

Schriftlicher Unterrichtsentwurf am Arbeitsbereich Didaktik der Informatik der WWU Münster¹

Erstellt von:	<input type="text" value="Andrea Borgscheiper"/>
Matrikelnummer:	<input type="text" value="██████"/>
Mastersemester:	<input type="text" value="1"/>
Zeitumfang (min):	<input type="text" value="90"/>
Klasse:	<input type="text" value="4"/>
Thema der Stunde:	<input type="text" value="Wie fühlt und reagiert die Scratch-Katze (Sensoren und Aktoren)?"/>
Thema der Reihe:	<input type="text" value="Programmieren in der Grundschule unter Nutzung der Programmiersprache „Scratch“"/>

¹ Diese Vorlage basiert auf dem Dokument Schriftliche Arbeit mit Kommentar (Stand 03/2013) des Zentrums für schulpraktische Lehrerbildung Krefeld (ZfsL), Seminar für das Lehramt an Gymnasien und Gesamtschulen.

Inhaltsverzeichnis

Schriftliche Planung des Unterrichts	1
1. Ziele und angestrebte Kompetenzen.....	1
2. Didaktische Schwerpunkte	3
3. Artikulationsschema	18
Literaturverzeichnis.....	I
Abbildungsverzeichnis	III
Anhang	IV
I. Stundenverlaufskarten	IV
II. Tafelbild für die Erarbeitung der fachlichen Begriffe mitsamt benötigter Karten	VI
III. Arbeitsheft.....	VIII
IV. Tippkarten.....	XXV
V. Musterlösungen	XXIX
Versicherung.....	XXXI
Verwertungsrechte.....	XXXI

Schriftliche Planung des Unterrichts

1. Ziele und angestrebte Kompetenzen

Ein operationalisiertes Stundenziel/Kernanliegen mit Indikator:

Grobziel:

- Die Schülerinnen und Schüler² sollen mithilfe der Programmiersprache Scratch ein **grundlegendes Verständnis der Begriffe „Sensor“ und „Aktor“** entwickeln, die Bedeutung von **Sensoren und Aktoren im Alltag** herausstellen bzw. reflektieren und ihr erworbenes Wissen auf die **Programmierung von Sensoren und Aktoren in Scratch** anwenden können.

Sie zeigen dies, indem sie

- im Stuhlkreis aktiv eine Definition der genannten Begriffe erarbeiten, deren Bedeutung in der eigenen Lebenswelt mit Mitschülerinnen und Mitschülern diskutieren und im Anschluss daran in den Aufgaben 3, 4 und 5 eigenständig Möglichkeiten entwerfen und erproben Sensoren und Aktoren in Scratch zu programmieren, um abschließend mit eigenen Worten zu begründen, an welcher Stelle Sensoren bzw. Aktoren programmiert wurden und was diese kennzeichnet.

Drei bis fünf operationalisierte Teilziele mit Indikatoren:

Sachkompetenz:

- Die SuS sollen die **Begriffe „Sensor“ und „Aktor“**, sowie deren Bedeutung und Funktionsweise verstehen und erklären können. Sie zeigen dies, indem sie das Zusammenspiel von Sensoren und Aktoren anhand der Nachprogrammierung vorgegebener Spielcodes spielerisch-selbstgesteuert erkunden und die Funktionsweise mit eigenen Worten beschreiben, sowie an Beispielen verdeutlichen. (S1)
- Die SuS sollen Möglichkeiten der **Programmierung von Sensoren und Aktoren in Scratch** kennenlernen und anwenden können. Sie zeigen dies, indem sie durch eigene Programmierung von Programm-Codes die allgemeine Funktionsweise der „Fühlen-Blöcke“ erproben und die Funktion dieser und anderer Blöcke im Bezug auf Sensoren und Aktoren untersuchen und ihren Mitschülerinnen und Mitschülern erklären. (S2)
- Die SuS sollen den **Nutzen und die Bedeutung von Sensoren und Aktoren in ihrem Alltag** erkennen und beschreiben können. Sie zeigen dies, indem sie die Begriffe „Aktor“ und „Sensor“ auf den eignen Körper übertragen und im Austausch mit der Klasse weitere Bezüge zur Lebenswelt erarbeiten. (S3)
- Die SuS entwickeln ein **Programmierverständnis** bzw. erfassen grundlegende Prinzipien der Informationsverarbeitung (Eingabe an den Sensoren → Verarbeitung → Ausgabe an den Aktoren) gemäß EVA-Prinzip, indem sie Blöcke zur algorithmischen Programmierung verwenden und an diesen Sensoren und Aktoren identifizieren. (S4)

Neben diesen Teilzielen werden zusätzlich auch personale, soziale und methodische Kompetenzen gefördert.

² Im Folgenden mit SuS abgekürzt.

Personale und soziale Kompetenz:

- Die SuS fördern ihre **sprachlichen Kompetenzen**, indem sie während der Partnerarbeit oder im Stuhlkreis kommunizieren und gemeinsam Begriffe erarbeiten, von ihren eigenen Erfahrungen und Schwierigkeiten beim Programmieren berichten, Beobachtungen erläutern und Schlussfolgerungen diskutieren. (PS1)
- Die SuS erweitern ihre **Kooperations- und Hilfsbereitschaft**, indem sie sich gegenseitig bei der Programmierung und Erkundung des Themengebietes unterstützen und gemeinsam eine Lösung finden können. (PS2)

Methodische Kompetenz:

Die SuS bauen ihre Kompetenzen im **selbstregulierten Arbeiten** aus, indem sie eigenständig Arbeitsaufträge bearbeiten, sich die ihnen zur Verfügung stehende Zeit einteilen und nach Bedarf geeignete Unterstützungshilfen nutzen. (M1)

Geförderte Kompetenzbereiche:

Im Rahmen der Bearbeitung der Thematik „Sensoren und Aktoren“ wird fächerübergreifend gearbeitet, wodurch Kompetenzen aus den Bereichen Informatik, Mathematik und ansatzweise auch des Sachunterrichts ausgebaut werden. Im Rahmen der dargestellten Unterrichtsstunde werden jedoch schwerpunktmäßig die im Folgenden genannten Kompetenzbereiche gefördert. Auf weitere Kompetenzen, die ebenfalls ansatzweise gefördert werden, wird an späterer Stelle verwiesen.

Informatik [GI18]:

- *Inhaltsbereich:* Informatiksysteme
- *Prozessbereich:* Begründen und Bewerten

Mathematik [MSW08]:

- *Prozessbezogene Kompetenz:* Problemlösen/ kreativ sein

Hierdurch sollen folgende Kompetenzen gefördert werden:

Im Folgenden soll nun erläutert werden, welche konkreten Kompetenzen aus den oben aufgeführten Kompetenzbereichen in der dargestellten Unterrichtsstunde gefördert werden sollen. Hierbei sind alle aufgeführten Kompetenzen solche, welche die SuS zu Ende der vierten Klasse erreicht haben sollten.

Inhaltsbezogene Kompetenz: Informatiksysteme [GI18]:

- „Die SuS geben grundlegende, allgemeingültige Beschreibungen der Funktion und Arbeitsweise [hier: Funktionsweise von Sensoren und Aktoren] von Informatiksystemen an.“ [GI18, S.14]
- „Die SuS wenden das EVA-Prinzip [hier: im Sinne von Sensoren, die die Eingabe empfangen und ein Signal an die Aktoren weiterleiten, an denen die Ausgabe erfolgt] auf Informatiksysteme an“ [GI18, S.14]

Prozessbezogene Kompetenz: Begründen und Bewerten [GI18]:

- Während erster Erprobungen und Betrachtungen von Alltagssituationen und Programmier-Codes in Scratch äußern sich die SuS begründet über den informatischen Zusammenhang von Sensoren, Aktoren und Falls-Dann-Befehlen. Zudem erklären sie informa-

tisch-algorithmische „Beziehungen und Gesetzmäßigkeiten auf unterschiedlichen Ebenen – mit ihren eigenen Worten – zunehmend auch unter Verwendung der Fachsprache“ [Gl18, S.8].

Prozessbezogene Kompetenz: Problemlösen/ kreativ sein [MSW08]:

- Die SuS entnehmen die für die Programmierung relevanten Informationen den Aufgabenstellungen (erschließen) [MSW08].
- „Die SuS probieren zunehmend systematisch und zielorientiert und nutzen die Einsicht in Zusammenhänge zur Problemlösung (lösen)“ [MSW08, S.59].
- Die SuS überprüfen ihre Programmier-Codes hinsichtlich der Durchführbarkeit, suchen ggf. nach Fehlern, korrigieren diese und vergleichen ihre verschiedenen Lösungswege (reflektieren und überprüfen) [MSW08].
- Die SuS übertragen ihre gewonnen Erkenntnisse über Sensoren und Aktoren, sowie die Vorgehensweise bei der Programmierung dieser auf ähnliche Sachverhalte (übertragen) [MSW08].

2. Didaktische Schwerpunkte

Ausgangspunkt einer jeder Unterrichtsplanung sind eine Analyse der Lernausgangslage der SuS, die Sachanalyse des Unterrichtsgegenstandes, sowie eine didaktische und methodische Analyse. Durch sie werden getroffene Ziel-, Inhalts- und Methodenentscheidungen der Unterrichtsplanung hinsichtlich Rahmenbedingungen, Zielgruppe, Lerngegenstand und methodischer Umsetzung untersucht und reflektiert. Zu beachten bleibt dabei, dass sich gemäß dem Modell der didaktischen Rekonstruktion nach Kattmann [Ka07] die Analyseaspekte gegenseitig beeinflussen, sodass beispielsweise die methodischen Entscheidungen unter einem ständigen Rückbezug und Wechselspiel von fachlicher Klärung und Lernerperspektive getroffen werden. Nachfolgend sollen die genannten Analysen anhand einer imaginären vierten Klasse durchgeführt werden, für die die dargelegte Unterrichtsstunde geplant wurde. Hierzu werden zunächst die Lernausgangslage der SuS, sowie notwendige organisatorische Voraussetzungen skizziert. Anschließend wird in einer Sachanalyse der informatisch-fachliche Hintergrund der Programmiersprache Scratch und der im Unterricht behandelten Themenbereiche unter Berücksichtigung didaktischer Reduktion dargelegt. An didaktische und methodische Analyse schließt sich zuletzt die Unterrichtsplanung mit dem Artikulationsschema der geplanten Unterrichtsstunde an.

Die Unterrichtsstunde ist aufgrund ihrer thematischen Komplexität und motorischen Anforderungen thematisch und methodisch auf die Durchführung in einer vierten Klasse ausgerichtet. Hierzu werden auf schriftsprachlicher Ebene grundlegende Fähigkeiten in Schrift und Sprache vorausgesetzt, die ein Lesen und Bearbeiten von Arbeitsaufträgen ermöglichen. Eingegliedert wird die geplante Doppelstunde in eine fächerübergreifende Projekt-Reihe, die das Programmieren mit Scratch thematisiert und über einen Zeitraum von sechs bis sieben Wochen einmal pro Woche stattfindet. Die dargelegte Doppelstunde ist hierbei die fünfte Einheit, sodass bereits einige Kompetenzen im Umgang mit Scratch bei den SuS vorausgesetzt werden können. Erforderlich für das Programmieren mit Scratch sind zunächst ausreichende motorische Fähigkeiten (z.B. für die Handhabung der Computermouse), sowie Kenntnisse im Umgang mit dem Computer. Durch die Verortung

**Lehr- und
Lernausgangslage
der SuS**

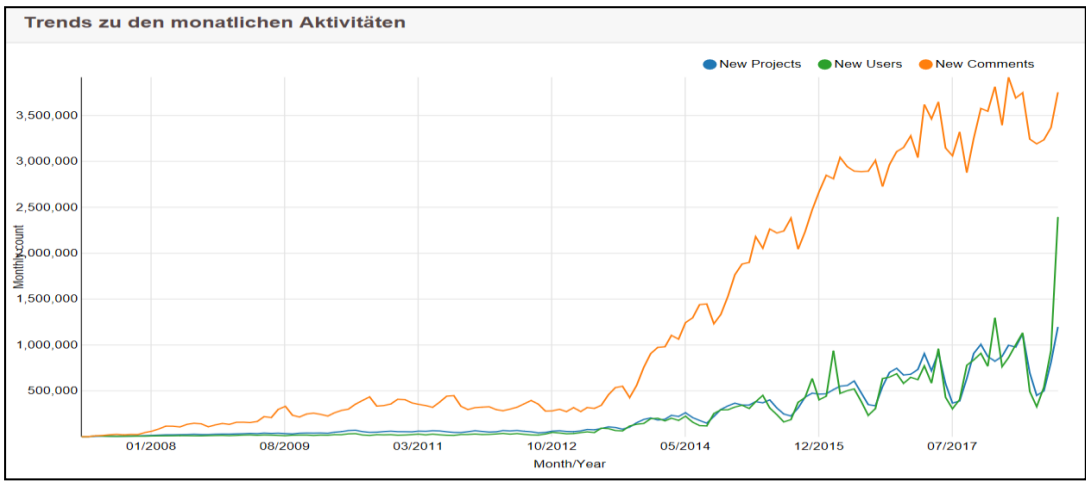
der Unterrichtseinheit in einer vierten Klasse und die bereits vorausgegangenen Unterrichtsstunden zu dieser Projekt-Reihe wird vorausgesetzt, dass die SuS mit dem Arbeiten am Computer vertraut sind und somit über grundlegende Kompetenzen (wie Starten und Herunterfahren des Computers, Ausführung von Programmen, Bedienen der Maus und Tastatur, Öffnen von Dateien) verfügen. In den vorangegangenen Unterrichtsstunden der Projekt-Reihe, haben die SuS bereits gelernt, wie Scratch gestartet wird, haben einen eigenen Zugang erhalten und die meisten Codeblockbereiche (Bewegung, Aussehen, Klang, Ereignisse und Steuerung), sowie die zugehörigen Blöcke kennengelernt und erprobt. Somit wird vorausgesetzt, dass die SuS bereits mit der Bedienung und Programmierung von Scratch (Erstellung von Figuren und Hintergrund, Hochladen und Speichern von Programm-Codes, Bedienfelder, Zusammenfügen von Codeblöcken) vertraut sind und selbstständig und relativ sicher eigene Programm-Codes entwickeln können. Zudem wurde der Begriff des Programmierens bereits mit den SuS erarbeitet. Erfahrungen mit dem Codeblockbereich „Fühlen“ besitzen die SuS zu Beginn der dargelegten Doppelstunde noch nicht.

Aus methodischer Perspektive heraus wird vorausgesetzt, dass die SuS mit dem Konzept des selbstregulierten Lernens, als „[...]eine Form des Erwerbs von Wissen und Kompetenzen, bei der Lerner sich selbstständig und eigenmotiviert Ziele setzen sowie eigenständig Strategien auswählen, die zur Erreichung dieser Ziele führen [...]“ [GN17, S.146], vertraut sind. Dies beinhaltet auch, dass den SuS bewusst ist, dass ein Lernprozess eine Planungs-, Durchführungs- und Reflexionsphase umfasst und komplexe Aufgaben somit in drei Schritten bearbeitet werden müssen [ebd.]. Darüber hinausgehend sind die SuS mit den Sozialformen der Einzel- und Partnerarbeit, sowie den hieran geknüpften Verhaltensweisen und Regeln vertraut. Um der Heterogenität der Klassengemeinschaft und den damit verbundenen unterschiedlichen Lernvoraussetzungen gerecht zu werden, ist die Klasse bereits mit dem Helfersystem, das auch in dieser Stunde Anwendung findet, vertraut. Aus verschiedenen Fächern wissen die SuS, dass sie, wenn sie bei der Bearbeitung einer Aufgabe nicht weiter kommen, Tippkarten am Lehrerpult finden, die ihnen weiterhelfen können. Alternativ oder zusätzlich können die Kinder Hilfe bei ihren Mitschülerinnen und Mitschülern holen, indem sie entweder ihren Partner um Rat fragen (während der Partnerarbeit) oder eine/n der SuS um Hilfe bitten, der die Aufgabe bereits bearbeitet hat und sein Namensschild an die Tafel gehangen hat (Helfersystem). Darüber hinausgehend wird vorausgesetzt, dass den SuS die Symbole und der Aufbau (z.B. Kompetenzcheck) des Scratch-Arbeitsheftes bereits erläutert wurden. Aufgrund der Programmierung erster kleiner Spiele und der Nutzung des Computers als Medium, sowie des hiermit einhergehenden Bezugs zum Alltag der Kinder wird eine hohe Leistungsbereitschaft der SuS erwartet [Sc15].

Neben den Vorerfahrungen und individuellen Voraussetzungen der SuS sind auch einige organisatorische Voraussetzungen zu beachten. Um den geplanten Unterricht durchführen zu können, sind gewisse räumliche Bedingungen und materielle Ausstattungen des Klassenraums nötig. So muss die Klasse mit mehreren Computern ausgestattet sein (mindestens ein Computer pro Partnerarbeitsgruppe), ein Internetzugang vorhanden bzw. die Scratch Software heruntergeladen worden sein, sowie ein Beamer zur Verfügung stehen. Darüber hinaus muss ein Scratch-Arbeitsheft für jedes Kinder bereitgestellt werden.

Die Programmiersprache Scratch:

Scratch wurde erstmals 2007 als Projekt der Lifelong-Kindergarten-Group am Media-Lab des MIT (Massachusetts Institute of Technology) veröffentlicht [LK19]. Es handelt sich hierbei um eine Programmiersprache inklusive ihrer Entwicklungsumgebung, welche entwickelt wurde, „[...] um einen leichten Einstieg in die Programmierung zu ermöglichen und dabei der Kreativität freien Lauf zu lassen“ [Ba17, S.17]. Als visuelle Programmiersprache können nahezu alle Elemente aus denen ein Scratch-Projekt besteht per Drag & Drop zusammengebaut werden [ebd.]. Zudem werden die Elemente durch intuitiv verständliche grafische Darstellungen (z.B. Programmierbefehle als Bausteinbilder oder Klänge durch das Bild eines Lautsprechers) repräsentiert, was die Nutzung bereits in der Grundschule ermöglicht. Wollen die SuS Bewegungen, Animationen oder Spiele in Scratch programmieren, müssen sie einen Algorithmus, also eine entsprechende Handlungsvorschrift erstellen [Sw17]. Die Darstellung entsprechend zu konstruierender und programmierender Algorithmen wird in Scratch wenig abstrakt mithilfe einfacher und ikonischer Codeblöcke abgebildet, welche ähnlich dem Prinzip von LEGO-Steinen, im Skriptbereich zusammengebaut werden können. Die verschiedenen algorithmischen Codeblöcke steuern hierbei die Bewegung, das Aussehen und das Fühlen einer Figur bzw. eines Gegenstandes, das Erzeugen bestimmter Klänge, das Einfügen weiterer Figuren, die Bedingungen, wann ein Ereignis eintritt, und die Kommunikation der Figuren untereinander. Auf diese Weise können ohne Tippen zu müssen, allein mit der Maus sowohl kleinere als auch größere Spiele und Animationen erstellt werden [Sw17]. Unter dem Motto „imagine, program, share“ regt Scratch zu einem kreativen und spielerischen Zugang zu digitaler Technik, sowie zu aktiven eigenen Programmierungen an, welche mit anderen Scratchern der Online-Gemeinschaft geteilt werden können [ebd.]. Obwohl die eigentliche Zielgruppe der kostenlosen Programmiersprache 8- bis 16 Jährige Kinder und Jugendliche sind, wird Scratch mittlerweile von Menschen jeder Altersgruppe Zuhause, an Schulen, Museen, Bibliotheken oder Gemeindezentren genutzt [LK19]. Vornehmlich richtet sich Scratch jedoch an SuS aller Stufen (von der Grundschule bis zur Hochschule) und aller Fächer (z.B. Mathematik, Informatik, Sachunterricht), sodass es bereits vielfältige Materialpakete und Umsetzungsideen gibt (vgl. beispielsweise BBC17) [LK19]. Folglich ist Scratch sowohl von der Zielgruppe, als auch von der Zielstellung und der Komplexität für das Programmieren mit einer vierten Klasse gut geeignet. Die Möglichkeit zwischen über 40 Sprachen zu wählen, sowie die einfache Handhabung und Benutzeroberfläche führen dazu, dass Scratch immer mehr genutzt (vgl. Abb.1) und in über 150 verschiedenen Ländern verwendet wird [ebd.].



(Abb. 1: Trends zu den monatlichen Aktivitäten in Scratch [LK19])

Überdies wurden Forschungen zur Wirksamkeit von Scratch insbesondere durch das MIT selbst vorgenommen. Im Rahmen des sogenannten Clubhouse-Projektes, in welchem Kinder unter anderen Medien auch Scratch nutzen konnten, wurden sehr positive Einflüsse der eigentätigen und spielerischen Auseinandersetzung, sowie der Abwesenheit programmierfähiger Mentoren auf das Erlernen grundlegender Konzepte der Programmierung nachgewiesen [Ma08]. Darüber hinaus zeigte sich, dass die Kinder Scratch am häufigsten nutzten und sich ohne Motivation von außen teilweise über ein Jahr mit der Erstellung eines Scratch-Projektes beschäftigten, obwohl ihnen viele alternative Programme zur Mediengestaltung zur Verfügung standen [ebd.]. Diese Motivation und auch der Ausbau elementarer Konzepte des Programmierens spiegeln das Potenzial von Scratch für die Grundschule und auch das selbstregulierte Arbeiten wieder. So werden für die unterrichtspraktische Verwendung auch im deutschsprachigen Raum, wie beispielsweise im Rahmen des Projektes *InfoSphere – Schülerlabor Informatik* an der RWTH Aachen [vgl. Ha19], vielfältige Scratch-Unterrichtsmaterialien entwickelt und in der Praxis erprobt.

Die während der hier dargestellten Unterrichtseinheit verwendete Programmiersprache Scratch fördert und schult durch ihre Anwendung seitens der SuS ein intuitives Verständnis für verschiedene informatische Themenbereiche, sowie prozessbezogene Kompetenzen, wie das kreative Denken und systematische Vorgehen, das selbstständige Gestalten von Projekten, die Zusammenarbeit und Kommunikation mit anderen SuS und das Entwickeln von Problemlösestrategien [BBC17]. Es werden entsprechend zentrale Schlüsselkonzepte der Informatik (z.B. Ereignisse, Bedingungen, Daten, Schleifen) und Schlüsseltechniken (z.B. experimentieren und wiederholen, austesten und Fehler beheben) gefördert [ebd.]. So stellt Scratch durch die Möglichkeit des kreativen Programmierens von einfachen bis zu komplexen Projekten lernergerecht zentrale Funktionsweisen, wie das Programmieren von algorithmischen Programmen oder das Steuern von Ein- und Ausgaben (an Sensoren und Aktoren) in den Vordergrund. Durch den Einsatz von Scratch im Unterricht ist somit die Behandlung unterschiedlicher informatischer Themenbereiche möglich. Im Folgenden soll der in der dargelegten Doppelstunde thematisierte Themenbereich der Sensoren und Aktoren mit Transfer auf die Programmiersprache Scratch kurz dargestellt werden.

Fachlicher Kontext: Sensoren und Aktoren:

Sensoren können definiert werden als „[...] Messwertaufnehmer für nichtelektronische Größen; sie bilden eine nichtelektrische Messgröße, die an ihrem Eingang anliegt, auf eine elektrische Größe an ihrem Ausgang ab“ [RW16, S.6]. Sie sind folglich „Fühler“ [FH11, S.803] bzw. Bauelemente, die an ihrem Eingang eine nichtelektrische physikalische oder chemische Größe (z.B. Temperatur, Helligkeit, Druck, chemische Konzentration oder Entfernung in Metern) messen bzw. aufnehmen und diese in eine elektrische Größe (z.B. elektrische Spannung, elektrischen Strom oder eine elektrische Ladung) am Sensorausgang umwandeln [RW16]. Je nachdem, welche nichtelektronischen Größen die Sensoren messen, unterscheidet man beispielsweise zwischen Temperatursensoren, Kraftsensoren, Bewegungssensoren und optischen Sensoren. Für die Aufbereitung bzw. Weiterverarbeitung des Sensorsignals und zur Präsentation der Messdaten werden elektronische Schaltungen, Mikrocontroller und geeignete Algorithmen eingesetzt [ebd.]. Die Scratch-Katze oder auch andere Figuren und Gegenstände in Scratch besitzen in gewissem Maße ebenfalls Sensoren, da sie nichtelektronische Größen, wie Entfernung-

gen/Bewegungen in Metern oder Berührungen (Berührungssensor) messen und darauf reagieren können. Natürlich sind die Sensoren aber begrenzt, sodass die Figuren beispielsweise keine optischen oder akustischen Sensoren besitzen. Haben die Sensoren einseitig einen bestimmten Zustand bzw. eine Zustandsänderung einer Größe erfasst und den Eingangswert einem informationsverarbeitenden System zugeführt, wird nach der Verrechnung der Messdaten (Istwert) und Berücksichtigung der Vorgaben (Sollwert) ein Steuersignal an die ausgangsseitig angeordneten Aktoren weitergeleitet [RW16]. **Aktoren** sind „Macher“ [FH11, S.34] und bilden das Gegenstück zu den Sensoren, denn sie „[...] setzen eine elektrische Eingangsgröße in eine nichtelektrische Ausgangsgröße um und dienen so als Stellglied oder Signalwandler“ [RW16, S.13]. Durch erhaltene Steuersignale wandeln die Aktoren die elektrischen Signale in sichtbare oder hörbare Wiedergabe (z.B. mechanische Bewegungen) um und zeigen auf diese Weise eine Reaktion auf einen vom Sensor wahrgenommenen Zustand bzw. einer Zustandsänderung [ebd.]. Die Scratch-Katze bzw. alle Figuren und Objekte in Scratch besitzen ebenfalls Aktoren. Für die Umwandlung der Signale von den Sensoren in Bewegungen, Klänge oder Änderungen des Aussehens, stehen ein Skriptbereich und Lautsprecher am Computer zur Verfügung. Programmieren die SuS durch das Zusammenfügen algorithmischer Bausteine Sensoren und Aktoren in Scratch, lernen sie gleichzeitig auch ein wichtiges informatisches Prinzip kennen: das **EVA-Prinzip**. Der eingegebene Algorithmus stellt eine Reihung von Rechenschritten dar, mithilfe derer eine bestimmte Eingabe in eine Ausgabe umgewandelt wird. Der Algorithmus enthält somit gemäß dem EVA-Prinzip Informationen über die Art der Eingabe (die das Gerät erreichende Information), die Art der Verarbeitung (Ausführen der Vorgaben des Algorithmus und Generation neuer Informationen), sowie die Art der Ausgabe (Darstellung der generierten Information über verschiedene Aktoren) [Wb17]. Wird das EVA-Prinzip nun auf Sensoren und Aktoren bezogen, liefern die Sensoren die Eingaben (in Form von gemessenen Größen die in elektrische Signale umgewandelt werden), welche von den Programmen der SuS zu Ausgabedaten verarbeitet werden. Diese Ausgabedaten steuern wiederum die Aktoren. Wichtig ist, dass die Programme der SuS eine Abfolge von Algorithmen (einzelnen Bausteinen) bestehend aus Befehlen und Wiederholungen enthalten, sodass mittels Sensorik unterschiedliche Aktionsverläufe möglich sind. Durch die Programme können somit sowohl einfachere („Wenn-dann-Beziehungen“) als auch komplexere („Wenn-Dann-Sonst-Beziehungen“) Abläufe beschrieben und generiert werden. Überträgt man die Funktionsweise von Sensoren und Aktoren in einer stark vereinfachten Analogie auf den menschlichen Körper, kann man „[...] die mechanische Grundstruktur des Systems mit dem Skelett, die Sensoren mit den Sinnesorganen, die Aktoren mit der Muskulatur und die Informationsverarbeitung mit dem zentralen Nervensystem, einschließlich dem Gehirn, vergleichen“ [Wa06, S.670].

Didaktische Reduktion und Schwerpunktsetzung:

Aufgrund der begrenzten zur Verfügung stehenden Zeit im Unterricht und der Komplexität der Thematik liegt der Schwerpunkt der dargelegten Doppelstunde auf der Erarbeitung der Begriffe „Sensor“ und „Aktor“ (Erklären und Anwenden der Begriffe, sowie deren Funktionsweise), sowie deren Programmierung in Scratch (Programmieren von „falls-dann-Algorithmen“ anhand ausgewählter Sensoren aus dem Blockbereich „Fühlen“ und Aktoren aus den Blockbereichen „Bewegung“, „Aussehen“ und „Klang“). Als Maßnahme der didaktischen Reduktion werden während des gesamten Scratch-Projektes nur ausgewählte Code-Blöcke aus den einzelnen Blockbereichen angewandt (diejenigen die im Arbeitsheft zu Beginn unter

„Blöcke in Scratch“ angeführt sind), da die hohe Anzahl bzw. Auswahl an Blöcken die SuS ansonsten überfordern und die Programmierzeit enorm verlängern könnte. Durch die Einschränkung der zur Verfügung stehenden Blöcke, entwickeln die Kinder über die Dauer des Projektes Routine im Umgang mit den Blöcken, die sie nutzen. Darüber hinaus führt die Einschränkung auch dazu, dass die SuS in der dargelegten Doppelstunde nur vier Blöcke aus dem Bereich „Fühlen“ und ggf. einzelne Blöcke aus dem Bereich Steuerung („Falls-dann“ und „Falls-Dann-Sonst“) neu kennen und anwenden lernen. Damit die SuS sich auf die Programmierung von Sensoren und Aktoren konzentrieren können, werden zusätzlich komplizierte Bühnen (Labyrinth und Pong) vorbereitet zur Verfügung gestellt.

Darüber hinaus werden in der Doppelstunde nur exemplarisch für alle anderen Sensoren und Aktoren, die Sensoren und Aktoren in Scratch, sowie die am eigenen Körper thematisiert. Dies rührt daher, dass der eigene Körper für die Kinder ein sehr anschauliches und gut nachvollziehbares Beispiel aus ihrem Alltag darstellt und er sich gut auf eine Figur in Scratch übertragen lässt. Des Weiteren messen viele andere Sensoren Größen, die den SuS noch nicht vertraut oder ihnen nicht sehr zugänglich sind (z.B. Drucksensoren). Durch die Diskussion über weitere Sensoren und Aktoren im Alltag wird dennoch schon ein erster Transfer auf weitere technische Geräte angebahnt (z.B. Thermometer oder Mikrofon als Sensor, Glühlampe oder Motor als Aktor). Überdies wird zwecks didaktischer Reduktion das EVA-Prinzip nur in ersten Ansätzen thematisiert, da der Schwerpunkt der Unterrichtsstunde auf dem Verständnis von Sensoren und Aktoren liegen soll. Weil diese aber eng mit dem EVA-Prinzip verknüpft sind und dieses ein grundlegendes informatisches Prinzip darstellt, sollte es dennoch angeführt werden. Bei den Erläuterungen der Funktionsweise der Sensoren und Aktoren in den in Aufgabe 3 und 4 eigenständig erzeugten Programm-Codes der SuS sollte demnach vornehmlich auf die richtige Erläuterung von Sensoren und Aktoren und nebensächlich auf den Einbezug des EVA-Prinzips geachtet werden. Auf diese Weise kann gemäß dem Prinzip der natürlichen Differenzierung sowohl eine Förderung der leistungsstärkeren, als auch eine Vermeidung der Überforderung leistungsschwächerer SuS stattfinden. Zudem können die SuS selbst entscheiden, ob ihnen das EVA-Prinzip bei der Erläuterung der Funktionsweise von Sensoren und Aktoren hilft oder nicht. Worauf allerdings schon Wert gelegt werden sollte, ist das Verständnis dafür, dass die Verarbeitung des Signals vom Sensor durch die, von den SuS entworfenen Programme erfolgt, die schließlich einen Befehl an den Aktor weiterleiten.

Bezüglich der Thematik der Sensoren und Aktoren soll nur die allgemeine Funktionsweise, also die Funktion der Sensoren als Fühler und Messer von Änderungen gewisser Größen in der Umgebung und die Funktion von Aktoren als Reaktoren auf die von den Sensoren vermittelten Signale, erarbeitet werden. Die Sensoren und Aktoren werden dabei als „black boxes“ behandelt, da ihr interner Aufbau sehr komplex ist, die Funktionsweise sich auch ohne diesen internen Aufbau erläutern lässt und die Erläuterung des internen Aufbaus mit der Einführung vieler weiterer Fachbegriffe verbunden wäre. Darüber hinaus würde die Zeit einer Doppelstunde nicht ausreichen, um zusätzlich noch den Aufbau der Sensoren und Aktoren zu erläutern. Zwecks didaktischer Reduktion soll ebenfalls auf ein tiefergehendes Eingehen der Signalumwandlung verzichtet werden. Grund hierfür ist, dass die Signalumwandlung von einer nichtelektrischen in eine elektrische Größe am Sensor und die umgekehrte Umwandlung am Aktor für Kinder nicht sichtbar ist. Eine Thematisierung der Signalverarbeitung würde somit das Verständnis der Kinder für die grundlegende Funktionsweise von Sensoren und Aktoren nur erschweren oder möglicherweise verhindern.

Die somit in der Auseinandersetzung mit Sensoren und Aktoren am eigenen Körper und in Scratch gewonnen exemplarischen Kompetenzen können in den weiterführenden Schulen ausgebaut werden und ermöglichen zudem durch Reflexion der im Programmieren erworbenen Erfahrungen die Förderung informatischer Mündigkeit.

Durch die aktive Teilhabe am Unterrichtsgeschehen erwerben die SuS Kompetenzen, die einen wichtigen Beitrag zur informatischen, mathematischen und sachunterrichtlichen Bildung leisten. Insgesamt erfolgt die Aneignung des Verständnisses von Sensoren und Aktoren, sowie deren Programmierung mit dem Informatiksystem Scratch spielerisch-entdeckend, beispielsweise durch das „[...] Ausprobieren und Beobachten, wie ein gegebenes Informatiksystem auf unterschiedliche Aktionen und Eingaben reagiert [...]“ [Gl18, S. 7]. Durch von der Lehrkraft moderierte Lernprozesse können die SuS neben dem spielerisch-entdeckenden Verhalten zu systematischen Beobachtungen und adäquaten Schlussfolgerungen angeregt werden [ebd.]. Die SuS erwerben bzw. erweitern somit ihre Kompetenzen sowohl auf inhaltlicher als auch prozessbezogener Ebene, sodass der Lernzuwachs mit den curricularen Vorgaben des Lehrplans NRW für Grundschulen [MSW08], sowie den Bildungsstandards Informatik für den Primarbereich [Gl18] einhergeht. Im Folgenden wird genauer erörtert, wie die zu Beginn der Arbeit genannten Kompetenzen in der dargelegten Unterrichtseinheit gefördert und ausgebaut werden können.

**Legitimation
des Vorhabens
durch
curriculare
Vorgaben**

Informatische Kompetenzen:

Nach Erprobung der Spiele und Animationen in Aufgabe 1 werden mit den SuS gemeinsam die Begriffe „Sensor“ und „Aktor“, sowie deren Funktionsweise unter Berücksichtigung des EVA-Prinzips erarbeitet, verbalisiert und visualisiert. Anschließend sollen die SuS in den Aufgaben 3,4 und 5 die Funktionsweise von Sensoren und Aktoren, als eine wichtige Funktionsweise des Informatiksystems Scratch, altersgerecht und unter Verwendung der Fachsprache beschreiben. Durch die Beschreibung der Begrifflichkeiten und Funktionsweisen mit eigenen Worten wird die Kompetenz „Die SuS geben grundlegende, allgemeingültige Beschreibungen der Funktion und Arbeitsweise [hier: Funktionsweise von Sensoren und Aktoren] von Informatiksystemen an.“ [Gl18, S.14] gefördert, da sich die SuS aktiv mit den Begriffen auseinandersetzen müssen, um deren Funktionsweise fachlich korrekt zu beschreiben. Zudem wenden sie in den Aufgaben 3,4, und 5 in ihren Programmierungen und Beschreibungen die Begrifflichkeiten „Sensor“ und „Aktor“, sowie das EVA-Prinzip auf ihre eigenen Scratch-Programme und im zweiten Teil von Aufgabe 2 auch auf ihren eigenen Körper an. Hierdurch wird die Kompetenz „Die SuS wenden das EVA-Prinzip [hier: im Sinne von Sensoren die die Eingabe empfangen und ein Signal an die Aktoren weiterleiten, an denen die Ausgabe erfolgt] auf Informatiksysteme an“ [Gl18, S.14] gefördert, da die gelernten Prinzipien und Funktionsweisen auch transferiert und angewandt werden müssen. Insgesamt leistet die Stunde also, wie bereits zuvor ausgewiesen, einen Beitrag zu Kompetenzen des **Inhaltsbereichs „Informatiksysteme“** [ebd.], da die SuS lernen grundlegende Funktionsweisen eines Informatiksystems zu beschreiben und das Informatiksystem Scratch effizient und zielgerichtet zu nutzen. Über diesen Inhaltsbereich hinausgehend werden durch die systematische Programmierung in Ansätzen auch Kompetenzen des Inhaltsbereich **„Algorithmen“** ausgebaut [ebd.]. Auch wenn die SuS noch nicht den Begriff „Algorithmus“ kennen, können sie einzelne Algorithmen aus Scratch (Blöcke) benennen bzw. als Handlungsvorschriften beschreiben und in den

Aufgaben 3, 4 und 5 unter Rückbezug auf Grundbausteine in Form von Code-Blöcken selbst Algorithmen realisieren, indem sie eigene algorithmische Programme entwickeln und diese anschließend erproben und ggf. überarbeiten. Zudem wird während der ganzen Unterrichtseinheit immer wieder (z.B. in der Reflexion nach Aufgabe 3 und ggf. 4) von den SuS gefordert, verschiedene eigene Programme (bestehend aus Algorithmen) unter Verwendung der erlernten fachsprachlichen Begriffe zu vergleichen [ebd.]. Somit lernen sie in der dargelegten Unterrichtseinheit bzw. während des ganzen Scratch-Projektes ihre Kompetenzen im Programmieren eines Informatiksystems auszubauen.

Darüber hinaus werden auch mehrere **Prozessbereiche** gefördert. Indem die SuS zu Beginn der Doppelstunde eine Sachsituation (hier: Sensoren und Aktoren beim Fangen) nachstellen und erfassen, dazu in Aufgabe 3 einen informatischen Programmier-Code erstellen und diesen in der anschließenden Reflexion wieder auf die Sachsituation beziehen und so ihre informatische Modellierung reflektieren, werden erste Kompetenzen im **„Modellieren und Implementieren“** geschult [Gl18]. Zentral ausgebaut werden jedoch die Kompetenzen im **„Begründen und Bewerten“** [Gl18]. Am Anfang der Sitzung wird mit den SuS gemeinsam eine Forschungsfrage für den Unterricht erarbeitet („Wie fühlt und reagiert die Scratch-Katze?“). So wird schon zu Beginn der Stunde das Begründen dahingehend geschult, dass die SuS sich in Anschluss an Aufgabe 1 zunächst unter Verwendung von Alltagssprache begründet über die beobachteten Zusammenhänge zwischen den Programmier-Codes und dem „Fühlen“ und „Handeln“ der Scratch-Katze bzw. anderen Objekten äußern sollen. Indem die SuS im Anschluss daran während ihrer Erprobungen und Programmierungen von Sensoren und Aktoren in Scratch (Aufgabe 3,4 und 5), sowie den anschließenden Reflexionen aufgefordert werden, informatische Zusammenhänge und Beziehungen zwischen Sensoren, Aktoren, Falls-Dann-Beziehungen und dem EVA-Prinzip „[...] mit ihren eigenen Worten – zunehmend auch unter Verwendung der Fachsprache“ [Gl18, S.8] zu erklären, werden sie geschult komplexere Beziehungen und Gesetzmäßigkeiten zu erklären und begründen. Auf diese Weise wird die zu Beginn formulierte Kompetenz im Bereich des „Begründen und Bewerten“ gefördert, da SuS darin geschult werden mündlich und schriftlich auf unterschiedlichen Komplexitätsniveaus informatische Beziehungen zu erläutern. Durch diesen zunehmend auf Fachsprache basierenden Austausch und die gegenseitige Kooperation bei der Bearbeitung informatischer Problemstellungen in Partnerarbeit (Aufgaben 3,4 und 5) wird durch den Unterrichtsentswurf zusätzlich die Kompetenz des **„Kommunizieren und Kooperieren“** gefördert [Gl18]. Die SuS tauschen sich sowohl mit ihrem Partner, als auch im Klassengespräch in der Umgangssprache und zunehmend auch unter Verwendung fachspezifischer Begriffe über ihre Vorgehensweisen bei Programmieren und die informatische Beziehung von Sensoren, Aktoren und dem EVA-Prinzip aus (ebd.). Dies ist auch gemäß der Kultusministerkonferenz wünschenswert, da die SuS durch das Schulen von Kooperation, Kommunikation und Zusammenarbeit aktiv an der Gesellschaft teilnehmen, sowie „Medienerfahrungen weitergeben und in kommunikative Prozesse einbringen“ [KM16, S.11].

Mathematische Kompetenzen:

Im Rahmen des Lehrplans Mathematik für die Grundschule wird in der dargelegten Doppelstunde inhaltlich keine Kompetenz explizit gefördert. Dennoch kann der Unterrichtsentswurf dem **Inhaltsbereich „Größen und Messen“** mit dem Schwerpunkt „Größenvorstellung und Umgang mit Größen“ zugeordnet werden, da Sensoren Messgeräte für verschiedene Größen darstellen. In der geplanten Unterrichts-

stunde wird zudem durch die Diskussion der Funktionsweise von Sensoren in Scratch und weiterer Sensoren im Alltag der Kinder (z.B. Temperaturregler) das Verständnis von Sensoren als Messgeräte für gewisse Größen und Eigenschaften angebahnt. Zwar werden in der geplanten Unterrichtsstunde nicht explizit Größen gemessen, aber es können gerade im Anschluss an diese Unterrichtsstunde im Mathematikunterricht verschiedene Sensoren, sowie die Größen, die sie messen, genauer untersucht, verglichen und gemessen werden.

Der zentrale **prozessbezogene Kompetenzbereich**, der in dieser Unterrichtsstunde gefördert wird, ist der Kompetenzbereich **„Problemlösen/ kreativ sein“** [MSW08]. Die Kompetenz „Die SuS entnehmen den Aufgabenstellungen die für die Programmierung relevanten Informationen (erschließen)“ [Formulierung angelehnt an MSW08], wird zu Beginn der Aufgaben 2 und 3 gefördert, da die SuS einer gegebenen Problemstellung (Wie fühlt und reagiert die Scratch-Katze?) nachgehen, indem sie den Aufgabenstellungen die für die Programmierung relevanten Informationen entnehmen. Zum Teil wird sogar auf den Tippkarten darauf verwiesen, die für die Programmierung relevanten Informationen zu unterstreichen, um sich daraufhin besser geeignete Programmier-Codes überlegen zu können. Nachdem die Problemstellung erschlossen ist, sollen die SuS in den Aufgaben 2, 3 und 4 durch systematisches und zielorientiertes Probieren zu einer Lösung gelangen (z.B. Programmierung eines Fangen-Spiels) [MSW08]. Die Kompetenz „Die SuS probieren zunehmend systematisch und zielorientiert und nutzen die Einsicht in Zusammenhänge zur Problemlösung (lösen)“ [MSW08, S.59], wird also dahingehend geschult, dass die SuS neben dem systematischen Ausprobieren in den Aufgaben 3, 4 und 5 auch ihr Wissen über die Zusammenhänge von Sensoren, Aktoren, dem EVA-Prinzip und den Code-Blöcken in Scratch nutzen sollen und müssen, um zu einer angemessenen Lösung bzw. einem Scratch-Programm zu gelangen. Im Anschluss an die Durchführung ihrer Programme werden die SuS in Aufgabe 3 und 4 in einem letzten Schritt aufgefordert, noch einmal ihre Programm-Codes zu überprüfen und hinsichtlich ihrer Durchführbarkeit zu vergleichen bzw. zu reflektieren. Hierdurch wird schließlich die Kompetenz „Die SuS überprüfen ihre Programmier-Codes hinsichtlich ihrer Durchführbarkeit, suchen ggf. nach Fehlern, korrigieren diese und vergleichen ihre verschiedenen Lösungswege (reflektieren und überprüfen)“ [Formulierung angelehnt an MSW08] gefördert, da die SuS in Aufgabe 3 und 4 die eigenen Scratch-Programme zunächst eigenständig überprüfen und versuchen müssen durch das Ersetzen oder Ergänzen von Code-Blöcken Fehler zu beheben. Zudem werden während der Präsentation unterschiedlicher Lösungswege in der abschließenden Reflexionsphase gemeinsam mit der ganzen Klasse Gemeinsamkeiten und Unterschiede der Lösungen und Lösungswege hervorgehoben und verglichen. Die letzte Kompetenz („Die SuS übertragen ihre gewonnenen Erkenntnisse über Sensoren und Aktoren, sowie die Vorgehensweise bei der Programmierung dieser auf ähnliche Sachverhalte (übertragen)“ [Formulierung angelehnt an MSW08]) wird zunächst von den SuS ausgebaut, wenn sie im Klassengespräch und auch im zweiten Teil von Aufgabe 2 aufgefordert werden, den erarbeiteten Sachverhalt von Figuren in Scratch auf den eigenen Körper zu übertragen. Darüberhinausgehend können die SuS die in Aufgabe 3 verwendete Vorgehensweise zur Programmierung von Sensoren und Aktoren in Scratch im Rahmen der Knobelaufgabe 4 oder zumindest bei der Hausaufgabe auf ähnliche Sachverhalte übertragen und somit zu einem tiefergehenden anwendungsbezogenen Verständnis über Sensoren und Aktoren gelangen. Außerdem werden von den SuS Kompetenzen im **„Darstellen/ Kommunizieren“** [MSW08] geschult. Die SuS sollen sich während der gesamten Doppelstunde über ihre Lösungswege und Denkprozesse zu den Begrifflichkeiten

und der Programmierung von Sensoren und Aktoren mit ihren Mitschülerinnen und Mitschülern (teils unter Verwendung von Fachsprache) austauschen [ebd.]. Hinzukommend wird von den SuS in Aufgabe 3 und 4 gefordert, kooperativ eine Problemstellung zu bearbeiten und die Lösungswege und Gedankengänge Anderer nachzuvollziehen. Durch die schriftliche Fixierung von Lösungsmöglichkeiten und die Bearbeitung des Kompetenz-Check wird die Vorgehensweise nachvollziehbar dargestellt und Arbeitsergebnisse, sowie Lernerträge dokumentiert.

Sachunterrichtliche Kompetenzen:

Durch den Transfer der Thematik auf die Alltagswelt der Kinder, leistet die dargelegte Unterrichtsstunde einen Beitrag zu dem sachunterrichtlichen Schwerpunkt **„Körper, Sinne, Ernährung und Gesundheit“** aus dem Bereich **„Natur und Leben“**. Am Ende der Schuleingangsphase sollen die SuS die Bedeutung der eigenen Sinne in Alltagssituationen, sowie die Aufgaben einzelner Sinnesorgane ermitteln und beschreiben können [MSW08]. Diese Kompetenz wird in der geplanten Unterrichtsstunde dahingehend gefördert, dass die SuS in Reflexionsphasen und auch im zweiten Teil von Aufgabe 2 ihre Sinnesorgane als Sensoren ihres Körpers beschreiben lernen und ihre Aufgabe als „Fühler“ oder „Messgerät“ zur Signalübertragung an die Aktoren des Körper (z.B. Muskeln) ermitteln und erklären können.

Resultierend lässt sich festhalten, dass sich die Arbeit mit der Programmiersprache Scratch, sowie die Auseinandersetzung mit der Thematik der Sensoren, Aktoren und des EVA-Prinzip, für die fächerübergreifende Arbeit in Schule und Unterricht gut eignet, da zahlreiche Kompetenzen erworben bzw. ausgebaut werden können. Im Verlaufe des Scratch-Projekts können natürlich je nach Thematik die Schwerpunkte auf die Kompetenzen einzelner Fachbereiche gelegt werden. Neben den fachlichen Kompetenzen werden in dieser Doppelstunde zudem auch überfachliche Kompetenzen, wie beispielsweise die zu Beginn als operationalisierte Teilziele formulierten personalen und sozialen Kompetenzen, sowie methodische Kompetenzen (Selbstreguliertes Lernen) geschult.

Wird das Thema „Wie fühlt und reagiert die Scratch-Katze (Sensoren und Aktoren)?“ in der Unterrichtsreihe bzw. dem Projekt „Programmieren in der Grundschule unter Nutzung der Programmiersprache „Scratch““ unter der Perspektive der heutigen Gesellschaft betrachtet, zeichnet es sich durch eine hohe Gegenwarts- und Zukunftsbedeutung aus. In der Gesellschaft, in der die Kinder leben, finden sich im Alltag viele Formen von Sensoren und Aktoren. Ohne dass die SuS es bewusst wahrnehmen, nutzen bzw. beobachten sie täglich Sensoren und Aktoren, die ihnen den Alltag erleichtern. Allein das Smartphone, welches viele SuS in der vierten Klasse schon besitzen, besitzt ein Mikrofon (akustischer Sensor), einen Schallsensor für die Telefonie, Lage- und Beschleunigungssensor, einen Photodetektor (zur Erfassung der Lichtintensität der Umgebung) und einen Lichtsensor in der Kamera als Sensoren und Aktoren, wie beispielsweise den Display für die Wiedergabe von Buchstaben bzw. Symbolik und die Lautsprecher zur Wiedergabe von Tönen. Aber auch Bewegungssensoren an Lampen oder Türen, Temperatursensoren in Thermostaten, CO-Sensoren in Rauchmeldern zur Erfassung des Kohlenmonoxidgehalts in der Luft, Gewichtssensoren an Wagen oder Abstands- und Geschwindigkeitssensoren im Auto begegnen den Kindern in ihrem täglichen Leben. Durch die Thematisierung der Definition und der Funktionsweise von Sensoren und Aktoren im Unterricht, werden die SuS angeregt Sensoren und Aktoren in ihrer Umwelt bewusst wahrzunehmen und ihre Wirkweise besser nachzuvollzie-

**Relevanz für
die SuS**

hen. Die Kinder werden in der Zukunft mit immer mehr Sensoren und Aktoren in Kontakt kommen, da die Sensorik ein Gebiet der modernen Messtechnik ist, welche eine hohe wirtschaftliche Bedeutung hat [RW16]. So werden die SuS je nach Beruf und Interesse in ihrer Zukunft auch in Kontakt mit Sensoren „[...] in der Fertigungstechnik, in der Sicherungstechnik, in der Medizintechnik, in Kraftfahrzeugen und in vielen anderen Bereichen“ [RW16, Vorwort] kommen. Vor allem in der Informatik und der Nutzung digitaler Medien bzw. Informatiksystemen spielen Sensoren und Aktoren eine wichtige Rolle. Da viele der genannten Systeme zu komplex und umfangreich für die Grundschule sind oder Größen und Einsatzgebiete thematisieren, die den SuS noch nicht vertraut sind, eignet es sich im Kontext des Unterrichts auf die Sensoren und Aktoren am eigenen Körper (als nicht-technische Sensoren und Aktoren), sowie die in Scratch zurückzugreifen. Insbesondere die Sensoren und Aktoren am Körper werden von den Kindern tagtäglich genutzt und sind für sie somit leicht nachzuvollziehen und auch auszuprobieren. Auch das exemplarische Beispiel des Fangen-Spielens bzw. von Spielen generell hat einen engen Bezug zur Lebenswelt der Kinder und ist nahezu jedem Kind schon vor der Schulzeit vertraut. Ein Spiel aus ihrer Freizeit unter einer informatischen Perspektive näher zu untersuchen, dürfte für die SuS sehr spannend und motivationsfördernd sein. Auch wenn die thematisierten Sensoren und Aktoren noch sehr eingeschränkte Größen messen und simplere Beispiele darstellen, kann den SuS die Bedeutung von Sensoren und Aktoren im Alltag, sowie die allgemeine Funktionsweise dieser nahegebracht werden und eine Basis für weiterführendes Lernen geschaffen werden. So wird durch den dargelegten Unterrichtsentwurf ein spielerischer und frühzeitiger Einstieg in Lehrplaninhalte geboten, die beispielsweise im Kernlehrplan Informatik der gymnasialen Sekundarstufe II als Inhaltsfelder „Informatiksysteme“ und „Algorithmen“ [MSW14] erneut aufgegriffen und vertieft werden.

Darüber hinaus kommen SuS häufig bereits mit vielfältigen Vorerfahrungen bezüglich dem Umgang mit digitalen Medien in den Unterricht, sodass informatisches Wissen frühzeitig vermittelt werden sollte. Die Doppelstunde fördert im Rahmen einer bereits im Grundschulalter beginnenden digitalen und informatischen Bildung, dass die SuS „[...] eine informatische Mündigkeit und mündige Teilhabe in einer digitalisierten Gesellschaft [anbahnen], indem sie kompetent und selbstbestimmt digitale Medien nutzen, diese Nutzung kritisch und konstruktiv reflektieren und auch informatische modellierte Prinzipien dahinter erkennen“ [Gl18, S.1]. Dies ist umso wichtiger vor dem Kontext, dass der Alltag der Kinder immer mehr durch den Einfluss von Informatik auf gesellschaftliche und technologische Veränderungen geprägt ist. Die Doppelstunde bzw. Projektreihe hilft den SuS durch ihre frühzeitige Integration des Fachs Informatik in der Grundschule, sich in dieser Gesellschaft zurechtfinden und ein positives Selbstwirksamkeitskonzept hinsichtlich der Informatik aufzubauen [ebd.].

Die bereits zuvor erläuterte didaktische Reduktion des Sachinhaltes, sowie die Sicherstellung der unterrichtlichen Zugänglichkeit für die SuS werden durch eine angemessen strukturierte und kognitiv aktivierende Unterrichtsverlaufsgestaltung möglich. Unterteilt wird die Doppelstunde zwecks Sequenzierung und Artikulation in eine inhaltliche und spielerische Einführungs- und Hinführungsphase (inkl. Aktivierung des Vorwissens der SuS), eine längere Erarbeitungsphase (inkl. Zwischenreflexion), in welcher die SuS in zwei Erprobungsphasen die Funktionsweise von Sensoren und Aktoren erproben bzw. reflektieren und aktiv eigene Programmierungen erstellen und eine Phase der Ergebnispräsentation und -sicherung am Ende der Doppelstunde.

**Begründung
der wichtigsten
Entscheidungen
des
geplanten
Unterrichts**

Die Einstiegsphase wird durch eine Simulation eines Fangen-Spiels seitens der SuS im Stuhlkreis eröffnet, wodurch ein spielerischer und motivierender Zugang zu der Thematik erfolgen soll. Zudem ermöglicht der Einstieg an eine Situation aus der Lebenswelt bzw. die Interessen der SuS anzuknüpfen und auf diesem Wege die SuS kognitiv zu aktivieren und eine Partizipation aller SuS an der Diskussion zu gewährleisten [Kl12]. Durch verbale Impulse der Lehrkraft werden die SuS im Anschluss daran aufgefordert, ihre Beobachtungen bezüglich des Spiels zu äußern. Indem die Lehrkraft wichtige Äußerungen hervorhebt und richtige Anteile in Erklärungen der Kinder identifiziert, strukturiert sie die Diskussion inhaltlich gerade so sehr, dass die SuS eine erste Idee der Funktionsweise von Sensoren und Aktoren entwickeln können [ebd.]. Durch eine anschließende gemeinsame Betrachtung der Scratch-Katze an der Leinwand wird ein vorsichtiger Übergang bzw. eine Hinführung zu der Erarbeitungsphase geboten. Eine kurze Einführungsgeschichte soll die SuS motivieren und zum Diskutieren über bereits bekannte Funktionen der Code-Blöcke in Scratch anregen. Zudem werden die SuS durch einen verbalen Impuls der Lehrkraft („Glaubt ihr die Scratch-Katze kann auch fangen spielen?“) dazu angehalten, ihre Ideen und ihr Vorwissen bezüglich der Programmierung von Sensoren und Aktoren (ohne Verwendung dieser Begriffe) bzw. bekannten Funktionen in Scratch darzulegen, sodass die Lehrkraft das an dieser Stelle erhobene Vorwissen in die Interaktionen des weiteren Unterrichtsgeschehens einfließen lassen kann. Darüber hinaus wird durch die Impulse der Lehrkraft eine problemhaltige Situation geschaffen, die die Formulierung von Fragestellungen evoziert [ebd.]. Die anschließende gemeinsame Formulierung zweier Forschungsfragen für den weiteren Unterricht ermöglicht eine, in der Interaktion von SuS und Lehrkraft entstehende, gemeinsame Zielsetzung für die Doppelstunde. Um nicht nur die Zielsetzung, sondern auch das Vorgehen dieser Unterrichtsstunde den SuS transparent zu machen, werden neben den Forscherfragen auch weitere Stundenverlaufskarten (siehe Anhang) an die Tafel gehangen und zwecks inhaltlicher Klarheit von der Lehrkraft kurz vorgestellt [ebd.]. Bilder neben den einzelnen Phasen geben den SuS Aufschluss darüber, ob sie die Aufgabe allein, in Partnerarbeit oder im Theaterkreis gemeinsam mit der ganzen Klasse bearbeiten sollen. Auf diesem Wege wird der Unterricht strukturiert und den SuS wird der Sinn des Lernens bewusst gemacht, sowie zielgerichtetes Arbeiten ermöglicht. In der Phase der Hinführung wird den SuS zudem das zu bearbeitende Kapitel im Arbeitsheft kurz vorgestellt. Dies ermöglicht bereits vor Beginn der Erarbeitungsphase auf Fragen und Verständnisschwierigkeiten der SuS einzugehen, damit eine effektive und selbstständige Bearbeitung der ersten Aufgabe erfolgen kann. Für die Projektarbeit bietet sich ein Arbeitsheft besonders an, da in ihm alle Informationen und erlernten Begriffe bzw. Prinzipien geordnet gesammelt werden können und ein Nachschlagen jederzeit möglich ist. Es können Hintergrundinformationen oder grundlegendes Wissen über die Funktionsweise von Scratch (wie die Blockübersicht zu Beginn des Arbeitsheftes) schriftlich fixiert darboten werden. Überdies können die SuS jederzeit nachverfolgen, wie viel sie schon gelernt haben und wie sie ihren aktuellen Wissenstand innerhalb der Projektreihe einordnen können. Ein Kompetenz-Check am Ende jedes Kapitels (siehe Anlage) ermöglicht es, den SuS einen Überblick über ihre möglichen Schwachstellen zu geben und ihnen so die Chance zu geben, an diesen zu arbeiten. Das Arbeitsheft ist so aufgebaut, dass zunächst Symbole erläutert werden, welche die SuS durch das ganze Heft begleiten. Sie dienen einerseits der kindgerechten und strukturierten Gestaltung des Arbeitsheftes und andererseits der Unterstützung der SuS beim selbstregulierten Lernen. Bevor die SuS selbstständig arbeiten, wird von der Lehrkraft nochmal das Bearbeitungsschema „Planen, Durchführen, Reflektieren“,

wiederholt. Dieses Schema ist den SuS aus der vorherigen Arbeit in dem Arbeitsheft, sowie dem Sachunterricht vertraut und schult das selbstregulierte Lernen, welches eine Kernkompetenz eines mündigen, autonomen Lebens, sowie selbstständigen, motivierten und lebenslangen Lernens darstellt [GN17]. Das genannte Bearbeitungsschema findet sich in den Aufgaben 3 und 4 wieder, welche nach diesem gegliedert sind. Zwecks inhaltlicher Strukturierung organisiert die Lehrkraft die anschließende Erarbeitungsphase (Partnereinteilung, Erläuterung der Tippkarten und Knobelaufgabe) und gibt noch einmal den Hinweis auf das, in der Klasse eingeführte Helfersystem, welches auf die gegenseitige Unterstützung der SuS und kooperatives Lernen abzielt. Dazu hängen SuS, die bereits fertig mit der Aufgabe sind, ihren Namen an die Tafel. Daraufhin können SuS, die alleine nicht weiterkommen, sich einen Partner suchen, dessen Namensschild von der Tafel nehmen und sich auf diesem Wege Hilfe holen.

In der folgenden ersten Erarbeitungsphase erproben die SuS die Möglichkeiten der Programmierung bzw. die Funktionsweise von Sensoren und Aktoren in Scratch anhand der Nachprogrammierung von Spielen bzw. Animationen, welche im Arbeitsheft (Aufgabe 1) vorgegeben sind. Die Sozialform der Partnerarbeit wurde bewusst ausgewählt, da der soziale Austausch anregt sich wechselseitig durch differentes Denken anzuspornen noch mehr Funktionsweisen der eingesetzten Code-Blöcke zu entdecken. Im Anschluss erfolgt ein lehrkraft gelenktes Gespräch im Plenum, in welchem unter Rückbezug auf Visualisierungen der von den SuS erprobten Animationen zentrale Begriffe erarbeitet werden sollen. Die Erarbeitung der Begrifflichkeiten erfolgt nicht durch selbstreguliertes Lernen, da die Begriffe „Sensor“ und „Aktor“, sowie die daran geknüpfte Funktionsweise (inkl. EVA-Prinzip) sehr komplex sind und besser in einem durch die Lehrkraft strukturierten Gespräch erarbeitet werden können [KI12]. Durch eine gemeinsame Erarbeitung der Definitionen, sowie die Anwendung der Begrifflichkeiten, auf die von den SuS erprobten Animationen durch die SuS, werden die erlernten Fachwörter oftmals klarer und stärker hinterfragt. Darüber hinaus kann die Lehrkraft eine Diskussion zwischen den Kindern anregen, welche zu einer intensiven kognitiven Auseinandersetzung mit dem Thema anregt und ihnen verhilft gemäß den Teilzielen S1 und S2 die Bedeutung und Funktionsweise von Sensoren und Aktoren, sowie den Zusammenhang verschiedener Code-Blöcke und der Programmierung von Sensoren und Aktoren in Scratch nachzuvollziehen und zu verbalisieren [ebd.]. Ein Gespräch im Plenum verhilft der Lehrkraft zudem sicherzustellen, dass alle SuS die Begriffe „Sensor“ und „Aktor“, als Grundlage für die weiterführenden Aufgaben, verstanden haben. Neben den Teilzielen S1 und S2, wird an dieser Stelle auch die Entwicklung eines Programmierverständnisses der SuS gefördert, da sie aufgefordert werden das EVA-Prinzip anhand der erstellten Animationen zu erklären, sowie Sensoren und Aktoren zu identifizieren (S4). Die Zuordnung von Bildern der Code-Blöcke zu den Begriffen „Sensor“ und „Aktor“ an der Tafel verhilft den SuS gemäß Teilziel S2 eine Verbindung zwischen Sensoren und Aktoren und den Code-Blöcken in Scratch herzustellen (S2). Durch die Visualisierung an der Tafel, sowie die Übertragung der Definitionen und der Code-Blöcke in das Arbeitsheft werden die Begriffe und die Funktionsweise anschaulich, richtig definiert und klar sichtbar an der der Tafel bzw. im Arbeitsheft dargestellt und das Gelernte nochmals gefestigt. Auch die Verbindungen zwischen einzelnen Begriffen (z.B. zwischen dem Sensor und der Eingabe) können an der Tafel gut veranschaulicht werden.

Die anschließende kurze Pause dient der Auflockerung und Bewegung, sodass Konzentration und Aufmerksamkeit nach der Erarbeitung des fachlichen Hintergrundes wieder steigen können. Ferner können die gelernten Inhalte verarbeitet und verin-

nerlicht werden, bevor die zweite sehr praxisorientierte Erarbeitungsphase beginnt. Um das Erreichen des Teilziels S3, sowie einen roten Faden des Unterrichts sicherzustellen, werden zu Beginn der zweiten Erarbeitungsphase die erarbeiteten Begriffe und Funktionsweisen auf das Fangenspiel rückbezogen. Durch die offene Diskussion über Sensoren und Aktoren am menschlichen Körper, sowie in anderen Geräten des Alltags der Kinder, werden den Kindern die Relevanz und das breite Anwendungsgebiet der Thematik bewusst (S2). Durch die anschließende Bearbeitung des zweiten Teils von Aufgabe 2 im Arbeitsheft, kann jeder SuS für sich selbst nochmal prüfen, ob er das gewonnene Wissen auf den menschlichen Körper übertragen kann. Anschließend erfolgt die zweite Erprobungsphase, welche zwecks sozialer Differenzierung in Einzel- oder Partnerarbeit ausgeführt werden kann. Insbesondere schwächere SuS, die noch keine Idee für eine Programmierung eines Fangenspiels in Scratch haben, profitieren von dieser Wahl der Sozialform, da sie die Möglichkeit haben, sich mit einem Partner austauschen zu können, welcher möglicherweise bereits einige Ideen mitbringt. Erklärungen von Gleichaltrigen können für die Kinder sehr gewinnbringend sein, einen anderen Blickwinkel auf die Thematik bieten und dazu anregen eigene Denk- und Handlungsmuster zu reflektieren und ggf. zu modifizieren. Leistungsstärkere SuS haben dagegen die Chance, die Aufgaben ihrem eigenen Lerntempo entsprechend bearbeiten zu können. Die konzipierte Erprobungsphase ist im Allgemeinen so aufbereitet, dass die Lernenden sich nach dem Prinzip der natürlichen Differenzierung gemäß ihrer individuellen Lernvoraussetzungen und Lerntempi mit der eigenen Programmierung von Sensoren und Aktoren in Scratch auseinandersetzen können (Li14). Hierzu werden Aufgaben auf unterschiedlichen Schwierigkeitsniveaus und mit differenziertem Materialangebot dargeboten, um einer Demotivation und Über- bzw. Unterforderung vorzubeugen (ebd.). So lässt die Aufgabe 3 verschiedene Lösungen zu und die SuS können gemäß ihrem individuellen Lerntempo zu unterschiedlich komplexen Lösungen gelangen. Während es für leistungsschwächere SuS bereits ein Erfolg sein kann, eine der in der Aufgabe 3 genannten Bedingungen in Scratch-Blöcke umzusetzen, haben schnellere SuS die Möglichkeit sich in Knobelaufgabe 4 mit komplexeren Programmierungen von Sensoren und Aktoren auseinanderzusetzen. Zusätzlich zur Verfügung stehende Tippkarten auf zwei unterschiedlichen Unterstützungsniveaus unterstützen die Arbeit entsprechend den eigenen Kompetenzen und Lerntempi und stellen darüber hinaus sicher, dass jedes Kind das Lernziel erreichen kann. Vor Beginn der Einzel- bzw. Partnerarbeit betont die Lehrkraft, dass es verschiedene Möglichkeiten der Bearbeitung gibt und die einzige Bedingung die Programmierung von Sensoren und Aktoren ist. Auf diese Weise wird sichergestellt, dass die SuS individuelle Lernwege beschreiten und der abschließende Austausch in der Präsentationsphase gewinnbringend ist. Zur Ergebniskontrolle testen die SuS ihre Programmierungen am Computer hinsichtlich ihrer Durchführbarkeit und erhalten so eine schnelle Rückmeldung. Eine derartige „[...] schnelle Selbstkontrolle kann Erfolgserlebnisse verschaffen, die sich wiederum positiv auf die Lernmotivation auswirken“ [BD00, S.6]. Zudem ermöglicht diese Form der Ergebniskontrolle der Lehrkraft ihre Zeit denjenigen SuS zu widmen, die trotz der zur Verfügung stehenden Unterstützungsmaßnahmen individuelle Hilfestellungen benötigen. In der Auswertungs-/Präsentationsphase schafft die Lehrkraft einen Diskurs, indem sie die SuS dazu auffordert ihre Erfahrungen hinsichtlich der Programmierung von Sensoren und Aktoren mit Scratch zu verbalisieren. Ein Vorteil hierbei ist, dass eventuelle Missverständnisse nochmal mit der ganzen Klasse besprochen werden können und die verschiedenen Erklärungen der einzelnen Kinder das Verständnis der SuS unterstützen können. Durch diesen Austausch werden verschiedene Schü-

lerlösungen gegenübergestellt und somit die divergenten Ergebnisse und Erfahrungen der SuS in einen Zusammenhang gebracht, systematisiert und durch die Integration in bereits vorhandene Wissensstrukturen gesichert [Ri05]. Die SuS sollen während dieser Phase die zuvor erlernten Begriffe und Funktionsweisen anhand der präsentierten Programmierungen mittels eigenen Formulierungen und unter Verwendung der Fachbegriffe erklären. Durch diese Anwendung der Begriffe „Sensor“ und „Aktor“, sowie des EVA-Prinzips auf eigene Programmierungen wird der gelernte Inhalt noch einmal gefestigt und transferfähig gemacht. Je nach Verlauf des Diskurses kann die Lehrkraft an dieser Stelle einzelne Aspekte in Bezug auf Sensoren und Aktoren, sowie deren Umsetzungen in konkreten Programmierungen aufgreifen und erneut erläutern, sodass das Erreichen des Stundenziels gewährleistet werden kann. In dieser Phase wird das Klassengespräch als Sozialform gewählt, da es zum einen den Vorteil bietet, dass die Lehrkraft unklare Begriffe und Funktionsweisen noch einmal erläutern kann und zum anderen die Möglichkeit schafft, einen Rückbezug zu dem Fangenspiel zu Beginn der Stunde zu nehmen und die Forscherfrage abschließend zu beantworten.

Die letzte Phase des Unterrichts stellt die Sicherung dar. In dieser Phase wenden die SuS ihr gewonnenes Wissen über Sensoren und Aktoren auf ihren eigenen Programmier-Code aus Aufgabe 3 an. Durch diese Anwendungsaufgabe setzen sie sich abschließend nochmal in Einzelarbeit mit der Thematik auseinander und können ihren Wissenszuwachs bzw. mögliche Schwachstellen ermitteln. Durch die Beschreibung von Sensoren und Aktoren mit eigenen Worten wird das erlernte Wissen zudem gesichert. Somit dient diese Phase sowohl der Kontrolle des Verständnisses der Lernenden, als auch der Sicherung der Erreichung des Stundenziels durch alle SuS. Die Hausaufgabe (Aufgabe 5) dient noch einmal der Vertiefung und Festigung der gelernten Inhalte durch die Anwendung und den Transfer des Gelernten auf ähnliche Programmierungen in Scratch. Eine Begrenzung auf kurze Code-Blöcke wurde deshalb gewählt, da der Fokus auf der Identifikation von Sensoren und Aktoren in Scratch, sowie der eigenständigen Beschreibung ihrer Funktionsweise liegen soll. Dabei wurden die Rätsel so gewählt, dass sie den SuS Spaß machen und die Kinder nochmal motivieren, sich Zuhause mit der Thematik auseinanderzusetzen.

Zusammenfassend ist festzuhalten, dass die dargelegte Unterrichtsverlaufsgestaltung eine geeignete Methode darstellt die zu Beginn der Arbeit formulierten Ziele und Kompetenzen zu erreichen. Durch die selbstregulierte und handlungsorientierte Bearbeitung des Arbeitsheftes entdecken die SuS vielfältige Möglichkeiten der Programmierung von Sensoren und Aktoren in Scratch. Diese werden in lehrer gelenkten Diskursen gesammelt, systematisiert und mit Fachbegriffen in Verbindung gebracht. Ein ständiger Rückbezug zu der Alltagssituation „Fangen“ gewährleistet ein Verständnis der Relevanz der Thematik für den eigenen Alltag. Anschließend wird durch die Sicherungsphase das erworbene Wissen gefestigt und transferfähig gemacht, sodass die SuS am Ende der Doppelstunde die Funktionsweise von Sensoren und Aktoren erklären können, die Relevanz dieser für ihren eigenen Alltag aufzeigen können, sowie verschiedene Möglichkeiten der Programmierung von Sensoren und Aktoren in Scratch anwenden können. Somit wurde das Stundenziel erreicht.

3. Artikulationsschema³

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/ Aktionsform	Materialien/ Medien/Werkzeuge	didaktisch-methodischer Kommentar
5	Einstieg	<p>Sensoren und Aktoren im Alltag</p> <ul style="list-style-type: none"> • Die Lehrkraft bittet zwei SuS Fangen zu spielen • Die Lehrkraft fordert die zusehenden SuS auf, zu beschreiben, was sie gerade beobachtet haben • Die SuS berichten entsprechend • Die Lehrkraft fordert zu erneutem genauen Beobachten bei einem zweiten Durchgang des Fangenspielens auf; erneute Durchführung • Gespräch mit Lenkung durch die Lehrkraft: <ul style="list-style-type: none"> ○ Die SuS erkennen, dass eine gewisse Bedingung (hier: Fühlen von Berührung bzw. Sehen des „Gejagten“) zu einer bestimmten Reaktion/ Aktion führt (hier: Stoppen und selber fangen bzw. in die Richtung des Gejagten laufen) 	Lehrkraftgelenktes Gespräch im Plenum (Theaterkreis)	-	<ul style="list-style-type: none"> • Anknüpfung an das Vorverständnis bzw. die Vorerfahrungen aus der Lebenswelt der SuS • Eingangsmotivation schaffen durch: Lernmotivierung der SuS durch das Fangenspielen, Partizipation der SuS, Ansprache intrinsischer Motivation <p>(Ziele: S3, PS1)</p>
8	Hinführung	<p>Gemeinsame Betrachtung der Scratch-Katze</p> <ul style="list-style-type: none"> • Überleitung der Lehrkraft zu Scratch, indem die Einführungsgeschichte erzählt und leitende Fragen gestellt werden und Scratch mit dem Beamer an die Wand projiziert wird: 	Lehrkraftgelenktes Gespräch im Plenum (Theaterkreis)	Computer mit Internetzugang oder installierter Scratch-Software, Projektion: Scratch-Startseite mit der Scratch-Katze per	<ul style="list-style-type: none"> • Aktivierung des Vorwissens zu den bekannten Funktionen von Scratch: regt zum Austausch über die bekannten Funktionen an • Einführung in die zentralen

³ Unter der Artikulation wird im didaktischen Kontext die (zeitliche) Abfolge der Unterrichtsphasen verstanden.

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/ Aktionsform	Materialien/ Medien/Werkzeuge	didaktisch-methodischer Kommentar
		<ul style="list-style-type: none"> ○ Geschichte: Kater Karlo und Maus Minnie spielen auch fangen. Aber immer, wenn Kater Karlo Mini berührt entwischt sie ihm wieder ○ „Glaubt ihr, die Scratch Katze kann auch fangen spielen?“ ○ „Habt ihr eine Idee, wie ihr so ein Spiel programmieren könntet?“ ○ „Glaubt ihr, die Scratch Katze kann auch Dinge wahrnehmen und darauf reagieren?“ ● Leitfragen für die Unterrichtsstunde (gemeinsam mit den SuS erarbeiten und visualisieren): <ul style="list-style-type: none"> ○ Wie fühlt und reagiert die Scratch-Katze? ○ Wie können wir die Scratch-Katze so programmieren, dass sie fühlt und reagiert? ● Überblick über den Stundenverlauf (Karten an der Tafel) ● Vorstellung des Arbeitsheftes und kurze Wiederholung des Bearbeitungsschemas: Planung - Durchführung – Reflexion (Bekannt aus der vorherigen Arbeit mit Scratch und dem Sachunterricht) ● Organisation der Arbeitsphase: <ul style="list-style-type: none"> ○ Einteilung der Klasse in Paare ○ Vorstellung der Tippkarten ○ Erläuterung der Aufgaben für SuS, die frühzeitig fertig sind (Helfersystem; 		<p>Beamer; Arbeitsheft; Tippkarten; Studententransparenzkarten</p>	<p>Aspekte des Themas</p> <ul style="list-style-type: none"> ● Vermittlung eines Orientierungsrahmens zum Stundenverlauf: Zielorientierung, Transparenz von Stundenziel und Unterrichtsphasen, Verortung der Stunde innerhalb der Unterrichtsreihe <p>(Ziele: Anbahnung von S1)</p>

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/ Aktionsform	Materialien/ Medien/Werkzeuge	didaktisch-methodischer Kommentar
		Knobelaufgabe)			
10	Erarbeitung	Erprobung vorgegebener Spiele bzw. Animationen <ul style="list-style-type: none"> Bearbeitung der Aufgabe 1 im Arbeitsheft Die SuS programmieren die einfachen Animationen bzw. Spiele (die Ihnen teils aus ihrer Lebenswelt vertraut sind) und testen sie aus (pro Kind/Partnerteam zwei Animationen bzw. Spiele) Genauere Beobachtung der Programmier-Codes durch die SuS 	Partnerarbeit	Arbeitsheft; mindestens für jedes zweite Kind, einen Computer mit Internetzugang oder dem installierten Programm Scratch	<ul style="list-style-type: none"> Aktive und selbstständige Erarbeitung in den Sachzusammenhang Strukturierte Erarbeitung <p style="text-align: center;">(Ziele: S1, S2, M1)</p>
22		Erarbeitung des fachlichen Hintergrundes <ul style="list-style-type: none"> Lehrkraftgelenktes Gespräch unter Rückbezug auf Visualisierungen, der von den SuS erprobten Spielen bzw. Animationen durch einen Beamer Leitende Fragen der Lehrkraft: <ul style="list-style-type: none"> Was ist euch an den Programmiercodes aufgefallen? (Erarbeitung der Fühlen- und Falls-Dann-Blöcke) Können Figuren oder Gegenstände in Scratch fühlen und reagieren? Wie können sie das? Was muss ich beim Programmieren beachten, wenn ein Gegenstand bzw. eine Figur fühlen und reagieren soll? Erarbeitung zentraler Begriffe (Die Lehrkraft bahnt durch gezielte Interaktion mit den SuS und deren Aussagen zu den 	Lehrkraftgelenktes Gespräch im Plenum (Theaterkreis)	Computer mit Internetzugang oder installierter Scratch-Software, Projektion: Spiele bzw. Animationen; Definitionskarten; Codekarten; Tafel	<ul style="list-style-type: none"> Aktive und schrittweise Einarbeitung in den Sachzusammenhang Erarbeitung der fachlichen Thematik unter Rückbezug auf die Beobachtungen der SuS Gemeinsames Erarbeiten der Begrifflichkeiten dient der Kontrolle des Verständnisses dieser Fachtermini <p style="text-align: center;">(Ziele: S1, S2, PS1)</p>

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/ Aktionsform	Materialien/ Medien/Werkzeuge	didaktisch-methodischer Kommentar
		<p>Spielen bzw. Animationen die Unterscheidung in Sensoren und Aktoren an)</p> <ul style="list-style-type: none"> ○ Visualisierung der erarbeiteten Definitionen an der Tafel (Wortspeicher) ○ Erarbeitung der Definition von „Falls-dann-Befehlen“, als Grundlage des Programmierens von Sensoren und Aktoren in Scratch ○ Zuordnung von Scratch-Codes zu den Begriffen „Sensor“, „Aktor“ und „Falls-dann-Befehl“ ○ Zuordnung und Erläuterung der Begriffe Eingabe, Verarbeitung und Ausgabe, als Anbahnung des EVA-Prinzips (Erläuterung nur in Bezug auf Sensoren und Aktoren) <ul style="list-style-type: none"> ● Übertragung der Definitionen und der Code-Blöcke in das Arbeitsheft 			
10	Pause	(z.B. Bewegungsspiele zur Auflockerung)			
30	Fortsetzung der Erarbeitung/ Anwendung	<p>Lebensweltbezug schaffen</p> <ul style="list-style-type: none"> ● Anwendung der erlernten Begriffe auf das Fangenspiel (Sensoren und Aktoren am menschlichen Körper) ● Diskussion weiterer möglicher Sensoren und Aktoren im Alltag der Kinder (z.B. Temperaturregler) ● Klärung der Relevanz von Sensoren und Aktoren im lebensweltlichen Kontext ● Ausfüllen des zweiten Teils von Aufgabe 2 	Lehrkraftgelenkte Diskussion im Plenum (Theaterkreis)	Computer mit Internetzugang oder installierter Scratch-Software; Arbeitsheft; Tippkarten	<ul style="list-style-type: none"> ● Festigung der erlernten Thematik durch Verknüpfung mit Aspekten der Lebenswelt der SuS ● SuS führen die Arbeitsanweisungen eigenständig anhand des Arbeitsheftes durch <ul style="list-style-type: none"> ○ Anwenden des bereits erlernten Wissens und Ausbau der neu erworbenen

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/ Aktionsform	Materialien/ Medien/Werkzeuge	didaktisch-methodischer Kommentar
		<p>im Arbeitsheft</p> <p>Eigene Programmierung von Sensoren und Aktoren</p> <ul style="list-style-type: none"> Lehrkraft leitet unter Rückbezug auf die Einstiegsgeschichte die Arbeitsphase ein Kinder arbeiten selbstständig im Arbeitsheft an Aufgabe 3 Lehrkraft betont, dass es verschiedene Möglichkeiten gibt, die Aufgabe zu lösen und unterschiedliche Codeblöcke möglich sind Lehrkraft verweist auf Knobelaufgabe für die schnelleren SuS (Aufgabe 4) 	<p>Einzelarbeit/ Partnerarbeit</p>		<p>Kompetenzen</p> <ul style="list-style-type: none"> Verschiedene Lösungen auf unterschiedlichem Niveau, sowie Zusatzaufgaben ermöglichen differenziertes Arbeiten Schulung kooperativen Lernens durch „Helferprinzip“ <ul style="list-style-type: none"> Lehrkraft gibt bei Bedarf individuelle Hilfestellungen und unterstützt die SuS in ihrem individuellen Lernprozess <p>(Ziele: S1, S2, S3, S4, PS2, M1)</p>
10	<p>Auswertung/ Präsentation</p>	<p>Vorstellung der Arbeitsergebnisse</p> <ul style="list-style-type: none"> Die SuS werden dazu aufgefordert, ihre Erfahrungen hinsichtlich der Programmierung von Sensoren und Aktoren mit Scratch zu verbalisieren Einzelne eigene Programme zu Aufgabe 3 werden am Beamer von den SuS präsentiert: Sensoren und Aktoren werden anhand dieser Programme aufgezeigt und mit eigenen Wörtern erläutert Relevante Aspekte in Bezug auf Sensoren und Aktoren, sowie deren Umsetzung in konkreten Programmierungen werden dabei durch die Lehrkraft erneut aufgegriffen und vertieft 	<p>Lehrkraftgelenkte Diskussion im Plenum (Theaterkreis)</p>	<p>Computer mit Internetzugang oder installierter Scratch-Software; Beamer</p>	<ul style="list-style-type: none"> Präsentation und Zusammenfassung der Ergebnisse unter Rückbezug auf die fachliche Thematik Methodenreflexion Beseitigung von letzten Fragen <p>(Ziele: S1, S2, S4, PS1)</p>

Dauer (min)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/Aktionsform	Materialien/Medien/Werkzeuge	didaktisch-methodischer Kommentar
		<ul style="list-style-type: none"> • Es wird nochmal ein Rückbezug zum Fangen am Anfang und zu der Forschungsfrage der Sitzung hergestellt 			
5	Sicherung Und Hausaufgabe	<p>Sicherung der Diskussionsergebnisse</p> <ul style="list-style-type: none"> • SuS werden von der Lehrkraft aufgefordert, in ihrem Arbeitsheft in ihrem Programmiercode die Sensoren und Aktoren zu markieren und daneben zu schreiben, warum es sich um Codes zur Programmierung der Sensoren und Aktoren handelt <p>Aufgaben der Hausaufgabe</p> <ul style="list-style-type: none"> • Die Kinder sollen das Erlernte Zuhause erneut anwenden, indem sie sich ein bis zwei Rätsel aussuchen und zu diesen einen Programmiercode aufschreiben, in welchem sie die Code-Blöcke der Sensoren und Aktoren markieren und erläutern, warum es sich um solche handelt 	<p>Einzelarbeit</p> <p>Lehrerinstruktion im Plenum (jeder SuS auf seinem Platz)</p>	Arbeitsheft	<ul style="list-style-type: none"> • Festigung der Erlernten • Integration des Erlernten • Verarbeitung, Transfer, Ausklang <p style="text-align: center;">(Ziele: S1, S2, M1)</p>

Literaturverzeichnis

- [Ba17] Bartmann, E.: Mit Scratch die elektronische Welt entdecken. Bombini, Bonn, 2017.
- [BBC17] Brennan, K.; Balch, C.; Chung, M. (Harvard Graduate School of Education): Kreative Informatik, <https://eis.ph-noe.ac.at/wp-content/uploads/2017/05/kreative-informatik-lhb.pdf>, Stand: 09.04.2019.
- [BD00] Beuren, A.; Dahm, M.: Lernen an Stationen. Unterricht Biologie 249, S.4-9, 2000.
- [En15] Enactus Aachen e. V. (Hrsg.): Informatik und Programmieren für Kinder. Modulhandbuch für die Verwendung in Informatikkursen der Klassen drei bis sechs, Aachen, 2015, <https://docplayer.org/8818565-It4-kids-modulhandbuch-c-2015-enactus-aachen-e-v-1-vorabversion.html>, Stand: 04.04.2019.
- [FH11] Fischer, P.; Hofer, P.: Lexikon der Informatik. 15. überarbeitete Auflage, Springer, Berlin/Heidelberg, 2011.
- [GI18] Gesellschaft für Informatik (GI) e. V. (Hrsg.): Kompetenzen für informatische Bildung im Primarbereich. Entwurfsfassung für Empfehlungen der Gesellschaft für Informatik e.V., <https://uni-w.de/14k>, Stand: Juni 2018.
- [GN17] Götz, N.; Nett, U.: Selbstreguliertes Lernen. In: Götz, T. (Hrsg.): Emotion, Motivation und selbstreguliertes Lernen. 2. Auflage, Ferdinand Schöningh, Paderborn, S. 144-185, 2017.
- [Ha19] Hartkopf, K. et.al. (RWTH Aachen University): Spielend Programmieren lernen mit Scratch, <https://schuelerlabor.informatik.rwth-aachen.de/module/scratch>, Stand: 09.04.2019.
- [Ka07] Kattmann, U.: Didaktische Rekonstruktion – eine praktische Theorie. In: Krüger, D. & Vogt, H. (Hrsg.): Theorien in der biologiedidaktischen Forschung. Springer, Berlin/Heidelberg, S. 93-104, 2007.
- [KM16] Kultusministerkonferenz (Hrsg.): Bildung in der digitalen Welt. Strategie der Kultusministerkonferenz, kmk, Berlin, 2016, https://www.kmk.org/fileadmin/Dateien/pdf/PresseUndAktuelles/2017/Strategie_neu_2017_datum_1.pdf, Stand: 10.04.2019.
- [Li14] Lilienthal, J.: Binnendifferenzierte Lernumgebung: Lernstationen im Biologieunterricht. Bachelor + Master Publishing, Hamburg, 2014.
- [LK19] Lifelong-Kindergarten-Group am Media-Lab des MIT: Scratch, <https://scratch.mit.edu/>, Stand: 03.04.2019.
- [Ma08] Maloney, J. et.al.: Programming by Choice: Urban Youth Learning Programming with Scratch, 2008, <https://web.media.mit.edu/~mres/papers/sigcse-08.pdf>, Stand: 06.04.2019.

- [MSW08] Ministerium für Schule und Weiterbildung des Landes NRW (Hrsg.): Richtlinien und Lehrpläne für die Grundschule in NRW, Ritterbach Verlag, Frechen, 2008, http://www.schulentwicklung.nrw.de/lehrplaene/upload/lehrplaene_download/grundschule/grs_faecher.pdf, Stand: 07.07.2016.
- [Ri05] Rieck, K.: Gute Aufgaben. Handreichungen des Programms SINUS an Grundschulen. IPN, Kiel, 2005.
- [RW16] Rost, M.; Wefel, S.: Sensorik für Informatiker. Erfassung und rechnergestützte Verarbeitung nichtelektrischer Messgrößen. De Gruyter, Berlin, 2016.
- [Sc15] Schaumburg, H.: Chancen und Risiken digitaler Medien in der Schule. Medienpädagogische und – didaktische Perspektiven. Bertelsmann Stiftung, Gütersloh, 2015.
- [Sw17] Schweigart, A.: Coole Spiele mit Scratch. Lerne programmieren und baue deine eigenen Spiele. dpunkt Verlag, Heidelberg, 2017.
- [Wa06] Wallaschek, J.: Sensoren und Aktoren. In: Steinhilper, W.; Sauer, B. (Hrsg.): Konstruktionselemente des Maschinenbaus 2. 5. Auflage, Springer, Berlin/ Heidelberg, S. 665-707, 2006.
- [Wb17] Braun, W.: Grundlagen der Informatik. Bildungsverlag EINS, Köln, 2017.

Abbildungsverzeichnis

- Fragezeichen: <https://pixabay.com/de/vectors/hilfe-frage-informationen-info-97859/>, 04.04.2019.
- Glühbirne: <https://pixabay.com/de/vectors/birne-strom-halo-idee-licht-1293332/>, Stand: 04.04.2019.
- Grüner Smiley: <https://pixabay.com/de/vectors/smiley-gr%C3%BCn-einfache-gl%C3%BCcklich-146093/>, Stand: 04.04.2019.
- Menschlicher Körper: <https://pixabay.com/de/vectors/menschlichen-k%C3%B6rper-kreislauf-system-311864/>, Stand: 28.03.2019.
- Roter Smiley: <https://pixabay.com/de/vectors/smiley-gr%C3%BCn-einfache-unzufrieden-146094/>, Stand: 04.04.2019.
- Schriftrolle: <https://pixabay.com/de/vectors/schriftrollen-rollen-papyrus-34607/>, Stand: 29.03.2019.
- Stern: <https://pixabay.com/de/vectors/stern-gelb-golden-weihnachten-35289/>, Stand: 04.04.2019.
- Symbole Sozialformen:
https://www2.zaubereinmaleins.de/2010/klassenraum/dina5_symbole1.pdf, Stand: 29.03.2019.
- Alle weiteren Abbildungen wurden dem Programm Scratch entnommen:
<https://scratch.mit.edu/projects/301987588/editor>, Stand: 29.03.2019.

Anhang

I. Stundenverlaufskarten

Forschungsfragen:

Wie fühlt und reagiert die Scratch-Katze?

Wie können wir die Scratch-Katze so programmieren, dass sie fühlt und reagiert?

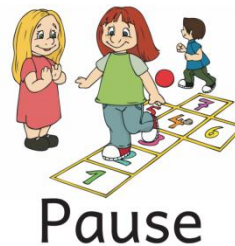
**Arbeit im Arbeitsheft
(Aufgabe 1)**



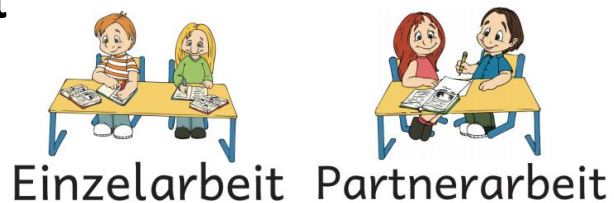
Klärung wichtiger Begriffe



Pause



**Arbeit im Arbeitsheft
(Aufgabe 3 und
Knobelaufgabe 4)**

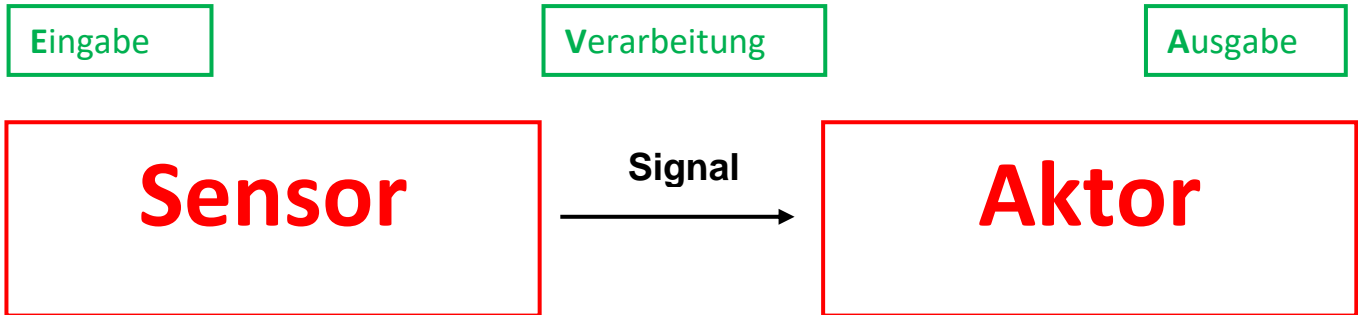


**Präsentation der
Ergebnisse**



Hausaufgabe: (hier ist Platz für einen Tafelanschrieb der Lehrkraft)

II. Tafelbild für die Erarbeitung der fachlichen Begriffe mitsamt benötigter Karten

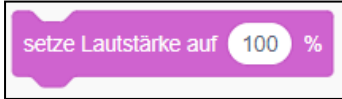
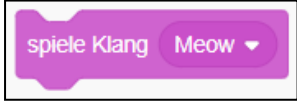
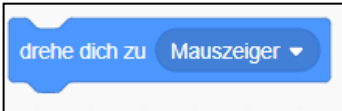
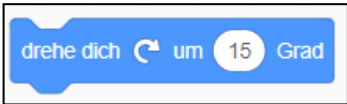
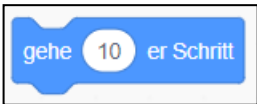
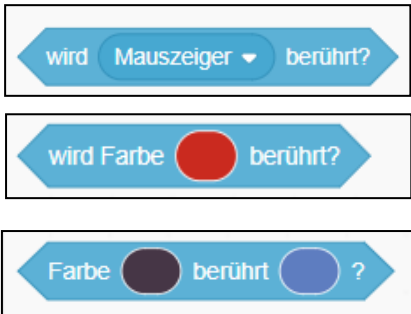


Definition: (Idealbeispiel → Gemeinsame Erarbeitung mit SuS)
 Sensoren sind Fühler an Gegenständen oder Figuren in Scratch, die auf Berührungen oder auf Änderungen in ihrer Umgebung warten und darauf reagieren. Die Scratch-Katze wartet hier z.B. auf eine Berührung einer Farbe. Durch Sensoren wird ein Befehl oder ein Signal ausgelöst, welches an die Aktoren weitergegeben wird.

Definition: (Idealbeispiel → Gemeinsame Erarbeitung mit SuS)
 Aktoren sind Macher. Sie setzen Befehle und Signale der Sensoren um. Die Scratch-Katze bewegt sich, ändert ihr Aussehen oder sagt etwas.

So programmiere ich Sensoren in Scratch:

So programmiere ich Aktoren in Scratch:



Falls-Dann-Befehle:

Durch einen »Falls-Dann-Befehl« kann man programmieren, wie die Aktoren einen bestimmten Befehl ausführen (z.B. Abspielen eines Tons), wenn etwas Bestimmtes an den Sensoren (z.B. Berühren einer Farbe) passiert.



III. Arbeitsheft

Mein

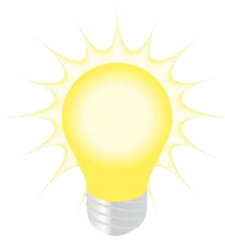


Arbeitsheft

Dieses Heft gehört: _____



Hi! Ich bin die Scratch-Katze. Ich begleite dich durch dein ganzes Arbeitsheft. Achte auf mich, denn ich gebe dir viele Ratschläge und Hinweise!



Die Glühlampe zeigt dir an, dass Tippkarten zu dieser Aufgabe am Lehrerpult liegen. Wenn du Schwierigkeiten bei einer Aufgabe hast, können sie dir weiterhelfen!

Nimm zuerst die **grüne Tippkarte**. Sie gibt dir Ideen für die Bearbeitung der Aufgabe. Falls du immer noch Schwierigkeiten hast, nehme die **rote Tippkarte**.



Knobelaufgabe! Du kannst diese Aufgabe bearbeiten, wenn du die restlichen Aufgaben schon fertig hast. Die Bearbeitung dieser Aufgabe ist freiwillig.



Kompetenzcheck! Hier kannst du ankreuzen, was du schon gut kannst und woran du noch arbeiten möchtest. Niemand außer dir darf diesen Bereich ansehen.

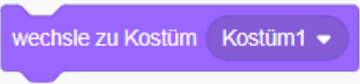
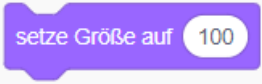
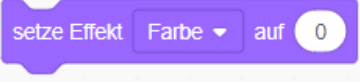
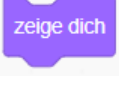
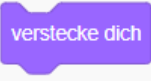
Blöcke in Scratch:

Bewegung

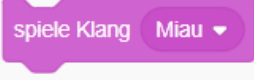
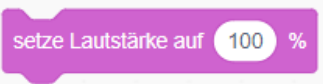
	<p>...benutzt Du, wenn Du willst, dass die Figur in ihrer Bewegungsrichtung läuft.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur sich ein wenig nach rechts dreht. Diagonal heißt 45 Grad, rechteckig heißt 90 Grad, einmal umdrehen heißt 180 Grad, keine Drehung heißt 360 Grad.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur sich ein wenig nach links dreht. Diagonal heißt 45 Grad, rechteckig heißt 90 Grad, einmal umdrehen heißt 180 Grad, keine Drehung heißt 360 Grad.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur sich an einen anderen Ort auf der Bühne sofort hinstellt. Den Ort gibst Du ein als x-Koordinate und y-Koordinate. Je weiter nach rechts, desto größer x. Je weiter nach oben, desto größer y.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur an einen anderen Ort auf der Bühne langsam hingleitet. Je langsamer die Figur gleiten soll, desto mehr Sekunden solltest Du eingeben. Den Ort gibst Du ein als x-Koordinate und y-Koordinate. Je weiter nach rechts, desto größer x. Je weiter nach oben, desto größer y.</p>
	<p>...benutzt Du, wenn Du willst, dass deine Figur immer in die Richtung zeigt, wo dein Mauszeiger ist.</p>
	<p>...benutzt Du, wenn Du willst, dass deine Figur nicht von dem Bildschirm verschwindet. Deine Figur prallt vom Rand des Bildschirms ab und kann den Bildschirm nicht verlassen.</p>

Aussehen




	<p>...benutzt Du, wenn Du willst, dass die Figur Hallo! sagt. Du kannst die Figur auch anderes sagen lassen. Durch die Zahl der Sekunden entscheidest Du, wie lange die Figur etwas sagen soll.</p>
--	---

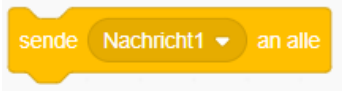
	<p>...benutzt Du, wenn Du willst, dass die Figur ein anderes Kostüm anzieht. Alle Kostüme, die eine Figur anziehen kann, findest Du unter dem Bereich <i>Kostüme</i> rechts neben <i>Skripte</i> und links neben <i>Klänge</i>.</p>
	<p>...benutzt Du, wenn Du willst, dass deine Figur ihre Größe ändert. Nimmst Du eine Zahl die größer als 100 ist, wird die Figur größer. Nimmst Du eine Zahl die kleiner ist als 100, wird die Figur kleiner.</p>
	<p>...benutzt Du, wenn Du willst, dass deine Figur ihre Farbe ändert.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur sich versteckt, also unsichtbar wird.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur sich zeigt, also sichtbar wird.</p>

Klänge

	<p>...benutzt Du, wenn Du willst, dass die Figur den Klang "Miau" macht. Alle Klänge, die eine Figur machen kann, findest Du unter dem Bereich <i>Klänge</i> rechts neben <i>Skripte</i> und <i>Kostüme</i>.</p>
	<p>...benutzt Du, wenn du Willst, dass sich die Lautstärke verändert. Setzt Du eine Zahl über 100 ein, wird die Lautstärke lauter. Setzt Du eine Zahl unter 100 ein, wird die Lautstärke leiser.</p>

Ereignisse

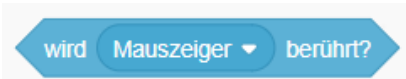
	<p>...schreibst Du am Anfang eines Programm-Codes, wenn Du Willst, dass dein Programm-Code durch den Druck auf die grüne Flagge ausgeführt wird.</p>
	<p>...benutzt Du, wenn Du willst, dass dein Programm-Code durch den Druck der Leertaste ausgeführt wird. Es kann auch ein beliebig anderer Tastaturknopf gewählt werden.</p>
	<p>...benutzt Du, wenn Du willst, dass diese Figur auf die Nachricht einer anderen Figur reagiert. Gib die Nachricht ein, auf die diese Figur reagieren soll.</p>




	<p>...benutzt Du, wenn Du willst, dass diese Figur den anderen Figuren eine Nachricht sendet. Gebe die Nachricht ein, die diese Figur versenden soll.</p>
---	---

Steuerung

	<p>...benutzt Du, wenn Du willst, dass die Figur 1 Sekunde lang nichts tut. Du kannst die Zahl der Sekunden, die diese Figur warten soll, auch verändern.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur etwas immer wieder macht. Die in der Klammer stattfindenden Blöcke werden die ganze Zeit über wiederholt.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur, etwas nur dann macht, wenn eine bestimmte Bedingung erfüllt ist (wenn ein bestimmtes Ereignis passiert). In die Lücke kannst du Blöcke aus der Kategorie <i>Fühlen</i> einsetzen.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur, etwas nur dann macht, wenn eine bestimmte Bedingung erfüllt ist (wenn ein bestimmtes Ereignis passiert). Falls die Bedingung nicht erfüllt ist, soll auch etwas Bestimmtes passieren.</p>
	<p>...benutzt Du, wenn Du willst, dass die Figur etwas so lange wiederholt, bis eine bestimmte Bedingung erfüllt ist (bis ein bestimmtes Ereignis passiert).</p>
	<p>...benutzt Du, wenn Du willst, dass alle Figuren zum Stillstand kommen. Nützlich für das Ende eines Spiels.</p>

Fühlen

	<p>...benutzt Du in einem Falls-Dann-Block (siehe oben), wenn Du willst, dass die Figur etwas tut, sobald sie mit dem Mauszeiger berührt wird. Du kannst auch eine andere Figur in dem Feld auswählen. Dann tut die deine Figur etwas, sobald sie eine andere Figur (z.B. ein Flugzeug) berührt.</p>
---	--

	<p>...benutzt Du in einem Falls-Dann-Block (siehe oben), wenn Du willst, dass die Figur etwas tut, sobald sie eine bestimmte Farbe (z.B. Blau) berührt.</p>
	<p>...benutzt Du in einem Falls-Dann-Block (siehe oben), wenn Du willst, dass die Figur etwas tut, sobald die erste angezeigte Farbe (z.B. rot) die zweite angezeigte Farbe (z.B. blau) berührt.</p>
	<p>...benutzt Du in einem Falls-Dann-Block (siehe oben), wenn Du willst, dass die Figur etwas tut, sobald Du eine bestimmte Taste (z.B. die Leertaste) drückst.</p>

Erläuterungen der Tabelle modifiziert, ergänzt und teils übernommen auf Basis der Quelle:

Enactus Aachen e. V. (Hrsg.): *Informatik und Programmieren für Kinder. Modulhandbuch für die Verwendung in Informatikkursen der Klassen drei bis sechs*, Aachen, 2015, <https://docplayer.org/8818565-lt4-kids-modulhandbuch-c-2015-enactus-aachen-e-v-1-vorabversion.html>, Stand: 04.04.2019.

Kapitel 5:



Wie fühlt und reagiert die Scratch-Katze?

1. Beispiele: Spiele und Animationen

Suche dir ein bis zwei Animationen oder Spiele aus. Baue die Programme nach und beobachte genau, wie die Scratch- Katze und andere Gegenstände fühlen und reagieren!

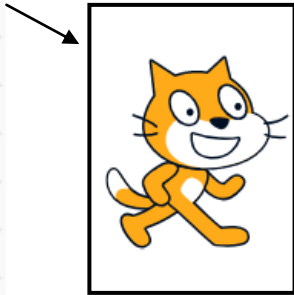


Du weißt nicht, was die hellblauen Blöcke bedeuten? Sehe dir die Kategorie **Fühlen** unter „Blöcke in Scratch“ an!

Beispiel 1: Lecker Apfel

Benötigte Figuren:

```
Wenn [ ] angeklickt wird
wiederhole fortlaufend
  falls [wird Farbe [ ] berührt?] , dann
    spiele Klang [Miau]
    sage [Lecker Apfel!] für [0.5] Sekunden
    stoppe [alles]
  sonst
    gehe [2] er Schritt
```



Beispiel 2: Klopfendes Herz

Benötigte Figuren:

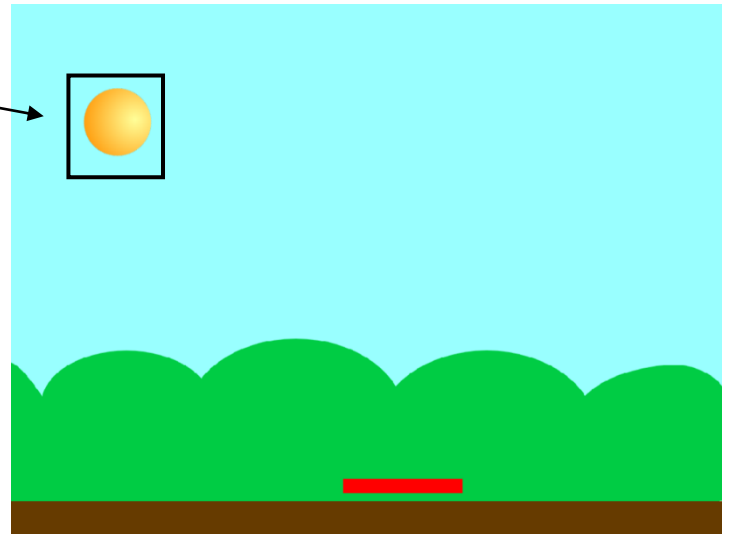
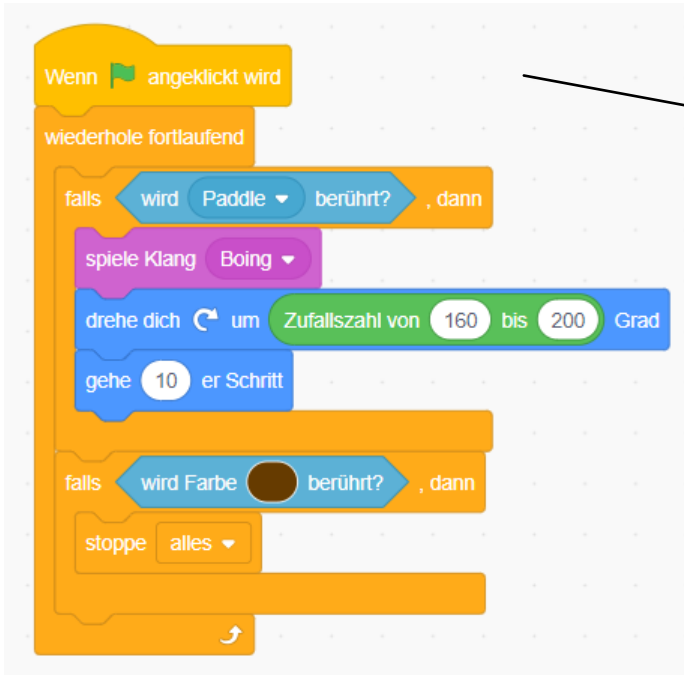
```
Wenn [ ] angeklickt wird
wiederhole fortlaufend
  falls [wird [Mauszeiger] berührt?] , dann
    wiederhole [10] mal
      spiele Klang [Pop]
      warte [1] Sekunden
      setze Größe auf [80]
      warte [0.2] Sekunden
      setze Größe auf [100]
    
```



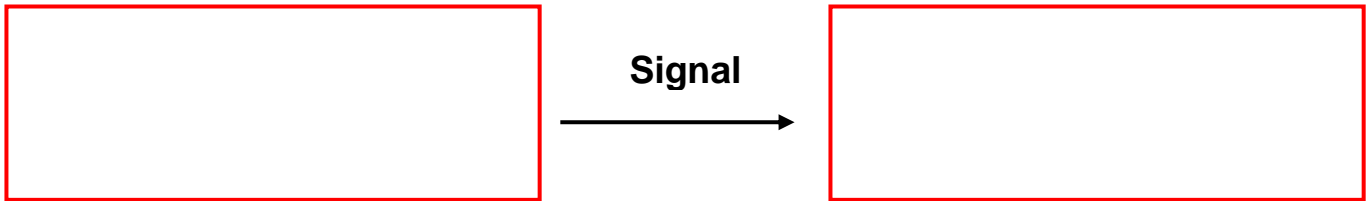
Beispiel 3: Pong

Bühnenbild und Figuren:

Lade zunächst die Datei „Pong“ aus dem Scratch-Ordner hoch. Danach musst du nur noch den Scratch-Bereich des Balls programmieren. Das Paddle kannst du mit der linken und rechten Pfeiltaste bewegen.



2. Mein Wortspeicher



Definition:

Definition:

Mit Blöcken aus diesem Bereich programmiere ich Sensoren in Scratch:

Mit Blöcken aus diesen Bereichen programmiere ich Aktoren in Scratch:

1. _____

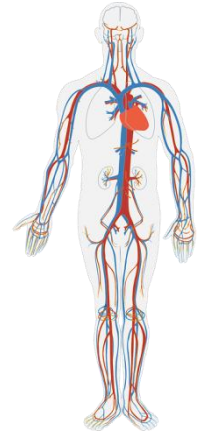
1. _____

2. _____

3. _____

Falls-Dann-Befehle:

Schreibe einen Sensor und einen Aktor auf, den du an deinem Körper finden kannst. Beschreibe wie der Sensor und Aktor funktionieren. Worauf wartet der Sensor? Wie reagiert der Aktor? Finde ein Beispiel, an dem du die Funktionsweise erläutern kannst.



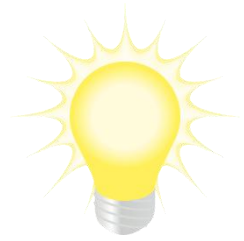
Wenn du kannst nutze die Begriffe „Eingabe“, „Verarbeitung“ und „Ausgabe“ in deiner Erklärung!

Zur Erinnerung:

An den Sensoren erfolgt die **Eingabe**. Dies ist eine Größe oder eine Veränderung in der Umgebung, die der Sensor messen kann. Bei der **Verarbeitung** wird die Eingabe weiter verarbeitet und an die Aktoren weitergeleitet. An den Aktoren erfolgt die **Ausgabe**. Die Ausgabe kann zum Beispiel eine Bewegung oder ein Ton sein. Diese Verarbeitung von Informationen nennt man übrigens auch **EVA-Prinzip**.



3. Fang die Maus!




Kater Karlo will Maus Minnie fangen! Immer, wenn der Kater Minnie rufen hört „Fang mich doch!“, rennt er in ihre Richtung. Doch immer, wenn Kater Karlo die Maus gerade berührt und ruft „Hab ich dich!“, entwischt sie ihm wieder und rennt weiter.


Nun bist du an der Reihe: Überlege, wie du Karlo und Minnie auf Scratch so programmieren kannst, dass sie Fangen spielen.

Achtung: Nur Karlo versucht Minnie zu fangen!

1. Planung: Notiere hier die Code-Blöcke von Karlo und Minnie:

Kater Karlo

Wenn  angeklickt wird



Denke an die Blöcke für die **Sensoren** und **Aktoren**! Sie können dir bei dieser Aufgabe helfen!

Maus Minnie



Wenn  angeklickt wird

2. Durchführung: Teste dein Programm.

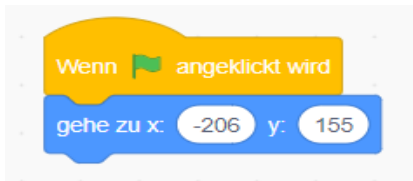
3. Reflexion: Etwas funktioniert nicht? Suche den Fehler und überarbeite deinen Programmier-Code.

4. Das Labyrinth



Kater Karlo hat Maus Minnie immer noch nicht gefangen. Nun ist Minnie auch noch in ein Labyrinth gelaufen. Immer wenn Kater Karlo die Wände berührt, ruft er laut „Miau“ und muss zum Anfang des Labyrinths zurückkehren. Er hört erst mit der Suche auf, wenn er Maus Minnie gefangen hat. Du kannst ihm dabei helfen, indem du deinen Code so programmierst, dass Kater Karlo deinem Mauszeiger folgt.

1. Öffne die Datei „Labyrinth“ in dem Scratch Ordner.
2. Programmiere Kater Karlo so, dass alle oben genannten Bedingungen erfüllt sind.
Planung: Notiere deinen Code hier:



Vielleicht hilft es dir, zuerst **im Text zu unterstreichen**, was du beim Programmieren beachten musst.

3. Teste dein Programm: Hilf Kater Karlo durch das Labyrinth, damit er Maus Minnie fangen kann.
4. Reflexion: Etwas funktioniert noch nicht? Suche den Fehler und überarbeite dein Programm.

5. Hausaufgabe: Rätselspaß

Hier findest du 6 verschiedene Rätsel. Du darfst dir aussuchen, welche zwei Rätsel du lösen möchtest. Löse die Rätsel, indem du zu jedem einen passenden Code programmierst. Hake die Rätsel ab, die du schon gelöst hast.

Rätsel 1: Wenn die Scratch-Katze und ein anderes Tier zusammenstoßen, sagt einer von beiden „Entschuldigung“.

Rätsel 2: Wenn die Scratch-Katze etwas Blaues berührt, hört man den Klang „Big Boing“. Wenn die Katze etwas Rotes berührt, hört man den Klang „Bonk“.

Rätsel 3: Immer wenn die Scratch-Katze ein Geräusch hört, wechselt sie die Farbe.

Rätsel 4: Immer wenn die Scratch-Katze den Hund berührt, dreht sich der Hund um und läuft vor der Katze davon.

Rätsel 5: Wenn du den Mauszeiger bewegst, folgt ihm die Scratch-Katze, aber berührt den Mauszeiger nicht.



Rätsel 6: Wenn du auf eine Figur drückst, führen die anderen Figuren einen Tanz vor.

Notiere hier deine Codes zu den Rätseln. Markiere diejenigen Code-Blöcke, mit denen du Sensoren programmierst rot und die mit denen zu Aktoren programmierst blau. Schreibe daneben, warum es sich hierbei um den Code-Block eines Sensors oder Aktors handelt.

Mein Code zu Rätsel __:

Mein Code zu Rätsel __:



				
Ich kann mit eigenen Worten erklären, was ein Sensor ist.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich kann mit eigenen Worten erklären was ein Aktor ist.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich weiß, wo ich Sensoren in Scratch finde.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich weiß, wo ich Aktoren in Scratch finde.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich kann Sensoren in Scratch programmieren.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich kann Aktoren in Scratch programmieren.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ich kann erklären, wo ich in meinem Alltag Sensoren und Aktoren finden kann.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

IV. Tippkarten

Tippkarte grün: Fang mich doch!

Kater Karlo will Maus Minnie fangen! Immer, wenn der Kater Minnie rufen hört „Fang mich doch!“, rennt er in ihre Richtung. Doch immer, wenn Kater Karlo die Maus gerade berührt und ruft „Hab ich dich!“, entwischt sie ihm wieder und rennt weiter.

Nun bist du an der Reihe: Überlege, wie du Karlo und Minnie auf Scratch so programmieren kannst, dass sie Fangen spielen.

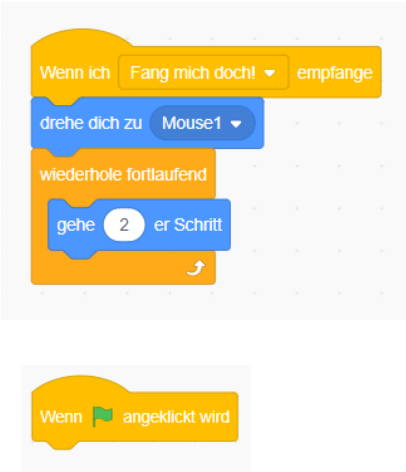

Achtung: Nur Karlo versucht Minnie zu fangen!

Tipp 1:

Spielt mit deinem Partner das Fangen von Karlo und Minnie nach. Achtet genau darauf, was ihr macht. Was müsst ihr dann auch beim Programmieren beachten?

Tipp 2:

Sehe dir die benutzten Code-Blöcke für Sensoren und Aktoren aus Aufgabe 1 an. Überlege, wie du diese sinnvoll einsetzen kannst. Starte so:

Kater Karlo	Maus Minnie
	

Tippkarte rot: Fang mich doch!

Kater Karlo will Maus Minnie fangen! Immer, wenn der Kater Minnie rufen hört „Fang mich doch!“, rennt er in ihre Richtung. Doch immer, wenn Kater Karlo die Maus gerade berührt und ruft „Hab ich dich!“, entwischt sie ihm wieder und rennt weiter.

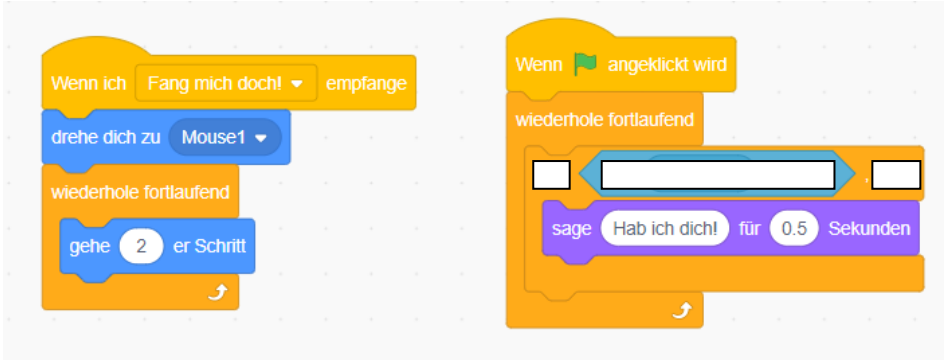
Nun bist du an der Reihe: Überlege, wie du Karlo und Minnie auf Scratch so programmieren kannst, dass sie Fangen spielen.

Achtung: Nur Karlo versucht Minnie zu fangen!

Tipp 3:

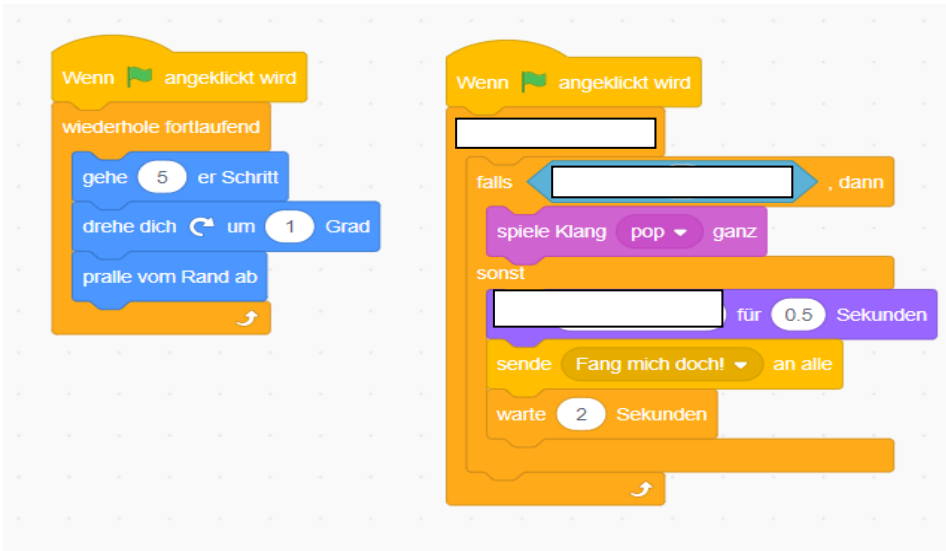
Hier findest du eine mögliche Lösung. Sie hat allerdings noch Lücken. Überlege, welche Blöcke du in die Lücken einsetzen kannst.

Kater Karlo



The Scratch code for Kater Karlo consists of two event-driven blocks. The first block is triggered by the 'Fang mich doch!' message and contains three actions: 'drehe dich zu Mouse1', a 'wiederhole fortlaufend' loop containing 'gehe 2 er Schritt'. The second block is triggered by 'angeklickt wird' and contains a 'wiederhole fortlaufend' loop with a 'sage Hab ich dich! für 0.5 Sekunden' block. There are empty input fields in the loops.

Maus Minnie



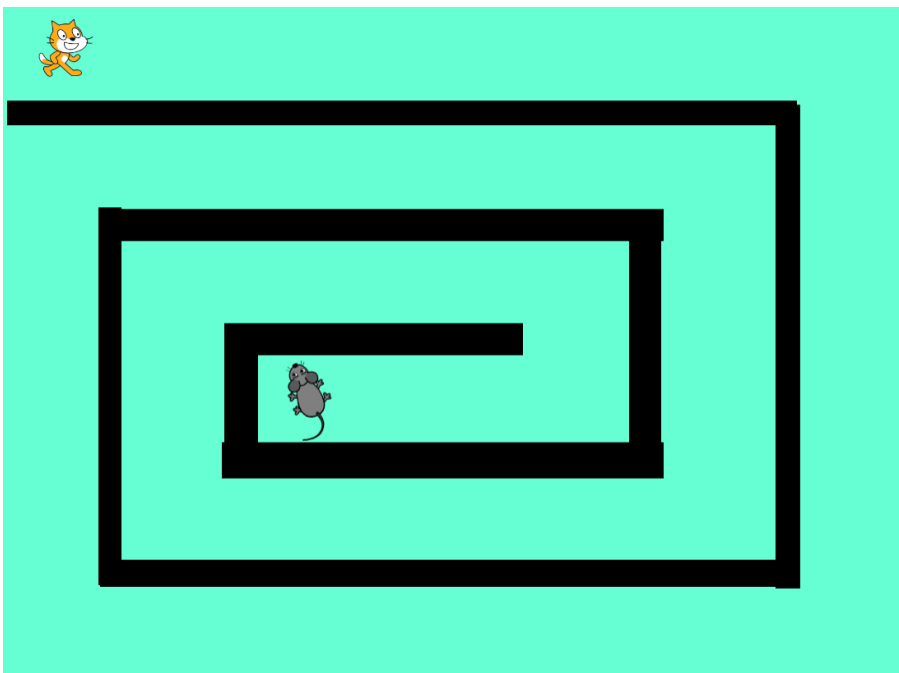
The Scratch code for Maus Minnie consists of two event-driven blocks. The first block is triggered by 'angeklickt wird' and contains a 'wiederhole fortlaufend' loop with 'gehe 5 er Schritt', 'drehe dich um 1 Grad', and 'pralle vom Rand ab'. The second block is triggered by 'angeklickt wird' and contains a 'falls' conditional block. The 'falls' block has an empty input field and contains 'spiele Klang pop ganz'. The 'sonst' block contains an empty input field, 'sende Fang mich doch! an alle', and 'warte 2 Sekunden'. There are empty input fields in the 'falls' and 'sonst' blocks.

Tippkarte grün: Das Labyrinth

Kater Karlo hat Maus Minnie immer noch nicht gefangen. Nun ist Minnie auch noch in ein Labyrinth gelaufen. Immer, wenn Kater Karlo die Wände berührt, ruft er laut „Miau“ und muss zum Anfang des Labyrinths zurückkehren. Er hört erst mit der Suche auf, wenn er Maus Minnie gefangen hat. Du kannst ihm dabei helfen, indem du deinen Code so programmierst, dass Kater Karlo deinem Mauszeiger folgt.

Tipp 1:

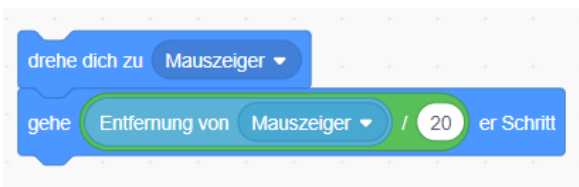
Lies dir den Aufgabentext aufmerksam durch. Unterstreiche im Text, was du beim Programmieren beachten musst. Du kannst auch das Bild des Labyrinths nutzen, um dir wichtige Programmierhinweise zu notieren.



Tipp 2:

Sehe dir die benutzen Blöcke zur Programmierung von Sensoren und Aktoren aus den Aufgaben 1 und 3 an. Überlege, wie du diese sinnvoll einsetzen kannst.

Dieser Programm-Code kann dir helfen, Kater Karlo so zu programmieren, dass er mit etwas Abstand dem Mauszeiger hinterherläuft:

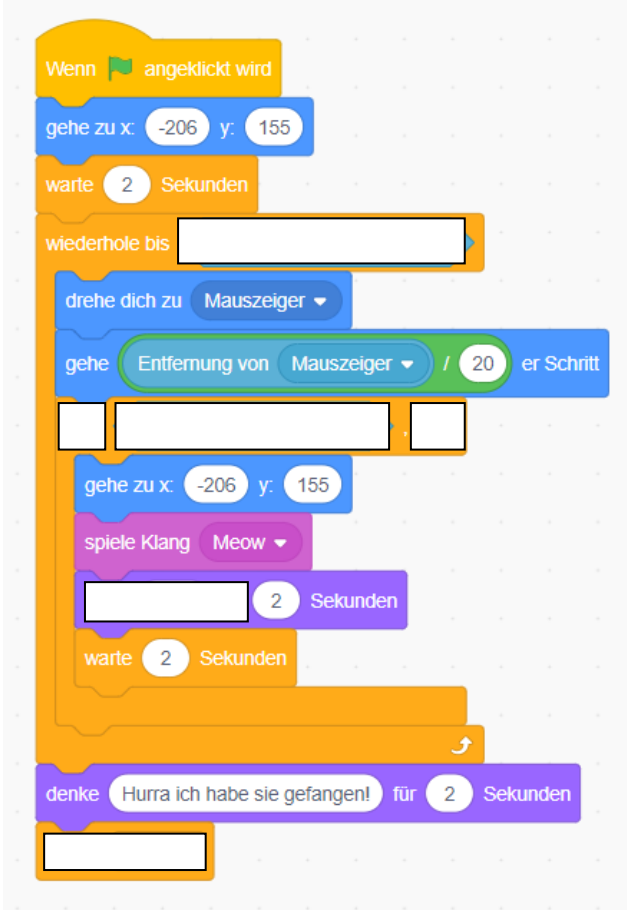


Tippkarte rot: Das Labyrinth

Kater Karlo hat Maus Minnie immer noch nicht gefangen. Nun ist Minnie auch noch in ein Labyrinth gelaufen. Immer, wenn Kater Karlo die Wände berührt, ruft er laut „Miau“ und muss zum Anfang des Labyrinths zurückkehren. Er hört erst mit der Suche auf, wenn er Maus Minnie gefangen hat. Du kannst ihm dabei helfen, indem du deinen Code so programmierst, dass Kater Karlo deinem Mauszeiger folgt.

Tipp 3:

Hier findest du eine mögliche Lösung. Sie hat allerdings noch Lücken. Überlege welche Blöcke du in die Lücken einsetzen kannst.



The image shows a Scratch code snippet for a maze game. The code is as follows:

```
Wenn [ ] angeklickt wird
  gehe zu x: -206 y: 155
  warte 2 Sekunden
  wiederhole bis [ ]
    drehe dich zu Mauszeiger
    gehe Entfernung von Mauszeiger / 20 er Schritt
    [ ] [ ]
    gehe zu x: -206 y: 155
    spiele Klang Meow
    [ ] 2 Sekunden
    warte 2 Sekunden
  denke Hurra ich habe sie gefangen! für 2 Sekunden
  [ ]
```

The code is written in Scratch and includes several empty blocks for user input. The visible blocks are:

- Wenn [] angeklickt wird
- gehe zu x: -206 y: 155
- warte 2 Sekunden
- wiederhole bis []
- drehe dich zu Mauszeiger
- gehe Entfernung von Mauszeiger / 20 er Schritt
- [] []
- gehe zu x: -206 y: 155
- spiele Klang Meow
- [] 2 Sekunden
- warte 2 Sekunden
- denke Hurra ich habe sie gefangen! für 2 Sekunden
- []

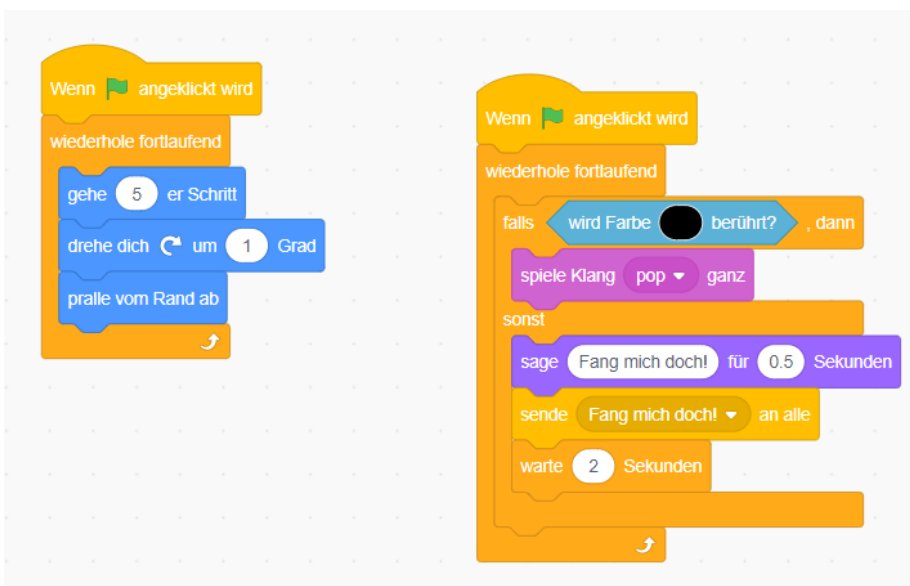
V. Musterlösungen

Aufgabe 3: Fang mich doch!

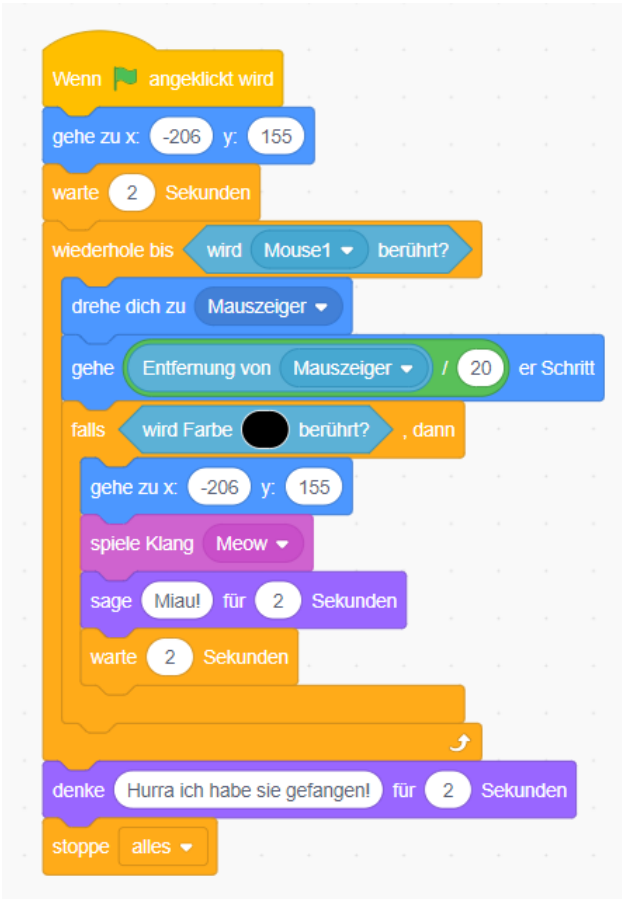
Kater Karlo



Maus Minnie



Aufgabe 4: Das Labyrinth



```
Wenn [ ] angeklickt wird
  gehe zu x: -206 y: 155
  warte 2 Sekunden
  wiederhole bis [wird Mouse1 berührt?]
    drehe dich zu Mauszeiger
    gehe Entfernung von Mauszeiger / 20 er Schritt
    falls [wird Farbe berührt?] , dann
      gehe zu x: -206 y: 155
      spiele Klang Meow
      sage Miau! für 2 Sekunden
      warte 2 Sekunden
  denke Hurra ich habe sie gefangen! für 2 Sekunden
  stoppe alles
```

The image shows a Scratch script for a maze game. The script starts with a 'When clicked' event, followed by a 'Go to x: -206 y: 155' block. It then includes a 'wait 2 seconds' block, a 'repeat until' loop with the condition 'is mouse1 touched?'. Inside the loop, there are three blocks: 'turn to mouse cursor', 'go distance of mouse cursor / 20 steps', and an 'if' block. The 'if' block checks 'is color touched?' and, if true, contains a 'go to x: -206 y: 155' block, a 'play sound Meow' block, a 'say Miau! for 2 seconds' block, and a 'wait 2 seconds' block. After the loop, there is a 'think Hurra ich habe sie gefangen! for 2 seconds' block, and finally a 'stop all' block.