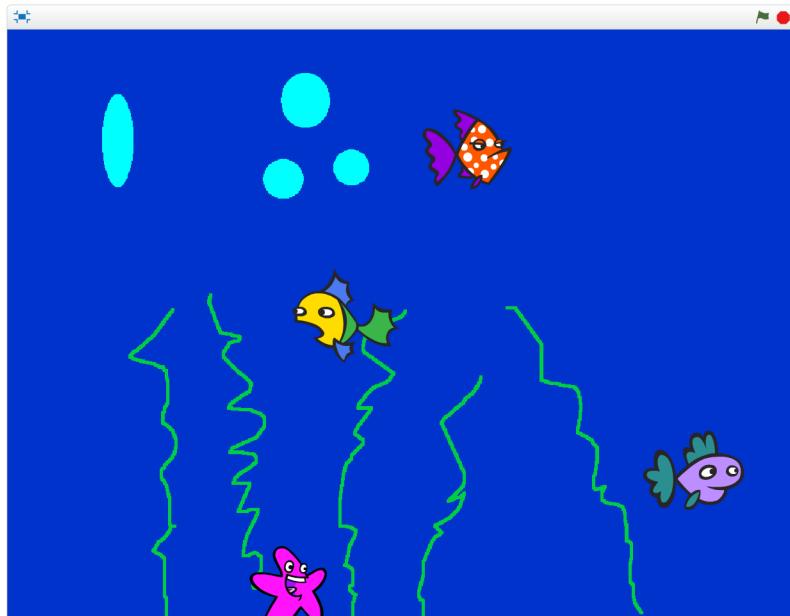


Abbildung 4.10: Aquarium

werden soll, ohne einen Fisch zu berühren. Wird ein Fisch berührt, so sinkt die Figur zurück auf die Startposition.

4.10.2 Durchführung

Phase 1: Planung

Während der Planungsphase sollten mit den Schülerinnen und Schülern die folgenden Schritte vollzogen werden:

- Erläuterung des Projektes
- Überlegungen und Ideensammlung für Hintergrund, Figuren, etc.
- Überlegungen zur Programmierung und der nötigen Aufgaben:
 - Bewegung der Fische

- Bewegung der Spielfigur/Steuerung
- Erkennen der Kollision zwischen Fisch und Figur
- Sammeln aller Ideen und Erstellung der TODO Liste

Anschließend kann mit der Umsetzung des Projekts begonnen werden.

Phase 2: Vorbereitungen

Zunächst sollte der Hintergrund gezeichnet werden. Für die Figuren bieten sich diverse Fische, sowie Taucher, Seesterne oder Oktopusse aus der Scratch-Bibliothek an. Die Gestaltung des Hintergrunds ist den Schülerinnen und Schülern überlassen.

Phase 3: Programmierung

Im Anschluss an die vorbereitenden Maßnahmen kann mit der Programmierung des Spiels begonnen werden. Als Vorschlag sei die folgende Reihenfolge genannt, in der Schritt für Schritt leicht validierbare Teile des Spiels entstehen:

1. *Bewegung der Fische:* Die Fische folgen einem linearen Bewegungspfad, der sie beim anstoßen an den Rand wieder am anderen erscheinen lässt. Wichtig ist eine fortlaufende Schleife.
2. *Steuerung der Spielfigur:* Die Figur lässt sich am besten über die Pfeiltasten steuern, wobei jede einer simplen Bewegung in die entsprechende Richtung zugeordnet wird. Hierbei lassen sich die Unterschiede zwischen den “gehe” und “ändere x/y” Befehlen erklären.
3. *Kollision* Es muss ein Erkennen verschiedener Figuren erfolgen können. Hierzu bietet sich die Einführung des “oder” Operators an. Nach der Kollision gleitet die Spielfigur zurück an ihren Startpunkt. Hierbei ist ein Kostümwechsel möglich.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, müssen alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte

4 Spieleprogrammierung

Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben!

Es kann anschließend eine weitere Programmierphase angestoßen werden, bei der jedem die Chance gegeben wird, die vorhandenen Mängel noch zu beheben.

4.10.3 Erweiterungen

Mögliche Erweiterungen sind z.B.:

- Einführung eines Ziels an der Wasseroberfläche
- Hinzufügen von statischen Hindernissen
- Kostümwechsel der Fische bei Kollision (erfordert Figurenkommunikation)

4.11 Katze fängt Maus

Erforderliche Kompetenzen: Steuerung, Bedingungen, Schleifen, Bewegung

Gewonnene Kompetenzen: Kostüme, Nachrichten

4.11.1 Beschreibung

In diesem Spiel steuert der Spieler eine Katze, die eine davonlaufende Maus jagt und frisst, falls sie diese fängt.

4.11.2 Durchführung

Phase 1: Planung

Zu Beginn soll das Programm an der Vorlage erläutert und durch die Schüler getestet werden. Dabei kann auf folgende Programmbestandteile aufmerksam gemacht werden:

- Die Steuerung der Katze durch den Spieler
- Die sich bewegenden Beine der Katze

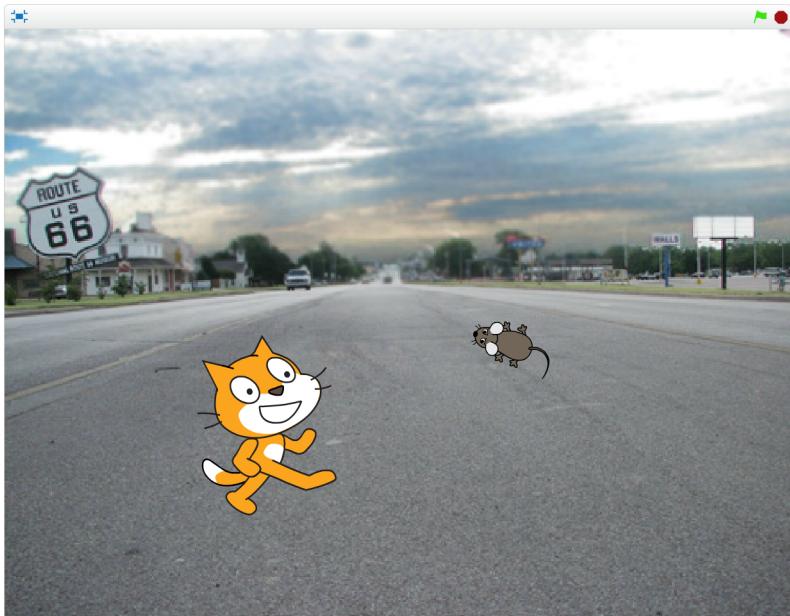


Abbildung 4.11: Katze fängt Maus

- Die automatische Flucht der Maus
- Das Erscheinen und Verschwinden der Maus

Phase 2: Vorbereitungen

Nun soll die neuen Kompetenzen erläutert und getestet werden. Dazu können kleine Beispielprogramme oder kleine Programmieraufgaben verwendet werden.

Als Vorlage für dieses Projekt wird ein Template verwendet, der bereits den Block "Lauf weg" der Maus und das Schrumpfen der Figuren im oberen Spielfeldbereich enthält. (Lässt sich aus Vorlagen entnehmen)

Phase 3: Programmierung

Die Bewegung der Katze programmieren: Zunächst soll die Katze programmiert werden. Durch Drücken der Pfeiltaste nach oben, soll die

4 Spieleprogrammierung

Katze einen Schritt gehen und ihr Kostüm wechseln. Durch Drücken der Pfeiltasten links und rechts, soll sich die Katze drehen. Am einfachsten ist dies durch eine “wiederhole bis”-Schleife zu realisieren. Als Bedingung wird die Berührung mit der Maus eingesetzt.

Interaktion zwischen Katze und Maus programmieren: Beim Starten des Programmes soll die Katze ihren Hunger auf Mäuse bekunden und anschließend die Nachricht “Lauf weg” an alle versenden, um der Maus den Beginn der Jagd mitzuteilen.

Die Maus soll nun bei Erhalten der Nachricht sichtbar werden (“zeige dich”-Block) und den bereits definierten Block “Lauf weg” ausführen.

Außerdem soll nach dem Kontakt mit der Maus, der ebenfalls detektiert werden muss, (z.Bsp. durch die “wiederhole bis”-Schleife) die Nachricht “Ich hab dich gefressen!” versendet werden.

Die Maus wiederum soll auf diese Nachricht mit reagieren, indem sie sich versteckt (“versteck dich”-Block) und alle Skripte dieser Figur stoppt.

Phase 4: Testen und vorstellen

In der letzten Phase, die auch zweimal durchgeführt werden kann, können alle (bei großen Gruppen: ausgewählte) Schülerinnen und Schüler ihr Projekt vorstellen. Hier bietet es sich an, wenn eine dritte Person das Programm bedienen muss und nicht die Programmiererin oder der Programmierer selbst. Der restliche Kurs schaut bei der Vorstellung zu und gibt hinterher eine Rückmeldung. Für jedes Projekt kann so analysiert werden, was gut und was weniger gut funktioniert und noch verbessert werden muss. Wichtig vor allem bei jüngeren Klassenstufen: Die guten Teile des Programmes besonders hervorheben! Es kann anschließend eine weitere Programmierphase angestoßen werden, bei der jedem die Chance gegeben wird, die vorhandenen Mängel noch zu beheben.

4.11.3 Erweiterungen

Das Spiel kann durch eine Zeitmessung erweitert werden, indem durch Verwendung der Stoppuhr die Zeit gemessen wird, die die Katze braucht, um die Maus zu fangen.

4.12 Fang mich!

Erforderliche Kompetenzen: Bewegungen, Steuerung, Fühlen, Senden und Empfangen

Gewonnene Kompetenzen: Stoppuhr, Variablen

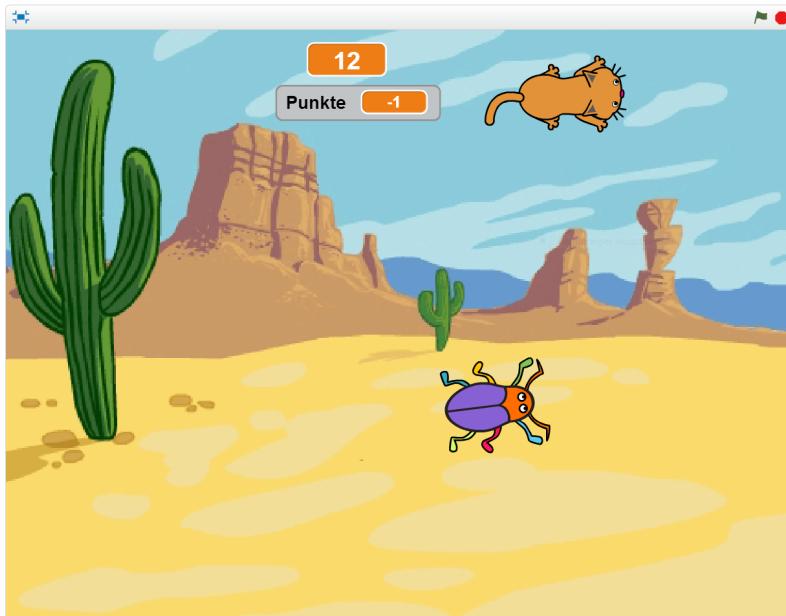


Abbildung 4.12: Fang mich!

4.12.1 Beschreibung

Die Katze und der Käfer jagen einander. Dabei tauschen sie die Rollen, sobald der Jäger sein Ziel gefangen hat, oder 30 Sekunden abgelaufen sind. Dabei soll der Käfer computergesteuert, aber die Katze spielergesteuert sein. Es gibt höchstens fünf Punkte zu erreichen.

4.12.2 Durchführung

Phase 1: Planung

In der Planungsphase werden mit den Schülern die folgenden Dinge besprochen:

- Wie programmiert man den Käfer so, dass er die Katze fängt?
- Wie programmiert man den Spielablauf, d.h. dass sich Katze und Käfer abwechselnd fangen?

Phase 2: Vorbereitungen

Die Schüler laden sich zwei beliebige Figuren aus der Scratchbibliothek herunter oder zeichnen eigene Figuren. Das Gleiche gilt für den Hintergrund.

Phase 3: Programmierung

Die benötigten Schritte der Programmierung umfassen:

- Steuerung der Maus per Tastatur
- Automatische Steuerung des Käfers, sodass er die Katze jagt
- Erkennen, dass die Katze den Käfer bzw. der Käfer die Katze gefangen hat
- Rollentausch implementieren, d.h. das der bisherige Jäger nun der Gejagte ist
- gegebenenfalls: Stoppuhr und Punktezählung

Phase 4: Testen und vorstellen

In der Vorstellungsphase kann ein beliebiger Schüler sein Spiel vorstellen, falls er es möchte. Dabei kann man explizit auf die einzelnen Teilprobleme hinweisen, welche der Schüler gelöst hat. Aber man kann die anderen auch fragen, was er verbessern könnte, wenn er an diesem Projekt nächste Stunde noch arbeiten will.

4.12.3 Detaillierte Programmbeschreibung

Die Bewegung ist die fortlaufend wiederholte Ausführung von “falls dann”-Blöcken realisiert, die einen Tastendruck detektieren. Grundsätzlich hat die Figur “Spieler”, d.i. die Katze, zwei Blöcke für die zwei Fälle “when I receive CompiFolgtSpieler” sowie “when I receive SpielerFolgtCompi”, welche sehr ähnlich ablaufen, deshalb nur die Erklärung für den Ersteren: Die Katze gleitet zur Anfangsposition (links unten) und solange sie den Käfer noch nicht berührt hat, führt sie die Bewegungen aus, welche mit der Tastatur an sie weitergeleitet werden. Dabei wird die Zeit aktualisiert und falls die Zeit abläuft, hat die Katze gewonnen (weil sie die Weglaufende war) und kriegt deshalb einen Punkt. Dann wird die Nachricht “SpielerFolgtCompi” gesendet, damit die Figuren die Rollen tauschen und das aktuelle Skript wird gestoppt, damit die Katze den Code fürs Fangen (anstatt fürs Weglaufen) ausführen kann. Falls die Zeit aber nicht abläuft und die Schleife verlassen wird, also die Katze gefangen wurde, wird ein Punkt abgezogen und die Nachricht “SpielerFolgtCompi”, d.i. die Abwechslung, wird gesendet. Wenn allerdings die Katze bereits -5 Punkte hat, erscheint ein “Verloren!” und das Spiel wird mit “stop all” komplett terminiert.

Die Figur “Compi” (der Käfer) macht grundsätzlich das Gleiche mit zwei Ausnahmen: Der Käfer hat eine Methode “Fange Spieler!”, welche nur die eigene Ausrichtung in Richtung Katze setzt und darauf losmarschiert. Die zweite Ausnahme ist die Methode “Zufällig weglaufen”: Alle 250 Millisekunden wird hier die Richtung des Käfers umverändert.

5 Animationen und Geschichten

In diesem Kapitel geht es um das erstellen von animierten Geschichten in Scratch. Diese bieten eine großartige Möglichkeit um in frühen Stadien des Kurses die Wichtigkeit der Reihenfolge zu vermitteln. Später eignen sie sich um Fähigkeiten in den Bereichen Kostüme, Nachrichten, Schleifen, Kommunikation etc. zu vertiefen.

Diese Form von Projekt lädt gerade bei kreativen Schülern dazu ein diese sich auch eigene Geschichten ausdenken zu lassen - man sollte nur darauf achten, dass diese auch noch realisierbar bleiben.

5.1 Anregungen für Geschichten

Die Geschichten lassen sich in Länge und Themenfeldern variieren. Es sollte gerade am Anfang darauf geachtet werden, dass bei einer vorgegebenen Geschichte alle nötigen Figuren in der Scratch-bibliothek vorliegen. Hintergründe können selbst gestaltet werden, liegen zu einigen Themenbereichen aber auch vor.

Mögliche Geschichten für Einsteiger wären beispielsweise:

- Ein Ritter geht zu einer Burg/Höhle oö und trifft dort einen Drachen, den er verjagt um eine Prinzessin zu retten.
- Eine Hexe lauert einer anderen Person auf, um sie in ein Tier zu verwandeln.
- Ein Taucher erkundet eine Unterwasserwelt und trifft dort unterschiedliche Tiere.
- Eine Rakete/ein Astronaut erleben Abenteuer im Weltall.
- Eine Party der Scratch-Maskottchen

5.2 Vertiefungen

Wie bereits am Anfang dieses Kapitels gesagt lassen sich mit animierten Geschichten ganz gezielt bestimmte Fähigkeiten erlernen.

Es können hierzu eigene Geschichten extra konzipiert werden, die Wert auf eine bestimmte Aktion legen. Dadurch können die Schüler kreativ die Geschichte gestalten, und lernen dennoch neue Fähigkeiten. Diese Form von Geschichte eignet sich vor allem, wenn bereits ein gewisses Grundverständnis besteht, aber noch nicht alle Funktionen von Scratch erlernt sind. Die Projekte "Maus zum Käse" und "Tanzstunde" fallen mehr oder weniger auch unter diese Form von Animation.

Einige Beispiele für Anwendungen von Geschichten im Bezug auf besondere Fähigkeiten:

- Um das Verständnis von Kostümen zu erhöhen: Eine Geschichte über Figuren, die sich verwandeln (egal ob willentlich oder nicht). Hierdurch muss verstanden werden, was der Unterschied zwischen dem Hinzufügen neuer Figuren und dem Hinzufügen von Kostümen zu einer bestehenden ist.
- Für das Erlernen von Nachrichten ist ein Szenario in dem miteinander Gesprochen wird oder Befehle ausgegeben werden sinnvoll, da der Gedanke einfach auf eine Nachricht zu übertragen ist und das Konzept schnell klar wird.

6 Abschließendes

6.1 Einige Worte zum Abschluss

Ich hoffe, Sie konnten dieses Buch für einige hilfreiche Anregungen finden und eine interessante Informatik-AG durchführen. Sollten Sie Anmerkungen haben, Fehler in diesem Buch gefunden haben oder auch neue Ideen für Projekte haben, so schreiben Sie gerne eine Email an

steffen.schneider@it-for-kids.org.

Mehr Informationen finden sich auch auf unserer Homepage, www.it-for-kids.org

6.2 Eigener Beitrag zum Lehrkonzept

Unser Lehrkonzept basiert auf den Ideen und Beiträgen unserer Kursbetreuer. Wenn du oder Sie einen Beitrag dazu leisten möchten, dass in Zukunft weitere Projekte in unser Modulhandbuch aufgenommen werden, ist dies nach Registrierung auf unserer Homepage möglich. Wir freuen uns auf viele spannende und interessante neue Projektideen!

Anhang

„Aquarium“

In diesem Projekt wollen wir einen Seestern Programmieren, der durch ein Aquarium gesteuert werden kann. Doch vorsicht, wird er von den Fischen berührt, fällt er wieder auf seine Ausgangsposition zurück.

Zusätzlich zu den bisher bekannten Befehlen zur Bewegung und dem Programmstart benötigt man bei diesem Programm noch ein paar mehr Befehle. Es wird empfohlen selbst auszuprobieren, unten auf diesem Blatt ist aber auch eine Erklärung zu finden. Der Hintergrund kann selber entworfen werden und die Figuren selbst ausgewählt, es kann jedoch sein, dass diese skaliert werden müssen.

Die wichtigsten in der Musterlösung verwendeten Befehle (die Lösung kann hiervom abweichen):



Wie programmiert man die Fische?

Die Fische sollen die ganze Zeit durch das Bild schwimmen. Neben einem Startereignis benötigt man also einen immer fortlaufenden Befehl. Nun sollen die Fische bis zum Rand schwimmen und dann auf der anderen Seite wieder auftauchen.

Wie Programmiere ich den Seestern?

Der Seestern soll über die Pfeiltasten steuerbar sein. Die Bewegung des Seesterns soll ohne Drehung in alle Richtungen möglich sein.

Damit der Seestern „fällt“, wenn er einen Fisch berührt, benötigt man eine dauerhaft laufende Schleife, die die Kollision mit den Fischen abfragt. Darauf folgt ein Befehl, was infolgedessen geschehen soll. In diesem Fall wäre das ein zurückgleiten auf die Startposition.

Komplettlösung

Wie programmiert man die Fische?

Die Fische sollen die ganze Zeit durch das Bild schwimmen. Neben einem Startereignis („grüne Fahne“) benötigt man also einen immer fortlaufenden Befehl. Hierzu findet man unter „Steuerung“ den Befehl „Wiederhole fortlaufend“. Alles, was in dieser Klammer steht, wird dauerhaft ausgeführt. Nun sollen die Fische bis zum Rand schwimmen und dann auf der anderen Seite wieder auftauchen. Dies wird erreicht, indem man den ebenfalls unter „Steuerung“ zu findenden Befehl „Wiederhole bis“ kombiniert mit dem Befehl „wird ...“ (auszuwählen ist im dropdown-Menü „Rand“) berührt“ aus der Kategorie „Fühlen“. Hierdurch lässt sich festlegen, was beim berühren des Randes passiert. Dies sollte sein, dass der Fisch wieder zurück „springt“. Dies ist möglich über den „gehe zu“ Befehl aus der Kategorie „Bewegung“. Achtung, mögliche Fehlerquelle: dieser „Startpunkt“ darf nicht so liegen, dass der Fisch mit dem Rand in Berührung ist. Die Koordinaten für „gehe zu“ werden automatisch der momentanen Position angepasst, sodass man den Fisch einfach auf die Stelle ziehen kann und dann den Befehl einfügen.

Wie Programmiere ich den Seestern?

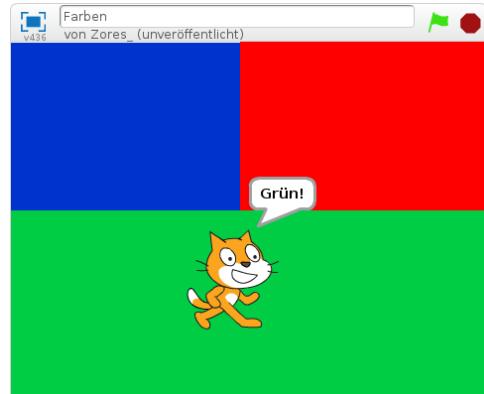
Der Seestern soll über die Pfeiltasten steuerbar sein. Hierfür verwenden wir den Befehl „Wird Taste gedrückt?“ aus der Kategorie Steuerung und wählen im Auswahlmenü die jeweilige Pfeiltaste aus. Die Bewegung des Seesterns geht nicht über „gehe ... Schritt“, sonder über „ändere x/y um ...“. Dies ermöglicht eine Bewegung in jede Richtung ohne eine Drehung. Anm. „ändere y“ bezieht sich auf die vertikale Ebene, „ändere x“ auf die horizontale. Um „vorwärts“ und „rückwärts“ gehen zu können muss man einmal einen negativen und einmal einen positiven Wert einsetzen.

Damit der Seestern „fällt“, wenn er einen Fisch berührt, benötigt man eine dauerhaft laufende („Wiederhole fortlaufend“) Schleife, die die Kollision mit den Fischen abfragt. Dies ist wieder möglich durch das Einfügen von „Wird ... berührt“ in ein „Warte bis“. Da es jedoch mehrere Fische geben soll müssen die „Wird ... Berührt“ Blöcke in „Oder“ Blöcke eingefügt werden. Darauf folgt ein Befehl, was infolgedessen geschehen soll. In diesem Fall wäre das ein zurückgleiten auf die Startposition mit dem Befehl „gleite zu x y in ... Sekunden“.

„Farben Erkennen“

Ziel des Projekts: In diesem Projekt soll die Katze sagen, welche Farbe sie gerade berührt. Berührt die Katze die Farbe grün, so soll sie zum Beispiel „grün“ sagen.

Dazu ist es wichtig zu verstehen, was eine **Bedingung** ist. Eine Bedingung ist sozusagen eine **Frage, die entweder mit ja oder mit nein** beantwortet werden kann. Lautet die Antwort ja, ist die Bedingung erfüllt. In diesem Programm soll die Katze gefragt werden, ob sie eine bestimmte Farbe berührt wird.



Das Bühnenbild:

Als erstes öffne nun ein **neues Projekt** unter „Datei“ und „neu“. Nun brauchst Du ein Farbiges Bühnenbild mit verschiedenen farbigen Flächen ähnlich wie im Bild. Das musst Du dir selber malen! Klicke dazu zunächst auf „**Bühne am unteren linken Rand**“ und anschließen auf den Reiter „**Bühnenbilder am oberen rechten Rand**“. Hier stehen dir nun verschiedene Werkzeuge zur Verfügung.

Am leichtesten geht es mit dem **Rechteck**!



Skripte Bühnenbilder Klänge



Wie kann ich herausfinden, welche Farbe die Katze berührt?

Zuerst brauchst Du den „**falls dann**“-Block. Den findest Du unter Steuerung.

Nun brauchst du die Bedingung. Die findest Du unter Fühlen, direkt unter Steuerung. Die Bedingung die Du brauchst lautet „**wird Farbe Grün berührt**“.



wird Farbe Grün berührt

In das sechseckige Feld des „falls dann“-Blocks kommt die Bedingung.

Am Ende sollte dein Programm ungefähr so aussehen:



Wie schaffe ich es nun, dass die Katze etwas sagt?

Dazu gibt es den „sage“-Block in der Kategorie Aussehen. Ziehe ihn **in** den „falls dann“-Block hinein und fülle den gewollten Text ein.

Programmiere nun alle Farben deines Bühnenbilds ein!



Viel Spaß und Erfolg! Bei Unklarheiten immer melden!

„Helikopter“

Beschreibung: In diesem Projekt sollst du einen Helikopter programmieren, der zufällig ins Bild fliegenden Hindernissen ausweichen muss oder sonst zerstört wird!

Schritt 1, der Helikopter:

Zuerst solltest du den Helikopter programmieren. Dafür musst du zunächst die passende Figur erstellen. Diese kannst du aus der Bibliothek in dein Programm laden. Klicke dazu auf den kleinen Kopf neben der Schrift „Neue Figur“.

Wie kann ich nun den Helikopter steuern?

Zuerst brauchst du ein „**Startereignis**“ aus der Kategorie



,,Ereignisse“. Hier brauchst du das Ereignis „**Wenn grüne Fahne angeklickt**“. Nun zur Steuerung:

Der Helikopter muss auf irgendeine Art und Weise **fühlen**, ob du eine Taste, z.Bsp. Pfeil nach oben, gedrückt hast.



Verwende nun den Block „**Taste Pfeiltaste nach oben gedrückt**“ aus der Kategorie „**Fühlen**“ in der

Werkzeugpalette und den „**falls dann**“-Block, damit der Helikopter merkt, dass eine Taste gedrückt wird. Dieser „falls dann“-Block muss noch eine wiederhole Schleife, damit er mehr als einmal reagiert.

Nun muss noch in den „falls dann“-Block eingetragen werden, was der



Helikopter tun soll, wenn Du eine Taste gedrückt hast. Verwende dazu die folgenden Blöcke:



Schritt 2, die Hindernisse:

Nun sollst Du die zufällig ins Bild fliegenden Hindernisse programmieren.

Zuerst brauchst du eine Figur, die das Hindernis darstellt. Die kannst du selbst malen oder eine Vorlage aus der Bibliothek nutzen.

Wie sorge ich dafür, dass die Hindernisse ständig ins Bild hinein fliegen?

Der Programmablauf der Hindernisse ist in vier Schritte aufgeteilt:

1. Ein **Klon** des Hindernisses wird erzeugt. Nutze dazu den Block:



2. Der Klon soll sich an den rechten Bildschirmrand bewegen. Nutze dafür die Blöcke:

Wenn ich als Klon entstehe

setze x auf 300

Nun soll das Hindernis aber nicht immer an der selben Stelle des rechten Bildschirmrandes erscheinen deshalb soll die **y-Position zufällig** gewählt werden. Nutze dazu:

Zufallszahl von -180 bis 180

setze y auf 0

3. Der Klon soll nun durch das Bild wandern, bis seine x-Position kleiner als -240 ist. Verwende:

wiederhole bis

<

x-Position

ändere x um -10

4. Der Klon soll sich löschen. Verwende dazu den „lösche diesen Klon“-Block.

So, nun hast du schon das allermeiste und schwierigste geschafft! Sehr gut!

Schritt 3, Helikopter berührt Hindernis:

Schritt drei ist sehr ähnlich zu Schritt eins. Der Helikopter muss wieder etwas fühlen. Und zwar soll der Helikopter fühlen, ob er ein Hindernis berührt. Die Bedingung findest Du in der Kategorie „Fühlen“ und den „falls dann“-Block findest du wieder in der Kategorie Steuerung. Was passieren soll, wenn das Hindernis berührt wird kannst du dir ganz frei überlegen.

Erweiterungen:

1. Ursprungshindernis verstecken mit „versteck dich“-Block und Klone wieder sichtbar machen mit „zeige dich“-Block. Diese Blöcke findest du in der Kategorie „Aussehen“.
2. Hindernisse mit Farbeffekt versehen. Nutze dazu den „ändere Farbeffekt um –“-Block aus der Kategorie „Aussehen“.

„Maus zum Käse“

Ziel des Projekts: In diesem Projekt sollst Du lernen, eine Figur über den Bildschirm zu bewegen. Dazu sollst Du eine Maus über eine Brücke zu einem leckeren Stück Käse steuern.

Das Projekt laden:



Zuerst musst Du ein neues Projekt öffnen, indem Du im **oberen rechten Rand** des Scratch-Editors auf **Datei** und anschließend auf **hochladen von deinem Computer** klickst. Danach klickst du auf die Dateien mit dem Namen „Maus zum Käse“ und dann nur noch auf „OK“ klicken.

Wie programmiert man die Maus?

Zuerst musst du die Maus **anklicken**. Dann brauchst du ein „**Ereignis**“, damit die Maus weiß, wann sie los laufen soll. Ereignisse findest du in der *Werkzeugpalette* unter Ereignisse. Wähle nun den Block „**Wenn grünes Fähnchen angeklickt**“ und ziehe ihn in das weiße Fenster rechts daneben.



Unter diesen **Ereignisblock**, kannst du nun Befehle ziehen, die die Maus ausführt, sobald du das grüne Fähnchen angeklickt wird. Die Befehle die du brauchst sind unter **Bewegung**.

Die wichtigsten Blöcke sind die drei rechts im Bild. Versuche sie so zu kombinieren, dass die Maus mit **einem Mal Anklicken** der grünen Fahne die Brücke überquert.

(Tipp: Du kannst du Schritt auch größer machen, das spart Zeit !)



Viel Erfolg und Spaß! Bei Unklarheiten immer melden!

(Beispielprogramm)

„Pong“

Ziel des Projekts: In diesem Projekt, soll das wohl älteste Computerspiel „Pong“ programmiert werden. In diesem Spiel geht es ähnlich wie beim Tennis darum, den Ball zurück zum Gegner zuschlagen und ihn nicht vorbei zu lassen. Die Programmierung ist in vier Schritte unterteilt.



Schritt 1, Malen:

Öffne zuerst ein neues Projekt. Du brauchst zwei rechteckige Figuren und einen Ball. Hintergrund ist nicht Pflicht.

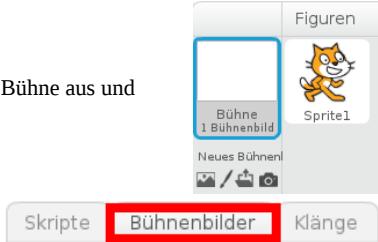
Wie erstelle ich eine neue Figur?



Für eine neue Figur klicke unten links neben der Schrift „Neue Figur“ entweder auf die kleine Figur, um eine Figur zu laden, oder auf den Pinsel, um eine Figur selber zu malen.

Wie erstelle ich einen neuen Hintergrund?

Für einen neuen Hintergrund wähle am unteren linken Rand die Bühne aus und klicke auf den Reiter Bühnenbilder am oberen rechten Rand.



Fragen:

Wie viele Figuren brauchst du für das Spiel?

Antwort:

Brauchst du unbedingt einen Hintergrund?

Antwort:

Schritt 2, die Schläger:

Nun wird programmiert! Die Schläger, also die rechteckigen Figuren am linken und rechten Rand sollen sich nach oben und unten bewegen können.

Wie programmiere ich die Bewegung der Schläger?

Zunächst musst du einen Schläger anklicken und auf „Skripte“ auswählen. Um den Schläger nach oben zu bewegen, muss abgefragt werden, ob die Pfeil-nach-oben-Taste gedrückt ist. Dazu benötigst Du einen „falls dann“-Block aus der Kategorie Steuerung und die „Taste

Pfeil nach oben gedrückt-Bedingung aus der Kategorie Fühlen.

Wenn diese Bedingung erfüllt ist, soll sich die Schläger-Figur sich einen kleinen Schritt nach oben bewegen. Dafür benötigst du den Block „ändere y um 10“, den Du in den „falls dann“-Block einfügen musst. Das Ergebnis sollte etwa so aussehen:



Frage:

Wie lautet die Bedingung, damit sich die Schläger nach oben bewegen?

Antwort:

Dasselbe musst Du nun noch für die Bewegung nach unten programmieren. Ändere dafür den Wert in dem **ändere y um 10** Block auf -10.

Die zwei erstellten „falls dann“-Blöcke müssen nun in einen „**wiederhole fortlaufend**“-Block aus der Kategorie Steuerung und benötigen noch ein **Anfangsereignis**.

Genauso muss nun noch der andere Schläger programmiert werden, diesmal aber mit **anderen Tasten** in den Bedingungen! Wenn du beide Schläger hoch und runter bewegen kannst hast du es geschafft!

Frage:

Wie heißt der Befehl, damit sich die Schläger nach oben bewegen?

Antwort:

Schritt 3, der Ball:

Der Ball soll sich in diesem Spiel kontinuierlich, also die ganze Zeit, bewegen, aber bei Berührung mit Wänden und Schlägern die Richtung ändern.

Wie kann ich die Richtung des Balles ändern?

Mithilfe des Blocks **setze Richtung auf** aus der Kategorie Bewegung, kann die Richtung des Balles gesetzt werden. Am einfachsten ist es die Richtung einfach umzudrehen. Dazu benötigst

du den  Block aus der Kategorie Operatoren, der zwei Zahlen voneinander anziehen kann, und den **Richtung** Block aus der Kategorie Bewegung, der einen Zahlenwert speichert, der für die Bewegungsrichtung des Balles steht. Versuche diese drei Blöcke so zu kombinieren, das die **Richtung** umgedreht wird, also **den negativen Wert von sich selbst** annimmt. Bei Fragen einfach melden!

Frage:

Mit welchem Block änderst du die Richtung des Balls?

Antwort:

Wann soll die Richtung des Balls geändert werden?

Die Richtung des Balls soll geändert werden, wenn eine bestimmte Bedingung erfüllt ist. Diese Bedingung lautet, der Ball berührt einen Schläger oder die Wand. Du kannst für jede dieser Bedingungen einen „falls dann“-Block wie in Schritt eins erstellen. Diese sehen dann in etwa so aus:



In die Bedingung hinein muss die Richtungsänderung des Balles aus vergangenem Absatz.

Frage:

Wann soll der Ball die Richtung ändern?

Antwort:

Wie bewegt sich der Ball?

Der Ball bewegt sich mithilfe eines „**wiederhole-fortlaufend**“-Blocks und des **gehe 10 er-Schritt** Blocks, immer gerade aus. Nach jedem Schritt in dem „wiederhole fortlaufend“-Block müssen die Richtungsänderungen des Balles in dem „falls dann“ Block eingefügt werden.

Also **mit nur einer Bedingung von insgesamt drei** sieht es etwa so aus:



Hier fehlt natürlich noch das Ereignis für den Beginn.

Der Ball benötigt außerdem noch **eine zufällig gewählte Startrichtung**. Dazu benötigst du **Zufallszahlen**, die es in der Kategorie **Operatoren** gibt. Setze damit die Richtung des Balls zu Beginn des Spiels auf einen Wert zwischen -90 und 90.

Alles geschafft? Wenn nicht, dann frag uns!

Zufallszahl von -90 bis 90

Ansonsten, herzlichen Glückwunsch dein Spiel kann jetzt gespielt werden!

Schritt 4, Punkte:

Nun wollen wir noch Punkte zählen. Wer hat wie oft den Ball durchgelassen? Dazu benötigen wir eine Variable. Sozusagen ein Gedächtnis, das sich merkt, wie viele Punkte jeder Spieler hat. Dazu müssen nun zwei Variablen angelegt werden. Klicke dazu auf „**Daten**“ und auf „**neue Variable**“ und lege die Variable „Punkte Spieler 1“ und „Punkte Spieler 2“ an. Mit dem Block  sollen die Punkt zu Beginn des Spiels auf Null gesetzt werden.

Wann und wie wird ein Punkt gezählt?

Die Punkte sollen im Skript des Balls gezählt werden. **Die Position des Balls bestimmt, wann ein Punkt fällt**, denn ist der Ball zu weit rechts, bekommt der linke Spieler einen Punkt, ist der Ball zu weit links, bekommt der rechte Spieler einen Punkt. Die Position des Balls auf der Bühne wird mit seinen „**Koordinaten**“ angegeben. Das ist ein „**y-Wert**“, der die Höhe des Balls angibt und einen „**x-Wert**“ der angibt, wie weit links oder rechts der Ball ist.

Wenn der x-Wert des Balls also besonders groß ist, dann hat der rechte Schläger den Ball vorbeigelassen. Andersherum, wenn der x-Wert besonders klein ist.

Mithilfe des „**Vergleichsoperators**“  aus der Kategorie **Operatoren** und dem  Block aus der Kategorie **Bewegung** kann die gesuchte Bedingung erstellt werden:



Frage:

Wann fällt ein Punkt?

Antwort:

Diese Bedingung muss nun wieder in das sechseckige Feld eines „falls dann“-Blocks wie in den Schritten 2 und 3. Außerdem brauchst du eine zweite Bedingung die überprüft, ob die x-Position **kleiner ist als -214**, für den Fall, dass der zweite Spieler einen Punkt erzielt.

Was soll also getan werden, wenn die x-Position besonders groß oder besonders klein ist? Es soll ein Punkt gezählt werden! Dies geschieht mithilfe des  Blocks aus

der **Kategorie** Daten. Dieser Befehl muss also wieder **in** beide „falls dann“-Blöcke.

Wenn du deine zwei „falls dann“-Blöcke hast, kannst du sie wieder in einen „wiederhole fortlaufend“-Block schieben, damit die Bedingungen ständig überprüft werden.

Gratulation! Bei deinem Spiel werden nun die Punkte gezählt!

Erweiterungen:

1. Überprüfe, ob ein Spieler gewonnenen, also zum Beispiel mehr als drei Punkte hat.

Nützliche Blöcke:  aus der Kategorie Daten,  aus der Kategorie Operatoren, außerdem:



*Dieser Block
stoppt das Spiel*