Tagung Mobiles Lernen mit Tablets und Co - #molol17

9.-10.3. 2017

Oldenburg

Referentin: Christiane Schicke

Programmieren in der Grundschule

1. Warum Programmieren in der Grundschule?

Ob bereits in der Grundschule Programmieren unterrichtet, sogar als Pflichtfach eingeführt werden soll, wird zur Zeit an verschiedenen Stellen heftig diskutiert. Tatsache ist, dass mehrere europäische Länder diese Inhalte bereits eingeführt haben (UK, Estland, Finnland, Schweden...), bzw. dieses in naher Zukunft planen. Ob, wann und in welchem formalen Rahmen Programmieren unterrichtet werden soll – dies kann an dieser Stelle nicht diskutiert werden.

Aus subjektivem Erleben weiß ich, das Kinder sehr früh dazu in der Lage sind, Programme - selbst in einer Scriptsprache - zu schreiben, wenn es sie interessiert. Heute, nach der Entwicklung visueller Editoren wie Scratch oder Snap!, ist es umso einfacher.

Ich habe mich entschieden, meine Schüler und Schülerinnen zu einfacher Programmierung hinzuführen, weil ich es für sinnvoll halte, denn

- Computer begegnen den Kindern in der Arbeitswelt der Eltern und Zuhause, er ist präsent in der Lebenswelt der Kinder, und nimmt teilweise eine sehr (zu) prominente Rolle in ihren Freizeitaktivitäten ein.
- Computer sind geheimnisvoll, denn selbst die Erwachsenen wissen nicht, wie sie funktionieren.
- Kinder, die programmieren, erleben sich als selbstwirksam gegenüber der Technik. Sie wechseln von einer überwiegend passiven Nutzung zum aktiven Gestalten.

- Programmieren fördert analytisches Denken: Handlungszusammenhänge müssen in Teilschritte zerlegt werden.
- Programmieren erfordert genaues Arbeiten und fördert das Durchhaltevermögen.
- Programmieren lehrt eine Arbeitskultur des Probierens, Testens, Verbesserns und der Kollaboration.
- Freie Programmierung ist ein kreativer Prozess.

Und: Viele naturwissenschaftliche und mathematische Zusammenhänge können beim Programmieren spielerisch erarbeitet, simuliert, ausprobiert werden. Wer zum Beispiel Kreise malen will, lernt Kosinus und Sinus nebenbei. Wer Kanonenkugeln fliegen lassen will, wendet das das Eulerverfahren für Differentialgleichungen an (und versteht ziemlich gut, wie Beschleunigung, Geschwindigkeit und Ort zusammenhängen). Dabei ist es höchst überraschend, zu welchem tiefem Verständnis Kinder dabei kommen. Ob das Programm letztlich funktioniert, ist objektiv bewertbar, selbst für Kinder. Sie haben ein Erfolgserlebnis, das nicht von einer Bewertung durch Erwachsene (oder andere Kinder) abhängt.

Die nun folgenden Inhalte sind nicht wissenschaftlich erforscht oder begleitet, sondern geben einfach meine eigenen Grundschullehrerinnenüberlegungen und -gedanken wieder. Die Beispiele wurden in verschiedenen Klassen und Jahren durchgeführt und ergeben nun eine nette kleine Unterrichtseinheit, die über einen längeren Zeitraum verteilt werden kann. Es hat funktioniert, es war und ist nachhaltig, es hat allen Beteiligten Spaß gemacht und Lernerfolge erzielt. Also – einfach mal ausprobieren!

Grundprinzip:

Dieses Grundprinzip versuche ich bei jeder „Untereinheit“ soweit wie möglich umzusetzen. Gerade wenn die einzelnen Einheiten zeitlich weit auseinanderliegen, kann auf diese Weise Wissen reaktiviert werden. Es ermöglicht auch, die Einheiten einzeln und unabhängig voneinander durchzuführen.

Grundprinzip des Programmierens mit Grundschülern:
(angelehnt an das EIS-Prinzip[1] im Mathematikunterricht)
Eigenes Erleben → zunehmende Abstraktion → Externalisierung: Programm muss auf die Umwelt zurückwirken

Eigenes Erleben:

Zuerst sollten die Schülerinnen und Schüler selbst aktiv erleben, was Programmieren bedeutet – , die Kinder programmieren sich gegenseitig, die Programmierung wird handelnd umgesetzt, die Kinder „erfahren“, welche Auswirkungen die Programmbefehle jeweils haben und lernen grundsätzliche Vogehensweisen des Programmierens kennen.

1 EIS-Prinzip: Variation der Repräsentation- bzw. Darstellungsformen enaktiv, ikonisch und symbolisch nach Jerôme Bruner. Meist wird das Prinzip aufbauend verstanden, ein Wechsel zwischen den Repräsentationsformen auch im ist aber auch späteren Lernprozess wichtig und führt zu einem tieferen Verständnis und stärkt lexibles Denken.

Zunehmende Abstraktion:
Von der enaktiven Ebene wechseln wir auf die ikonische, in höheren Klassen vielleicht auch auf die symbolische (Scriptsprache). Statt des Klassenraums nutzen wir nun Programmierumgebungen mit visuellen Editoren als App oder am PC als „Spielfeld“.

Externalisierung:
Programme, die auf ikonischer oder symbolischer Ebene erstellt wurden, wirken wieder auf die Umwelt der Kinder zurück - sei es, dass ein Roboter in Bewegung versetzt wird oder eine kleine Platine zu leuchten und zu brummen beginnt. Die Kinder erleben, dass ihre Programmierung etwas bewirken kann und bekommen ein Verständnis dafür, wie ihre zunehmend automatisierte Welt funktioniert.

Curriculare Anbindung:

Wie legitimiere ich es, diese Inhalte zu unterrichten?
In Niedersachsen bietet bereits das jetzige Kerncurriculum Sachunterricht Anlass, mit Schülerinnen und Schülern der Grundschule am PC zu recherchieren und Produkte zu erstellen.
Derzeit (Stand März 2017) wird ein neues Curriulum entwickelt, das Informatik in die Perspektive Technik explizit mit einbezieht. Dort steht:

Kerncurriculum Sachunterricht
Perspektive Technik
„MINT-Bildung
Ausgehend von den Fragen der Kinder steht im Sachunterricht das aktiv entdeckende, handlungsorientierte Lernen im Vordergrund. Indem die Schülerinnen und Schüler subjektiv bedeutsame Problemstellungen bearbeiten und Arbeitsergebnisse präsentieren, kann anwendungsorientiertes, anschlussfähiges Wissen in den MINT-Bereichen Mathematik, Informatik, Naturwissenschaften und Technik aufgebaut werden. Ziel ist die Überprüfung und Weiterentwicklung naturwissenschaftlicher Präkonzepte sowie der Aufbau tragfähiger Lernmotivation (vgl. Empfehlungen zur Arbeit in der Grundschule, KMK, 2015)“
http://db2.nibis.de/1db/cuvo/datei/kc_su_n-line.pdf S. 14 f

Auch das Kerncurriculum Mathematik bietet Ansatzpunkte. So lassen sich die in den Beispielen gezeigten Inhalte in die Bereiche „Orientierung im Raum“ (Gittternetz, Koordinatensystem), und „einfache Algorithmen“ einordnen. Im Programmieren, besonders in der Robotik, wird Mathematik bedeutungsvoll, sie wird angewandt um „echte“[2] Probleme zu lösen.

Zeiträume schaffen:

Wann soll man denn nun auch noch Informatik unterrichten?
Ich habe für mich folgende Zeitfenster gefunden:

systematischer Unterricht:
- als Unterrichtseinheit im Sachunterricht (spiralcurricular denkbar)
- im Wochenplan, nachdem Grundlagen gemeinsam gelegt wurden
- in AGs, im Rahmen der Begabugsförderung, im Hochbegabtenverbund

unsystematisch:
- in Freiarbeit und freier Spielzeit (Geräte und Apps/Programme stehen zur freine Verfügung)
- in Vertretungsstunden

Auf der Suche nach dem eigenen Curriculum:

Und was will ich unterrichten?
Ich habe mir die Themen zusammengesucht, von denen ich denke, dass sie im Bereich Programmieren wichtig sind, andererseits aber auch die Kinder interessieren und in eine enaktive Repräsentationsform gebracht werden können.

Was ist mir wichtig?
Wissen über den Computer (Hardware, von Neumann-Architektur)
unplugged Programming
Automatenprogrammierung
Robotik

2 Zugegeben: Ohne diese UE hätten die Kinder die Probleme nicht, aber es sind zumindest Probleme, die sie aus sich heraus lösen wollen, zu deren Bewältigung sie hoch motiviert sind

„Hallo Welt“ - der traditionelle, übliche Einstieg ins Programmieren - funktioniert nicht in der Grundschu1e!

Zum Bereich „Informatik“ gehören noch viele weitere Bereiche, die im Folgenden **nicht** aufgegriffen werden[3], wie Datensouveranität, digital Citizenship und anderes, das im Kerncurriculum Informatik des Landes Niedersachsen formuliert ist. Vieles davon lässt sich herunterbrechen, so dass es auch in der Grundschule ansatzweise behandelt werden kann. Umgang mit Chat-Räumen und Social Media sind Themen, die auch schon Grundschüler betreffen und frühzeitig in entsprechenden Kontexten (wie z. B. im Sexualkundeunterricht) behandelt werden müssen.

Der Umgang mit PC und Tablet beginnt bei uns fast mit dem ersten Schultag. In 3 von vier Grundschulklassen steht ein Interaktives Whiteboard zur Verfügung (das vierte kommt…), jede Klasse hat mindestens einen PC mit Internetzugang, dazu offline PCs, wenn gewünscht. Der halbe Klassensatz Tablets genügt derzeit für unsre Einsatzzwecke.

Einstieg:

- Einfache Bedienkompetenz ab Klasse 1
  (Anschalten, Ausschalten, CD-Einlegen, Programme öffnen und ordnungsgemäß beenden, Dateien speichern, Umgang mit OO4Kids, Malprogrammen)
- Erfassen von Vorerfahrungen
  - Was ist ein Computer?
  - Wo gibt es Comnputer bei euch zuhause? Was macht ihr damit?
  - Wie funktioniert ein Computer?

- Basics: Woraus besteht ein Computer?
  - Prinzip Eingabe – Verarbeitung/Speicher – Ausgabe (von Neumann Architektur)
  - für viele Kinder ist nicht klar, dass der Monitor NICHT der Computer ist!
  - Peripheriegeräte

Diese Inhalte lasse sich mit Bilder- und Kinderbüchern ergänzen, es existieren auch bereits fertige Werkstätten zu diesem Thema, die von Verlagen angeboten werden.

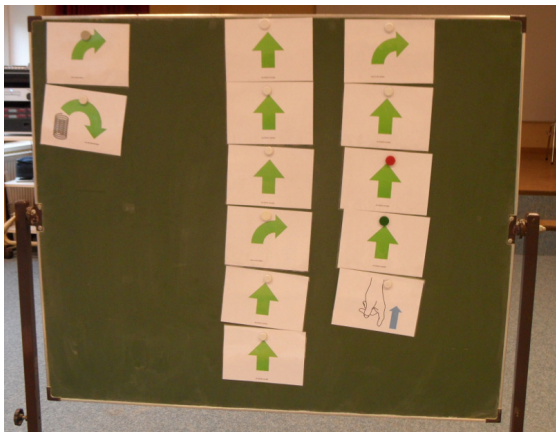
3 Schließich heißt der Workshop nicht „Informatik in der Grundschule“, sondern ist bewusst auf das Programmieren reduziert.

Dekonstruktion

Einen alten PC auseinandernehmen lassen!

- Perspektive „Technik“ im Sachunterricht
- Entzauberung des Gerätes
- Einzelteile zuhause zerlegen lassen – dort sind oft besseres Werkzeug und ein neugieriger, hilfreicher Erwachsener zu Hand. Die vollständig zerlegten Teile werden wieder mit in die Schule gebracht und den Mitschülerinnen und -schülern vorgestellt. So kommt es zu
- Referaten + Präsentationen, denkbar ist auch eine Ausstellung innerhalb des Schulhauses.

Unplugged Programming



Programmieren ohne Computer

- Anforderung: Zerlegen von Handlungsabläufen in Teilschritte
- Überlegen: Welche Handlungsoptionen werden benötigt?
- Kinder programmieren sich gegenseitig
    - schwer: nur ausführen, was genannt wird! Nicht selbst denken!
    - auch fehlerhafte Programme werden soweit wie möglich durchgeführt

Erfassen der Vorerfahrungen:
Was ist ein Roboter? Wie funktioniert er? Wer hat einen Roboter zuhause (Staubsauger, Rasenmäher, Spielzeug...)?
Sehr schnell kamen die Kinder stets zu den Aussagen:
„Ein Roboter tut nur, was man ihm sagt.“
„Ein Roboter muss programmiert werden.“

Vorstellungen: Ich wünsche mir einen Roboter, der….
→ z. B. FÜ mit Kunst: einen Wunschroboter malen oder bauen
→ z. B. Fü mit Deutsch: eine Robotergeschichte schreiben oder lesen („Der Erziehungsroboter“)

Einstieg in die Programmierung einfacher Automaten:

- Programm planen: Welche Befehle benötigen wir? Was muss der Roboter „können“?
- Befehlskarten aufhängen/auslegen
- Programm ablaufen lassen: Programmierer, Steuermann und Roboter
  Das „Programm“ kann gemeinsam in der Gruppe oder von einzelnen erstellt werde. Wir haben die ersten Programme immer gemeinsam erstellt. Bei den nächsten Aufgaben durften einzelne Kinder die Programmschritte auswählen. So gelangt man zu mehreren unterschiedlichen Lösungen, die miteinander verglichen werden können und Fehlern.
  Bei uns bricht der Roboter zusammen, sobald er – vom Programm geführt – das Spielfeld verlassen muss und ruft „Error! Error!“. Schwer fällt es den Kindern, bei einer falschen Programmierung am Ziel vorbei zu laufen, besonders, wenn es sich um einen Keksteller handelt... Aber es gibt je einen neuen Versuch!
  Rolle Programmierer: erstellt das Programm
  Rolle Steuermann: liest das Programm Schritt für Schritt vor und zeigt die Befehle an der Tafel mit
  Rolle Roboter: führt die angesagten Befehle (und nur die!) aus.
  Möchte man noch mehr Kinder aktiv in jedem Durchlauf beteiligen, kann ein Kind die Programmbefehle aufhängen, ein anderes den Programmablauf mitzeigen, während der Steuermann nur vorliest.

- Programm optimieren – verschiedene Programme vergleichen
    - kurz / lang
    - geschickt / umständlich
    - alles, was zum Ziel führt, ist „richtig“ → neue Erfahrung für die meisten Kinder!

Vom einfachen Parcours zum schwereren
→ einfache Wege laufen zum Ziel
→ mehrere Wege zum Ziel finden und vergleichen
→ Hindernisse umgehen oder überspringen

mehr Ideen zu „CS Unplugged“:
CS Education Research Group at the University of Canterbury, NZ (aka "Department of Fun Stuff"). Csunplugged.org → Activity-Book. Stand 2015
Webseite und Inhalte befinden sich in Überarbeitung → voraussichtlich Ende 2017 fertig!

Abstraktion: Übertragen des Gelernten auf ein Computerprogramm
In meiner Klasse nutzen wir die Programme

- Run Marco!
- Lightbot

→ beide Programme bieten eine visuelle Programmierumgebung
→ es werden ähnliche Symbole wie in der „unplugged Programmierung“ verwendet → hoher Wiedererkennungswert!

Run Marco!:

- Vorbereitung des Programmierens in Scratch und ähnlicher Programmiersprachen, da scratchähnliche Bausteine verwendet werden → anschlussfähig!
- Nach kurzer Einführung zum Gebrauch der Programmbausteine können Kinder selbstständig damit arbeiten (Klasse 1)
- Inhalte:
    - einfache Wiederholungsschleifen
    - Wenn – dann - Abfragen
    - kostenlos erhältlich für IOS – Android - PC

https://www.allcancode.com/web

LightBot:

- eigene Darstellung der Befehle, ähnlich den gebastelten Pfeilkarten
- Inhalte
  - Einsatz von Unterprogrammen und Rekursionen → konnte von Drittklässlern bewältigt werden! Ein Erstklässler hat sich die Benutzung der Unterprogramme selbstständig erarbeitet!
  - Erklärung neuer Programmbausteine leider nur auf Englisch
  - kostenlos erhältlich für IOS – Android – PC im Rahmen der Hour of Code
  - App mit weiteren Programmieraufgaben kostenpflichtig

https://lightbot.com/

Ablauf:

Gemeinsamer Einstieg über das Interaktive Whiteboard
gemeinsames Erarbeiten der ersten Level
am Tablet: eigenständige Wiederholung der ersten Level,
Weiterarbeit, solange die Zeit/Konzentration reicht
möglichst immer in Partnerarbeit! „Pair Programming“ - Driver and Navigator
https://www.iste.org/explore/articleDetail?articleid=221 (International Society for Technology and Education)
https://cs.brown.edu/courses/csci0170/content/docs/pair-programming.pdf

Einen guten Einstieg ins Programmieren bietet auch das Angebot der „Hour of Code“

- kostenlose Programme zum Einstieg ins Programmieren
- sortiert nach Alter, beginnend auf Kindergartenniveau
- ausprobieren, was einem zusagt…
- immer wieder neue Ideen, oft angelehnt an Disney Blockbuster.. (Anna und Elsa / Moana)
- Programmieren als „Challenge“ - vorgegebene Aufgaben müssen gelöst werden, um ins nächste Level zu gelangen. Nach und nach entsteht dabei ein immer komplexerer Code.
- oft wenig flexibel! „Funktionierende“ Lösungen werden nicht immer akzeptiert.

Kara – Programmierung endlicher Automaten

„Kara, der programmierbare Marienkäfer“ stammt aus dem Informatikunterricht der Sek1 und beschäftigt sich mit der Programmierung endlicher Automaten. Diese Programmierumgebung entstand 1999 bis 2003 an der ETH Zürich, erhielt zwei Auszeichnungen[4] und wird laufend weiterentwickelt.

Die Programmelemente – Marienkäfer, Kleeblätter, Bäume und Pilze kommen den Grundschulkindern sehr entgegen, das Programmieren wird anschaulich und spielerisch. Kara kann über die visuelle Umgebung in Java hinaus aber mit etlichen anderen Sprachen programmiert werden. Damit ist die Beschäftigung mit dem Program anschlussfähig bis in die Berufsausbildung hinein, für die Kara einst auch mit entwickelt wurde.

Einen Überblick über die Möglichkeiten, die Kara älteren Schülerinnen und Schülern bis hin zu Fachhochschulstudenten bietet, findet man auf der Seite der ETH Zürich
http://www.swisseduc.ch/informatik/karatojava/kara/

Kara läuft nur auf PCs, dort aber auch offline und ist kostenlos.

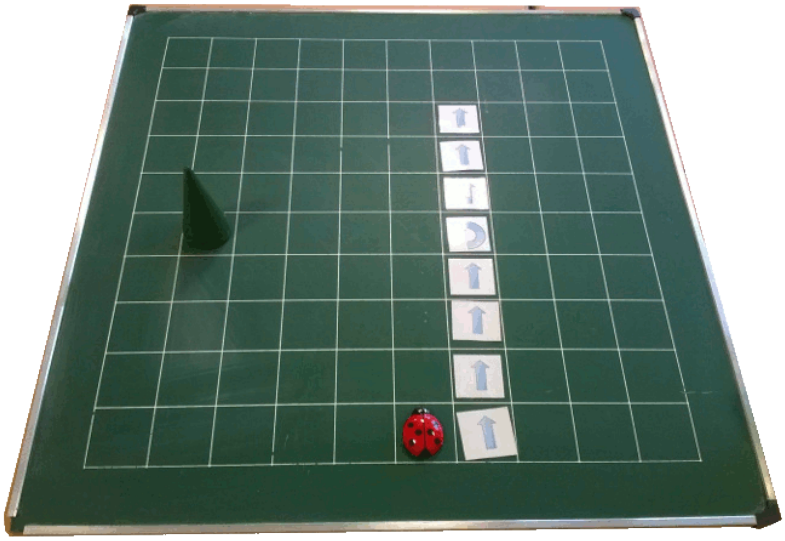
Mit Kara gelangt ein neuer Aspekt in den Programmierunterricht:
Kara besitzt „Sensoren“ und kann somit auf ihre Umgebung reagieren. Es müssen Fallunterscheidungen eingeführt werden: „Was muss Kara machen, wenn…?“
Auch hier versuche ich, möglichst viele Kinder gleichzeitig aktiv einzubinden und verwende die drei Rollen Programmierer, Navigator und Steuermann.

Eingesetztes Material:

- Überhangtafel mit Kästchen
- Magnete in Marienkäfer-Form
- Bäume, z. B. aus der Weihnachtsdeko
- Kleeblätter (gedruckt und laminiert in Kästchengröße)
- Programmbefehle (Pfeilkarten wie in den unplugged-Übungen plus die für Kara zusäzlich nötigen Befehle „ablegen“ und „aufnehmen“)

4 2000 gewann Kara als Diplomarbeit Raimond Reicherts den Fritz-Kutter-Preis zur Förderung praxisbezogener Informatik mit dem Titel „Ein spielerischer Einstieg ins Programmieren: Kara, der programmierbare Marienkäfer“. Die Weiterentwicklung KaraToJava erhielt 2002 den European Academic Software Award im Bereich „Computer science“. Quelle: https://de.wikipedia.org/wiki/Kara_(Programmierumgebung) [Stand 25.3.2017]

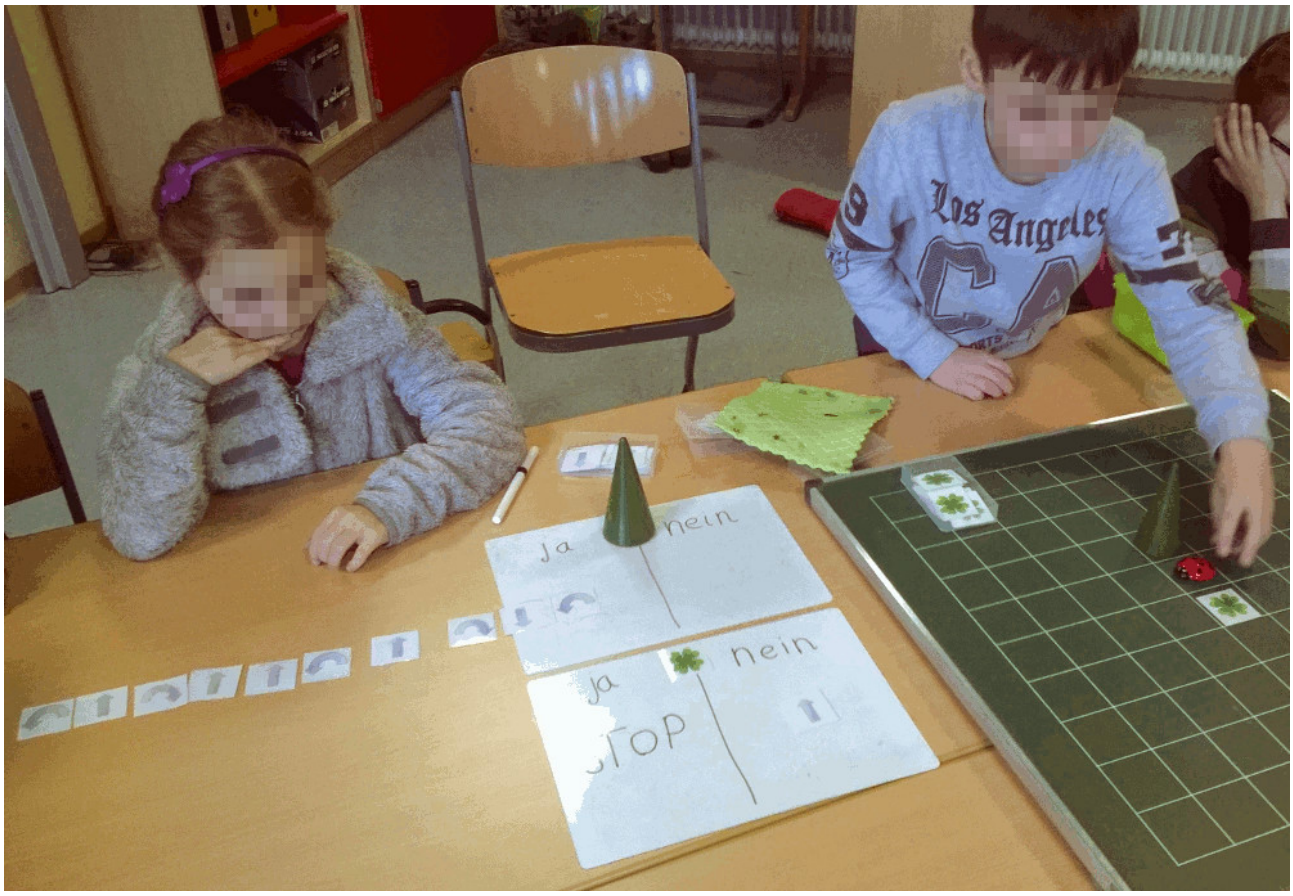
- Mini-Whiteboards[5], um die Fallunterscheidungen zu notieren (besser noch: vorbereitete Tabellen, in die man Programmbefehle legen kann)
- Pilze tauchen in den Anfangsaufgaben nicht auf, man kann sie aber im Floristikbedarf besorgen oder aus Knete oder Holz selbst herstellen, falls nötig



Auch mit Kara, dem Marienkäfer, sollen die Kinder das Programmieren zuerst handelnd erleben. In der Großgruppe werden Kara und ihre Welt vorgestellt.
Als erstes löst Kara ähnliche Aufgaben wie Marco oder der Lightbot – er soll sich zu einem vorgegebenen Ziel hinbewegen. Anders als bei den bereits bekannten Programmierumgebungen benötigt man zum Geradeauslaufen hier nur einen einzigen Befehl, da das Programm stets wieder von vorne beginnt, solange es nicht einen expliziten Stop-Befehl gibt. Diese Darstellung ist für die Kinder neu.
Nun werden die “Sensoren“ eingeführt. Kara „sieht“, ob ein Kleeblatt unter ihr liegt oder nicht. Wenn unter ihr ein Kleeblat liegt, soll sie es aufnehmen, wenn nicht, eines ablegen. Die Fallunterscheidung wird auf einem Mini-Whiteboard als zweispaltige Tabelle notiert und die Programmbefehle entsprechend darauf abgelegt. Wichtig ist nun die Sprachregelung für den „Steuermann“. Er muss die Fallunterscheidung verbalisieren. Hier sollte man als Lehrkraft die ersten Durchläufe modellhaft selbst durchführen. Mit den Erstklässlern hat sich bewährt:
„Kara überlegt: Liegt ein Kleeblatt unter mir?“ (Sensorabfrage)
„Ja – also …..“ / „Nein - also…..“ und die entsprechenden Befehle im Programmablauf wieder mitzuzeigen. Auch der Einsatz von zwei Sensoren – also vier Fällen – war für die Kinder kein Problem.

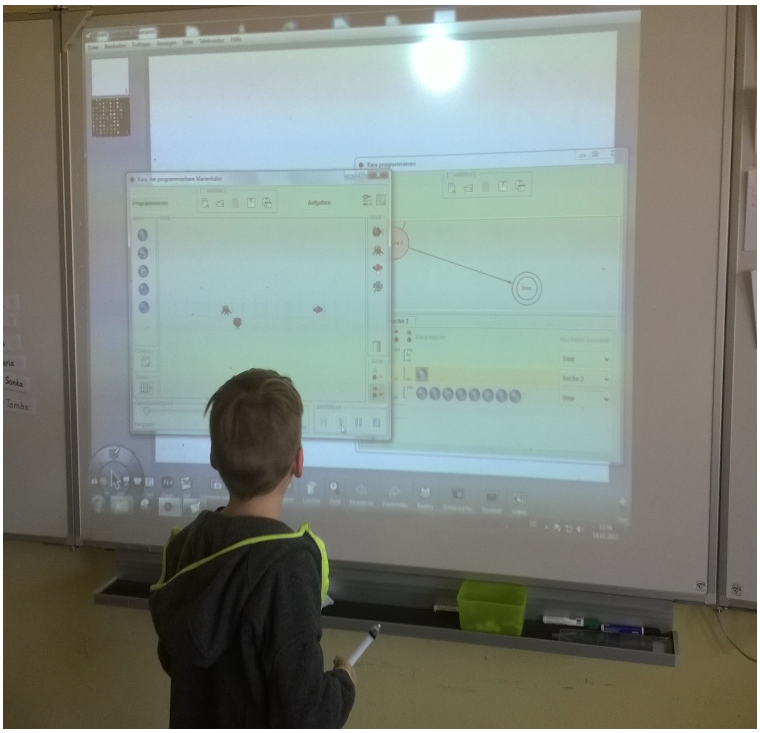
5 z. B. diese: https://www.educationsupplies.co.uk/stationery-supplies/whiteboards-and-accessories/mini-whiteboards/value-a4-superlight-mini-whiteboard-kits

Drittklässler waren - nach entsprechender Vorbereitung – in der Lage, Aufgaben, die in der Programmierumgebung vorgegeben sind, in Kleingruppen selbstständig zu lösen. Eine kleine Gruppe programmierte später am PC auch Lösungen für Aufgaben, bei denen zwei Zustände definiert werden mussten. Für die Erstklässler habe ich nur Aufgaben mit einem Automaten-Zustand ausgesucht.



Der nächste Schritt ist die Einführung der Programmoberfläche. Gemeinsam haben wir am IWB die ersten Aufgaben gelöst, die den Kindern bereits aus dem „Spiel“ bekannt waren. So konnten sie sich ganz auf die neue Darstellungsweise konzentrieren. Die Oberfläche von Kara sieht relativ kompliziert aus, da sie aus mehreren Fenstern besteht. Die Darstellung des Graphen habe ich zu diesem Zeitpunkt nicht thematisiert – die Kinder haben sie anscheinend ausgeblendet, es fragte keiner danach. Für viele der Erstklässler war an diesem Punkt die Aufnahmefähigkeit erreicht.
Wer mag, kann nun am PC in der Freiarbeit weiterarbeiten und sich (oder einem Freund bzw. einer Freundin) auch eigene Aufgaben in eigenen „Welten“ stellen. Kara bietet dazu einen Editor an. Schön wäre es, für die Freiarbeit eine Aufgabenkartei zur Hand zu haben, an der die Kinder arbeiten können, wenn sie möchten. Er könnte aus vereinfachten Aufgaben des Aufgabenpools bestehen, der mit der Programmierumgebung mitgeliefert wird, oder aus selbst erstellten Aufgaben.

Auch die „analoge“ Fassung, das „Marienkäferspiel“, bleibt vorerst zur individuellen Nutzung in der Klasse.



Robotik

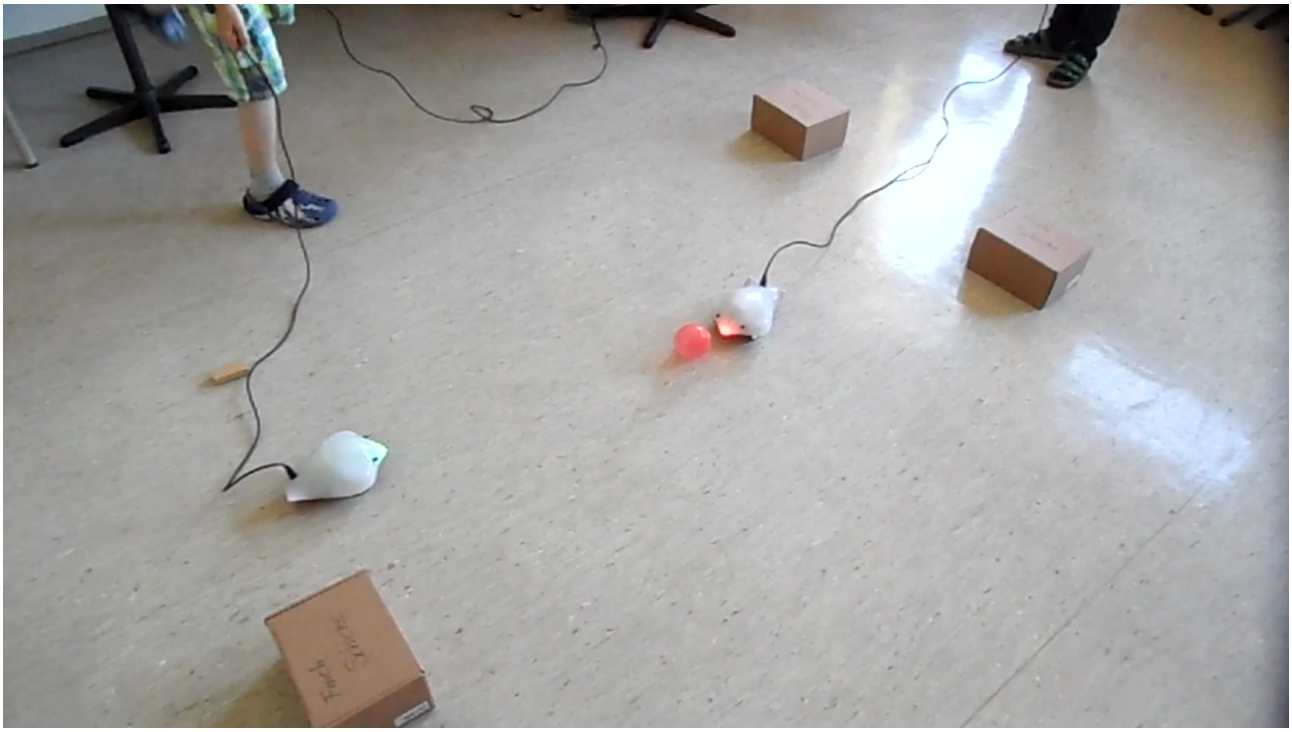
Einen virtuellen Marienkäfer in einer sehr einfachen Welt zu bewegen ist nett, viel schöner ist aber, wenn sich wirklich etwas bewegt. Daher lag es nur nahe, sich nun mit Robotik zu beschäftigen. Wir verfügen über drei Finch-Robots aus Privatbesitz.
Dieser Roboter ist robust und einfach genug für die Grundschule – er überlebte auch bereits mehrere Stürze von der Tischkante – aber dennoch mit seinen Möglichkeiten attraktiv genug für die Sek1. Er bietet ausreichend technische Möglichkeiten für anspruchsvolle Aufgaben, ist aber „niedlich“ genug, um auch jungen pubertierenden Damen zu gefallen.
In der Grundschule ist es ein großer Vorteil, dass dieser Roboter nicht erst zusammengesetzt werden muss – es spart Zeit und die Kinder fokussieren sich auf das Programmieren.
Der Finch besitzt einen Helligkeitssensor, Abstandssensor, Temperatursensor und einen Lagesensor, einen Buzzer, der tonhöhengenau brummen kann und eine ansteuerbare Farb-LED.
2 Räder und ein Stützschwanz machen ihn mobil. Durch die Rille im Schwanz, die einen Stift aufnehmen kann, kann er seinen Laufweg aufzeichnen oder geometrische Muster malen[6].

6 Es gibt auch eine Erweiterung für den Rasberrry Pi für den Nerd im Lehrer.

Der Finch ist kabelgebunden und braucht bei längeren Wegen einen Kabelträger oder eine Kabelträgerin. Praktischerweise haben wir so ein weiteres Kind aktiv eingebunden.

Der Finch Robot kann derzeit in 13 verschiedenen Sprachen programmiert werden, darunter Java, Python, C# und Processing. In der Grundschule nutzen wir Snap! .
Snap! verwendet eine visuelle Programmieroberfläche und ist in vier Leveln verfügbar. In Level eins werden einfache Icons zusammengeklickt, mit denen man die Bewegung, den Buzzer und die LED steuern kann, Level 2 bietet eine Auswahl an Befehlen an, die eng an Scratch angelehnt sind, und lässt die Eingabe einiger Parameter zu. Level 4 bietet dann den vollen Funktionsumfang.

Als Einstieg soll eine Steuerung für den Finch programmiert werden. Die Kinder sind mit der Darstellung der Programmierbefehle aus Run Marco! weitgehend vertraut. Nun kommt mit „Wenn … gedrückt“ ein erster Steuerungsbefehl dazu. Spätestens nach den ersten beiden Vorgaben können die Kinder das Programm vervollständigen, sodass die Bewegungen des Finchs über die Tastatur gesteuert werden können. Nun kann er durch ein selbstgebautes Labyrinth, z. B. aus Bauklötzen fahren. Auch hier werden wieder möglichst viele Rollen an die Kinder vergeben. Hat man mehrere Finchs, kann man z. B. auch ein „Fußballturnier“ organisieren, uns diente dabei eine aufgeblasene Wasserbombe als Ball.



Weitere, anspruchsvollere Programmideen, bei denen die Kinder nach und nach mit den vorhandenen Sensoren und ihrem Gebrauch vertraut gemacht werden, sind:

- der ängstliche Finch

  Der Finch soll anhalten, sobald er in einen Tunnel kommt.

  Der Finch soll Geräusche machen, wenn er durch den Tunnel fährt („Pfeifen im Walde“).

- der vorsichtige Finch

  Der Finch hält an, sobald er vor einem Hindernis steht.

Beispiele aus der Sek1:

In der Sek1 stellten sich die Schüler nach einer Einführung in die Möglichkeiten des Finchs eigene Aufgaben. Dabei entstanden u. a.:

- ein Programm, mit dem der Finch eigenständig ein Labyrinth durchfahren kann
- ein Programm, bei dem der Fich auf einer schrägen, sich langsam in alle Richtungen bewegenden Ebene seine Lage stets hält und nicht abstürzt
- ein Programm, bei dem der Finch als Steuerung für ein Computerspiel (in diesem Fall passenderweise Flappy Bird) dient. Seine Bewegungen werden auf den „Flappy Bird“ übertragen.

Der Finch besitzt sehr viel Potential und kann auch in höheren Jahrgängen eingesetzt werden. Dies ist sein großer Vorteil gegenüber anderen für die Grundschule angebotenen, einfachen Robotern wie dem BeeBot. Die Beschäftigung mit dem Finch ist damit ebenfalls anschlussfähig, sowohl was die Robotik angeht als auch in Hinsicht auf spätere Pogrammiereinheiten mit Scratch und seinen Derivaten.