SCHULE AKTIV!
Sonderheft des BMB

CODING

ALS BAUSTEIN DER DIGITALEN GRUNDBILDUNG

Foto: DaVinciLab.at

Mit Beiträgen von: Anton Reiter • Gerhard Brandhofer • Peter Micheuz • Manuela Appl • Gudrun Heinzelreiter-Wallner • Alfons Koller • Emmerich Boxhofer • Stefan Hametner • Silvia Nowy-Rummel • Hubert Egger • Hubert Pöchtrager • Walter Fikisz • Josef Buchner • Claus Zöchling • Harald Meyer • Maria Grandl • Bettina Höllerbauer • Martin Ebner • Wolfgang Slany • Martina Friedl • Elisabeth Weißenböck • Anna Relle Stieger • Martin Weissenböck • Wilfried Baumann • Johann Stockinger • Christine Wahlmüller-Schiller

Vorwort des Herausgebers

Zu dem im Oktober 2016 vom Bildungsministerium (BMB) gemeinsam mit dem CDA-Verlag gestalteten und an fast alle österreichischen Schulen verteilten Sonderheft der Zeitschrift „Schule aktiv" unter dem Titel „CODING – ein Baustein der informatischen Bildung" (downloadbar unter: http://pubshop.bmbf.gv.at/detail.aspx?id=632) ist angesichts internationaler Trends und curricularer Vorhaben in Österreich, die digitale Bildung, insbesondere logisches Denken und ansatzweise ablauforientiertes (algorithmisches) Problemlösen mit geeigneten (nicht nur informationstechnischen) Hilfsmitteln auch schon im Unterricht in der Grundschule etwa ab der 3. Schulstufe zu vermitteln und zu fördern, nun ein Jahr später eine Nachfolgeedition erschienen, deren Zielgruppe in erster Linie die Pädagoginnen und Pädagogen an den österreichischen Pflichtschulen sind. Im Rahmen der BMB-Initiative „Schule 4.0" ist ab dem Schuljahr 2018/19 die Einführung einer verpflichtenden Übung „Digitale Grundbildung" an der Sekundarstufe I geplant, die schulautonom oder integrativ in anderen Unterrichtsgegenständen erfolgen wird – der Pilotbetrieb wird bereits in diesem Schuljahr auf Basis eines derzeit noch in Verordnung befindlichen Lehrplans aufgenommen. Ein eigenes Pflichtfach „Coding", das in Großbritannien bereits für alle Schülerinnen und Schüler von fünf bis 14 Jahren an den Grund- und später weiterführenden Schulen auf dem Lehrplan steht – worauf Miles Berry im letztjährigen Special Bezug genommen hat und in dem die Grundlagen der Informatik, zu denen Programmieren gehört, erarbeitet werden, ist in Österreich (derzeit noch) nicht vorgesehen.

Ein Trend nicht nur hierzulande ist die Programmierung von Kleinsystemen wie Arduino oder Raspberry Pi, die wie die Nutzung von Coding Apps für Smartphones und Tablets auch außerhalb des Schulbereichs bei Kindern und Jugendlichen auf großes Interesse stößt. Die am MIT entwickelte visuelle Programmiersprache Scratch hat bereits Kultstatus im Web. „Coding macht Schule", auch in diesem Jahr und weiterhin. Die Europe Code Week 2017 (7.-22. Oktober) steht unter dem Motto: "Learning to code helps us to make sense of how things work, explore ideas and make things, for both work and play. What's more it helps us to unleash our creativity and work collaboratively with wonderful people both near us and all over the world."(http://codeweek.eu/)

Programmierfähigkeiten gehen über „Codeschreiben" hinaus, gefördert werden so analytisches Denken und Problemlösefähigkeit, die in der heutigen Berufswelt unentbehrlich sind. Wer die Funktionsweise von Computern im weitesten Sinne und in der Vielfalt heutiger Applikationen versteht und sie über Schnittstellen und Codes bedarfsorientiert programmieren kann, der bewegt sich nicht mehr auf der Ebene des unwissenden Nutzers und Nur-Konsumenten neuer Medienbehelfe und elektronischer Lifestyle-Gadgets. Er erlebt sich als Verursacher, als „Maker" einer computergesteuerten Problemlösung und erhöht mit dem Aha-Erlebnis auch seine digitale Kompetenz. Zahlreiche Coding-Tutorials im Web bieten dabei außerschulische Einstiege an.

Bei der Neukonzeption des vorliegenden zweiten Themenheftes wurden zwecks Wahrung der Kontinuität auch bisherige Autorinnen und Autoren auf einen weiteren Artikel angesprochen sowie zusätzlich Pädagoginnen und Pädagogen aus dem Bereich der Sekundarstufe I, tlw. aus der Grundschule und der Lehrerausbildung, kontaktiert, die alle bereits Programmiererfahrungen aus ihrem Unterricht haben und diese nun in einem exklusiven, didaktisch ausgerichteten Beitrag bereitstellen. Obwohl die Zielrichtung des Specials unter dem Motto „Coding aus der Praxis für die Praxis" umschrieben werden kann und so bereits in der Pflichtschule „Lust aufs Programmieren" bereitet werden soll, ist der eine oder andere Beitrag auch kritisch gehalten – nicht was die Bedeutung des Programmierens betrifft, sondern bezogen auf den großen Nachholbedarf im österreichischen Schulwesen.

Für Gerhard Brandhofer als Vertreter der Hochschule mit zusätzlicher Pflichtschullehrerausbildung ist „Programmieren [...] nötig, um in die fundamentalen Meta-Ideen der Informatik vorzudringen: Algorithmisierung, strukturierte Zerlegung, Sprache." – ICT Literacy, nur partiell mit der tiefer gehenden informatischen Bildung kongruent, zählt längst zu den 21st Century Skills.

Auch für Peter Micheuz, der abgesehen von der Grundschule als Informatiklehrer in allen Schulstufen bis hinauf in den tertiären Bereich tätig war und ist, sind „Programmieren (Denken) und Coding (Schreiben) [...] unabdingbare Teile heutiger Allgemeinbildung"– ob aber diese Kompetenzen (neben sieben anderen im gegenwärtigen Pilot-Lehrplan) in einer verbindlichen Übung „Digitale Grundbildung" mit einer Vorgabe von zwei bis maximal vier Stunden effizient vermittelt werden können, bleibe seiner Meinung nach abzuwarten. Digitale Kompetenzen können nicht in einer „Hour of Code" erworben werden.

Als betroffene Pädagogin der Sekundarstufe I und zudem in der Lehrerausbildung tätig, geht Manuela Appl in Grundzügen auf die Bildungsinitiative „Schule 4.0" des BMB ein und spricht die neuen Herausforderungen an, die sich daraus (abgesehen von den Schülern) auch für die Lehrenden ergeben.

Gudrun Heinzelreiter-Wallner, Emmerich Boxhofer, Stefan Hametner und Alfons Koller verweisen in ihrem Gemeinschaftsbeitrag auf IMST-Projekte des Instituts für

Unterrichts- und Schulentwicklung an der Uni Klagenfurt zum Bereich Computational Thinking, Coding und Robotik, die u.a. mit Lego® Education Europe durchgeführt und evaluiert wurden bzw. werden.

Silvia Nowy-Rummel berichtet als Volksschullehrerin über ihre Erfahrungen mit den Kindern ihrer Klasse bei der Programmierung eines einfachen Computerspiels mit Scratch und verweist auf die zahlreichen positiven Auswirkungen in kognitiver und sozialer Hinsicht bei den Schülerinnen und Schülern.

Hubert Egger, als Gymnasiallehrer und Lehrbeauftragter auf unterschiedlichen Niveaus im Fachbereich Informatik tätig, hebt die Vielzahl heutiger Webtools hervor, mit denen man u.a. auch programmieren (lernen) kann. Er selbst hat eine Fülle an didaktischen Beispielen unter Nutzung von Scratch und dem Web-Tool MakeCode für die micro:bit-Programmierung kreiert, die tlw. im Web abrufbar sind (http://gmk.eLearningCluster.at).

Hubert Pöchtrager, u.a. in der Sekundarstufe I als Informatiklehrer tätig, bestätigt mit dem Titel seines Beitrags „Programmieren – ein Weg zu Problemlosekompetenz und informatischer Bildung“ die Grundaussage des vorliegenden Specials, wobei Informatik-Know-how u.a. aus dem (didaktischen) Einsatz von Gratisspiele-Tools im Web wie z.B. das Puzzle Game Lightbot, Angry Bird oder Silent Teacher bezogen werden kann.

Walter Fikisz und Josef Buchner haben Beebots-Bodenroboter nach einem didaktischen Konzept im Fach „informatische Bildung“ des Bachelorstudiums Primarstufe an der PH NÖ eingesetzt, um logisches Denken und kreative Problemlösung in bestimmten Aufgabenfeldern zu trainieren. Die daran angeschlossene Fragebogenerhebung bestätigt den didaktischen Erfolg.

Claus Zöchling ist PTS-Lehrer, Mechatroniker und Entwickler der Raspbotics Coding-Lernboards, die zahlreiche Anwendungsbereiche eröffnen. Kinder und Jugendliche können auf diese Weise spielerisch und intuitiv eine Programmiersprache erlernen (https://www.raspbotics.at/).

Harald Meyer erläutert die digitale Grundbildung im Kompetenzbereich Messern-Steuern-Regeln und Robotik, wobei für die didaktische Aufbereitung und zur Veranschaulichung Experimentierboxen mit elektronischen Bauteilen verwendet werden. Zudem kommen verschiedene Boards zum Einsatz wie z.B. das Arduino-Uno-Board.

Maria Grandl, Bettina Höllerbauer, Martin Ebner und Wolfgang Slany stellen die an der TU Graz entwickelte App Pocket Code, mit der Programme am Smartphone oder Tablet erstellt werden können, vor. Zudem wurde dazu ein MOOC-Kurs gestaltet (http://imoox.at/wbtmaster/startseite/) und als weitere App steht Pocket Paint zur Verfügung.

Martina Friedl berichtet, dass das „Mobile Klassenzimmer“ von Samsung auch im kommenden Schuljahr wieder quer durch Österreich in einer Roadshow unterwegs sein wird. In Coding-Workshops lernen Kinder und Jugendliche unter Einsatz von Pocket-Code das Programmieren von Apps.

Elisabeth Weißenböck und Anna Relle-Stieger blicken auf das erste Jahr ihrer acodemy, eine Programmierschule für Kinder in Wien 3, zurück: „1580 begeisterte Kinder, 1074 absolvierte Programmierstunden und 810 kleine Coding-Projekte“ – das ist ihre Success-Story. Logisches und strukturiertes Denken wird beim Einsatz der Programmiersprachen ScratchJr. und Scratch geschult.

Martin Weissenböck, vormals HTL Direktor und in der BHS beheimatet, erläutert anhand von codierten Beispielen, warum für ihn die bereits 1991 von Guido van Rossum entwickelte Programmiersprache Python, die mehrere Programmierparadigmen unterstützt, den Vorzug bekommt.

Wilfried Baumann, der Coding Workshops in der Österreichischen Computer Gesellschaft (OCG) durchführt, zeigt anhand von drei anschaulichen Beispielen mit angeführten Skripts, wie sich Python mit dem Minecraft Codebuilder verbinden lässt.

Johann Stockinger, Coding-Experte in der OCG, bietet eine Auswahl von empfehlenswerten Ressourcen aus der Plattform Coding4you.at zum Einstieg und zur Vertiefung.

Christine Wahlmüller-Schiller, in der OCG für den Bereich Marketing & Kommunikation zuständig, stellt zwei neue ECDL Coding Module, nämlich den ECDL Junior Coder und ECDL Computing, die im Herbst 2017 angeboten werden, vor.

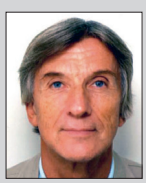
Das vorliegende, primär für die Sekundarstufe I gedachte Sonderheft, wird wie letztes Jahr der regulären Ausgabe von „Schule aktiv“ beigepackt und über den Verteiler des CDA-Verlages an die österreichischen Pflichtschulen geschickt werden. Zudem wird eine pdf-Version auf der BMB-Publikationen-Seite frei zum Download verfügbar sein.

Als redaktioneller Betreuer und Herausgeber danke ich allen Autorinnen und Autoren für Ihre wertvollen Beiträge sowie dem CDA-Verlag für die gute Zusammenarbeit.

Viel Spaß bei der Lektüre wünscht Anton Reiter.

Autor

MinR Dr. Anton Reiter
Bundesministerium für Bildung,
Abteilung II/8 – IT Didaktik und
Digitale Medien
E-Mail: anton.reiter@bmb.gv.at



Programmieren in der Schule im Zeitalter der Digitalität

Mache ich das Richtige wichtig und das Wichtige richtig? Angesichts der zwar unglaublichen und dennoch begrenzten Speicherkapazität unseres Zentralnervensystems und der überschaubaren Zeitkontingente für die Bildungsinstitutionen, stellt sich diese Frage für uns Lehrende immer wieder aufs Neue. Was soll Teil des Unterrichts sein und wie ist mein Unterricht gewinnbringend? Soll Programmieren und Robotik sowie Computational Thinking Platz finden in der Primar- und Sekundarstufe I, wo doch schon die grundlegenden Fertigkeiten wie Schreiben, Lesen und Rechnen von vielen Schülerinnen und Schülern nur teilweise beherrscht werden?

Mache ich das Richtige wichtig?

Welche Kompetenzen brauchen unsere Schüler/innen um sich in einer zunehmend komplexeren, digitalisierten Welt und in Anbetracht absehbarer globaler ökologischer wie sozialer Umwälzungen zurechtzufinden? So wichtig die Kompetenzen Schreiben, Lesen und Rechnen auch sind, sie werden nicht reichen. Neben anderen Institutionen hat auch das World Economic Forum eine Übersicht zu den 21st-Century-Skills erstellt. Bei den grundlegenden Fertigkeiten wird als eine von sechs ICT-Literacy angeführt. Was aber noch wichtiger ist: kreative Informatik, Coding und Robotik beeinflussen die vom WEF genannten Kompetenzen für komplexe Herausforderungen: kritisches Denken, Problemlösen, Kreativität, Kommunikation und Kollaboration. Informatische Bildung als Wert und Mehrwert.

Abb. 1: Zu den 21st Century-Skills zählt auch ICT-Literacy

Wenn man sich für die Implementierung des Themas Coding in den Unterricht einsetzt, ist man dem Vorwurf ausgesetzt, dass Inhalte für eine kleine Minderheit in der künftigen Arbeitswelt propagiert werden. Dabei hat Informationstechnologie für die Wirtschaft enorme Bedeutung und bringt der Jugend gleichzeitig hervorragende Berufsaussichten. Programmieren ist Teil informatischer Bildung und diese ist unmittelbare Voraussetzung für eine Vielzahl von Berufen. Hinzu kommt, dass der Wandel in der Berufswelt durch die Leitmedientransformation uns vor neue Herausforderungen stellt, Faktenwissen ist heute sehr vergänglich, andere Kompetenzen werden von den Schulabgängern und Schulabgängerinnen verlangt. Dieser Tatsache wird das Bildungswesen noch nicht gerecht. Die bei der Auseinandersetzung mit Informatik erworbenen Kompetenzen beschränken sich nicht nur auf den Fachbereich, bei der Beschäftigung mit den 0en und 1en kann man auch viel über Physik, Mathematik, Deutsch, Philosophie, über Kooperation, Kommunikation lernen, Problemlösestrategien trainieren und Kreativität ausleben.

Neben diesem Bezug zur Arbeitswelt gibt es weitere Argumente, die für den Einsatz von digitalen Medien im Unterricht im Allgemeinen und Programmieren als Unterrichtsinhalt im Besonderen sprechen (Methodenvielfalt, Lebenswelt, Wechselwirkung, Reflexion). Unsere Lebenswelt verändert sich grundlegend. Das Digitale ist mitten unter uns, angesichts des Zusammenpralls mit einer Schuldidaktik, die auf dem Leitmedium Buch gegründet ist, sind wir planlos, ziellos und nicht in der Lage, die Folgen abzuschätzen.

Sollen jetzt etwa alle Schüler/innen künftig Programmierer werden? Genauso wenig, wie alle Schriftsteller werden sollen, die das Schreiben erlernen. Aber: wer weiß, wie man programmiert, hat einen tieferen Einblick in die Mechanismen der digitalen Welt. Um die Abstraktion weiter zu treiben: Programmieren ist nötig um in die fundamentalen Metaideen der Informatik vorzudringen: Algorithmisierung, strukturierte Zerlegung, Sprache; Informatik wiederum kann wichtige Beiträge zu den genannten 21st Century Skills liefern.

Mache ich das Wichtige richtig?

Programmieren ist nur ein Teilbereich und nicht gleichzusetzen mit informatischer Bildung an sich. Informatikunterricht sollte daher nicht ausschließlich ein Programmierkurs sein, Themen wie Datenbanken, Security, Kryptologie, Modellbildung sind ebenso notwendiger Bestandteil informatischer Bildung. Wichtig ist, dass Programmieren im Unterricht so umgesetzt wird, dass es die Kinder kreativ tätig werden lässt, algorithmisches Denken gefördert wird und man nicht bei

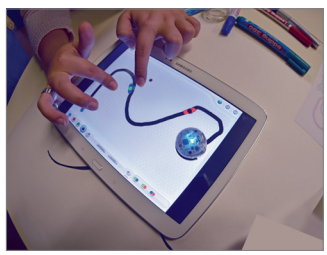


Abb.2: Mit dem Ozobot-Roboter kann Computational Thinking/Programmieren auch ohne Computer veranschaulicht werden.

Rekonstruktion und Dekonstruktion vorhandener Inhalte verharrt.

Wenn Informatik unterrichtet wird, dann beschränkt sich das des Öfteren auf Applikationsschulung, auf die Gewöhnung an die Maschine. Medienpädagogische Themen werden gestreift oder ausgelagert und Programmieren sowie Computational Thinking waren bislang kaum Thema im Unterricht der Primarstufe und wenig in der Sekundarstufe I. Mittlerweile gibt es eine Vielzahl an erziehungsorientierten Programmiersprachen und Robotern, die algorithmische Fähigkeiten fördern können, kindgerecht sind, mit denen man Elemente des Game Based Learning aufgreifen kann und die den Kindern Spaß machen. Trotz allem sollte man nicht in digitaler Euphorie über das Ziel schießen. Eine breite Bildung ist das Wichtigste, was man unseren Kindern mitgeben kann - und Programmieren ist ein Teil davon, für Buben wie für Mädchen.

Autor

HS-Prof. Mag. Dr. Gerhard Brandhofer, BEd. hat eine Hochschulprofessur für Mediendidaktik und informatische Bildung an der Pädagogischen Hochschule Niederösterreich inne. Zu den Arbeitsschwerpunkten zählen der Einsatz visueller Programmiersprachen im Unterricht, digitale Kompetenzmodelle für Schüler/innen und Lehrende. Forschungsaktivitäten und Veröffentlichungen umfassen die Themenfelder der Nutzung digitaler Medien in der Schule wie auch in der Hochschule, die Bedingungen für gelingende informatische Bildung.

Literatur:

Brandhofer, G. (2017). Mehrwert oder ein Wert an sich? Das Digitale und die Schule. Schule neu denken und medial gestalten; KidZ Sammelband, in press.

Schwill, A. (1993). Fundamentale Ideen der Informatik. Zentralblatt für Didaktik der Mathematik, 25 (1), 20-31.

World Economic Forum. (2016). New Vision for Education: Fostering Social and Emotional Learning through Technology. Genf: World Economic Forum.

Vom Coding zur Digitalen Grundbildung

Vage Begriffe und Terminologien

„Coding steht hier für einen moderneren, umfassenderen Begriff für das Programmieren. Weltweit wird bereits von einer neuen Kulturtechnik gesprochen. Grundlegende Kenntnisse darüber können in Zukunft für alle in verschiedenen Lebenslagen von Bedeutung sein." So steht es auf dem ambitionierten Portal http://www.coding4you.at, das die österreichische Coding-Initiativen vernetzt und die Vielfalt im Umfeld von Coding veranschaulicht. Allerdings sollte das so nicht stehen gelassen werden.

Wer sich mit dieser nur scheinbar akademischen Frage und den Antworten intensiver beschäftigen möchte, ob es einen Unterschied zwischen „Coding" (Codieren) und „Programming" (Programmieren) gibt, sollte sich im empfehlenswerten Frage-Antwort-Forum http://www.quora.com umsehen. Hier wird Coding von vielen Experten definitiv als Teil von Programmierung ausgewiesen. Ich sehe das auch so.

Ich bin auch skeptisch, Coding – neben Lesen, Schreiben und Rechnen – gleich als neue Kulturtechnik zu bezeichnen, ist doch Codieren im engeren Sinne nichts anderes als Schreiben, und zwar in einer formalen (Programmier)Sprache. Gleiches gilt m. E. auch für den Begriff „Computational Thinking", der im neuen Lehrplan(entwurf) für Digitale Grundbildung aufscheint. Neben sieben anderen Themenbereichen bietet er als achtes - last but not at all least - Lehrplanthema jenen Rahmen, in dem Coding im Nahbereich von Algorithmen und Programmierung implizit aufscheint.

Eine letzte Anmerkung zu coding4you.at: „Grundlegende Kenntnisse darüber (über Coding) können in Zukunft für alle in verschiedenen Lebenslagen von Bedeutung sein" ist in vorausschauender Weise im Konjunktiv formuliert. Allen gutgemeinten Behauptungen aus den Echokammern diverser Digitalapologeten (zu denen ich mich m. E. auch zähle) muss (leider?) entgegengehalten werden, dass es noch keine schlüssigen Evidenzen hinsichtlich der Übertragbarkeit von Coding-Kompetenzen auf generelle kognitive Fähigkeiten gibt. Peter Denning formuliert das in pointierter Form so [1]: „In 1997, Koschmann weighed in with more of the same doubts and debunked a new claim that learning programming is good for children just as learning Latin once was. (There was never any evidence that learning Latin helped children improve life skills.)". In diesem

Zusammenhang ist die Frage erlaubt, ob Javascript - als Metapher für eine aktuelle Programmiersprache - wirklich das Latein des 21. Jahrhunderts ist [2], und welche Programmiersprache(n) in der Sekundarstufe I wirklich gekannt und beherrscht werden solle(n). Der neue Lehrplan für die Digitale Grundbildung schreibt es jedenfalls vor.

Eine schlüssige Beziehung von Coding, in welcher formalen Sprache auch immer, und Computational Thinking kann im lesenswerten Buch „Computational Thinking {and coding} for Every Student" von Kiki Prottsman, einer Mitarbeiterin von code.org und Protagonistin für mehr Informatik in US-Schulen nachgelesen werden [3]. Es lohnt sich auch ein Blick auf das Portal der ISTE, einer globalen Organisation für Technologie in der Bildung [4], in dem beispielsweise im Beitrag „Turn coders into computational thinkers" auf die Gefahr der Engführung von Coding hingewiesen wird: „Computer science is more than just coding. Thinking like a computer scientist involves more skills than just being able to write code." Damit ist der Bogen zum Lehrplanthema, Buzzword (und Hype?) Computational Thinking und auch seine explizite Stellung im Lehrplan des neuen Fach(gebiet)es „Digitale Grundbildung" für die Sekundarstufe I gespannt.

Mit dem aus dem angelsächsischen Raum importierten Begriff Computational Thinking gibt es nach Coding also einen weiteren Anglizismus, der erst einmal verdaut werden muss: Von Bildungsexperten wie z.B. dem Lehrplanteam, aber vor allem den vielen unsicheren Lehrkräften - wie mich - in ihrem Bemühen, den Lehrplan zu erfüllen und den Schülern das Programmieren im Allgemeinen und das Codieren im Speziellen als unabdingbaren Teil von Allgemeinbildung „beizubringen". Das wird eine Herkulesaufgabe, sowohl in der täglichen Unterrichtspraxis als auch in der Lehreraus- und -fortbildung.

Lehrpläne und das Fach(gebiet) Digitale Grundbildung

Lehrplänen werden im Gegensatz zu unverbindlichen Referenzrahmen und Kompetenzmodellen wie dem seit ein paar Jahren existierenden Digikomp8-Modell, das auch als Grundlage für den Lehrplan Digitale Grundbildung herangezogen wurde, unter anderem folgende Funktionen zugeschrieben [5]:

- Politische Willensäußerungen über verbindliche Bildungsziele
- Steuerung des Unterrichts
- Vorgaben für Lehrbücher
- Gewährleistung einheitlicher Lehr- und Lernbedingungen für gleichartige Schulen
- Kriterien für die Kontrolle und Beurteilung des Unterrichts durch Organe der Schulaufsicht

Es wird interessant sein, wie der vier Jahre umspannende Lehrplanentwurf zur Pilotierung der DGB von der Basis aufgenommen, verdaut und letztlich an den Schulstandorten umgesetzt werden wird. Spätestens in dieser Phase werden die Besonderheiten des neuen Lehrplan(entwurf)es und der Rahmenbedingungen offensichtlich.

- Die Einführung und das Inkrafttreten eines neuen Lehrplanes für ein neues Fach(gebiet) in Verbindung mit einer verpflichtenden Fachbezeichnung ist in Anbetracht des übervollen Fächerkanons immer ein politischer Kraftakt. In gegenständlichen Fall hat er bereits lange auf sich warten lassen. Das ist zumindest aus Sicht der Vorreiter digitaler Bildung an Schulen (wer kann heute noch dagegen sein?) sehr zu begrüßen. Allerdings ist die Konstruktion einer „Verbindlichen Übung" ebenso halbherzig wie ihre nicht verbindliche Abhaltung für den Fall, dass es Schulen gelingt, alle Lehrplaninhalte in andere traditionelle Fächer wegzuintegrieren. (Einschub: Sollte in Österreich ein Pflichtfach DGB und keine „Übung" - wie vereinzelt in anderen Ländern – nicht bereits ein integraler Teil eines modernen Fächerkanons sein?). Auch mutet die Vorgabe von zwei bis maximal vier Stunden für die „Übung" etwas seltsam an, zumal es bereits seit fast 30 Jahren Schulen gibt, die Informatik jetzt schon in einem Ausmaß von über vier Stunden als Pflichtfach anbieten. Mehr war für die Pilotierungsphase anscheinend politisch nicht durchzusetzen. Nach erfolgreicher Pilotierung darf es ja dann mehr sein.
- Die Steuerung des DGB-Unterrichts wird den Pilotschulen viel Koordinationsarbeit abverlangen. Das trifft auf die Abstimmung der einzelnen Lerninhalte in allfälligen eigenen Zeitgefäßen mit der Fachbezeichnung Digitale Grundbildung ebenso zu (was wann und wie unterrichten?) wie auf die im Lehrkörper zu orchestrierende Aufteilung der Inhalte auf andere Fächer und damit auch fachfremde Lehrkräfte.
- Es ist nicht zu erwarten, dass es in Kürze ein österreichisches Standardlehrwerk für das Fach Digitale Grundbildung geben wird, oder doch? Da es um einen Lehrplan eines neuen Fach(gebiet)es geht, gibt es noch keine „geheimen Lehrpläne", sprich approbierte Lehrbücher, sondern nur den digitalen Ozean nicht approbierter offener Lernressourcen und Unterrichtsmaterialien. Die Schulen und die involvierte Kollegenschaft sind angehalten und der Qual der Wahl ausgesetzt, sich der Fülle an bestehendem, zum Teil sehr guten, digital verfügbarem Unterrichtsmaterial selektiv zu bedienen und eine „Auto-Approbation" vorzunehmen. Vielleicht wird hier die angekündigte Eduthek Abhilfe schaffen?
- Der Gewährleistung einheitlicher Lehr- und Lernbedingungen für alle Schüler der Sekundarstufe I ist wohl jene Lehrplanfunktion, die im gegenständlichen

Fall realiter nicht zu erfüllen sein wird, da es (leider?) keinen jahrgangsmäßig gestuften, sondern nur einen All-in-One Lehrplan gibt. Verzerrungen hinsichtlich der Lehrstoffverteilung sind aufgrund unterschiedlicher Rahmenbedingungen an den einzelnen Schulen vor"programmiert". Das muss nicht notwendigerweise schlecht sein. Viele Wege führen nach Rom.

- Die Rolle der Schulaufsicht, die in den einzelnen Bundesländern im Bereich des Fachinspektorats sehr unterschiedlich organisiert ist, muss in Bezug auf Assessment-Aspekte (wieder so ein Anglizismus!) erst definiert werden. In den ersten Jahren wird ihr, wie den eEducation-Beauftragten, wohl eine koordinierend-beratende Funktion zukommen.

Algorithmen, Programmierung und Coding im Lehrplan Digitale Grundbildung

Im österreichischen (Pilot)Lehrplan für Digitale Grundbildung entfaltet sich das Thema Computational Thinking entlang der Inhaltsstränge Algorithmen, Programmierung und „kreative Nutzung" von Programmiersprachen. Der geneigte Leser ist jetzt schon eingeladen, die 13 explizit ausgewiesenen Kompetenzen zu Computational Thinking zu interpretieren und sich die unterrichtliche Praxis dazu zumindest einmal in Ansätzen vorzustellen. Zum Beispiel, welche unterschiedlichen Programmiersprachen es sein sollen, was einfache Programme sind und wie lange es braucht, bis grundlegende Programmierstrukturen (besser: Programmstrukturen) beherrscht werden, und letztlich in welchem Ausmaß Coding eine Rolle spielen soll. Das wird mit dem Lehrplan, der sehr breit angelegt ist, nicht beantwortet. Apropos kreative Nutzung von Programmiersprachen: HTML ist keine Programmiersprache, und deren Verwendung nicht notwendigerweise kreativ. Über den Umfang der ambitionierten Lernziele zum kreativen Einsatz von FabLab-Projekten, Educational Robotics bzw. 3D-Druck gibt es keine näheren Angaben. Aber Lehrplanentwürfe sind ja nicht in Stein gemeißelt.

Welcher Stellenwert dem Computational Thinking im Reigen der anderen sieben Lehrplanbereiche: Gesellschaftliche Aspekte, Medienwandel und Digitalisierung, Informations-, Daten- und Medienkompetenz, Betriebssysteme und Standard-Anwendungen, Mediengestaltung, Digitale Kommunikation und Social Media, Sicherheit und Technische Problemlösung in der Umsetzung zukommen wird, kann jetzt noch nicht eingeschätzt werden. Mit Sicherheit stellt dieser Lehrplanbereich an den Großteil der Kollegenschaft hohe Anforderungen. Speziell in Bezug auf Programmierung inklusive Coding, Webdesign und FabLab-Projekte besteht ein großer Aus- und Fortbildungsbedarf.

Programmieren (Denken) und Coding (Schreiben!) sind unabdingbarer Teil von Allgemeinbildung. Gut, dass dies zunächst im Lehrplan steht. Falls dies - auch mit spielerischen Zugängen - ernsthaft, nachhaltig und ergebnisorientiert unterrichtet werden soll, so ist das nur in einem speziellen Unterrichtsfach mit viel verbindlicher Übung (sic!) realistisch. Man sollte sich nämlich nicht der Illusion hingeben, diese Kompetenzen könnten in einer „Hour of Code" erworben werden.

Autor

Peter Micheuz, Prof. Mag.
ist seit 1981 EDV-, ab 1985 Informatiklehrer und IT-Kustos (ab 2016 IT-Manager) am Alpen-Adria-Gymnasium Völkermarkt und seit 2000 Lehrbeauftragter für Informatikdidaktik an der Alpen-Adria-Universität Klagenfurt. Er ist mehrfacher Teilnehmer und Organisator von einschlägigen regionalen, nationalen und internationalen Konferenzen sowie Lehrbuchautor, Herausgeber von Sammelbänden und sechs CDA-Sonderausgaben, Mitarbeiter in diversen Arbeitsgruppen des Bildungsministeriums, und Publizist von einschlägigen Fachartikeln. Er war eLSA-Bundeslandkoordinator, ist derzeit langjähriger ARGE-Leiter für Informatik an AHS in Kärnten, Vorstandsmitglied im Verein ECDL an Schulen, und seit 2015 Vice-Chair der IFIP Working Group 3.1 (International Federation for Information Processing, Arbeitsgruppe „Informatics and Digital Technologies in School Education") und damit auch Kenner der internationalen Szene.
E-Mail:
peter.micheuz@aon.at

Quellen:

[1] **Peter Denning,** Remaining Trouble Spots with Computational Thinking, in: Communications of the ACM, Vol. 60, S. 35, 2017.

[2] **Beat Döbeli:** Javascript - Latein des 21. Jhdts.? Vortrag 2013. http://tinyurl.com/lateinjavascript

[3] **Prottsman, Kiki; Krauss, Jane.** Computational Thinking and Coding for Every Student: The Teacher's Getting-Started Guide. SAGE Publications. 2017.

[4] **ISTE,** International Society of Technology in Education, www.iste.org/explore/categorylist?code=Computational+thinking

[5] **Thonhauser J. (1997):** Neuer Lehrplan – neue Hoffnung. In: Erziehung und Unterricht 147, S. 367–377.

Schule 4.0 – Digitale Kompetenz als Unterrichtgegenstand

Digitale Medien und digitale Kompetenz

Die Nutzung digitaler Medien ist inzwischen nicht nur zu einem wesentlichen Bestandteil des Alltags der Schülerinnen und Schüler geworden, sondern fließt zunehmend auch in die Gestaltung des Unterrichts an den verschiedenen Schulstufen ein. Der verantwortungsvolle und sinnstiftende Umgang mit digitalen Inhalten und Methoden setzt dabei freilich nicht nur technische Fähigkeiten voraus, sondern erfordert ebenso eine Sensibilität für die damit einhergehenden Risiken, etwa bei der Recherche im Internet oder bei der Nutzung sozialer Netzwerke. Da die Erfahrung leider zeigt, dass diese Fähigkeiten üblicherweise nicht schon bei der Nutzung erworben werden, ist es deshalb umso wichtiger geworden, den Erwerb digitaler Kompetenz als festen Bestandteil des Unterrichts an den verschiedenen Schulstufen zu verankern.

Verbindliche Übung „Digitale Grundbildung"

Auch das Bundesministerium für Bildung hat diese Notwendigkeit erkannt und beabsichtigt im Rahmen der Initiative „Schule 4.0" voraussichtlich ab dem Schuljahr 2018/19 die Einführung einer verpflichtenden Übung „Digitale Grundbildung" an der Sekundarstufe 1 im Ausmaß von 2 mal 32 Jahresstunden. Diese Übung kann schulautonom entweder in Form eines eigenen Unterrichtsgegenstandes oder integrativ in anderen Unterrichtsgegenständen abgehalten werden, wobei auch Mischformen zwischen diesen beiden Varianten möglich sein sollen.

Beginnend mit dem Schuljahr 2017/18 ist eine Pilotierung der Initiative auf freiwilliger Basis vorgesehen, zu der sich Schulen bis zum 29. Mai 2017 melden konnten. Das Bundesministerium für Bildung hat in einem eigenen Rundschreiben[1] vorläufig festgelegt, welche Lehrinhalte im Rahmen der verpflichtenden Übung vermittelt werden sollen. Diese lassen sich grob in folgende Themenbereiche untergliedern

- „Gesellschaftliche Aspekte von Medienwandel und Digitalisierung": Schülerinnen und Schüler kennen und verstehen die Wechselwirkung zwischen digitaler und realer Welt, reflektieren über die eigenen Erfahrungen im Umgang mit digitalen Medien, verstehen die Bedeutung von Werten und Normen im Zusammenhang mit digitalen Medien, erkennen Gefahren bei der Nutzung digitaler Medien;
- „Informations-, Daten- und Medienkompetenz": Schülerinnen und Schüler suchen und finden Informationen in digitalen Medien, vergleichen und bewerten die verfügbaren Daten, organisieren und teilen zuverlässige Daten mit anderen Nutzern;
- „Betriebssysteme und Standard-Anwendungen": Schülerinnen und Schüler kennen die gängigen Betriebssysteme und Standardanwendungen;
- „Mediengestaltung": Schülerinnen und Schüler kennen und verstehen die Bedeutung digitaler Medien, erwerben Kompetenzen für die Weitergabe und eigenhändige Entwicklung digitaler Medien;
- „Digitale Kommunikation und Social Media": Schülerinnen und Schüler kennen die digitalen Kommunikationswerkzeuge, gestalten verantwortungsvoll die eigene digitale Identität, wenden Verhaltensregeln bei der digitalen Kommunikation an, kennen die Einsatzgebiete digitaler Kommunikation zur zielgerichteten Zusammenarbeit mit anderen Nutzern;
- „Sicherheit": Schülerinnen und Schüler sind für die Risiken im Umgang mit digitalen Medien sensibilisiert, kennen die notwendigen Vorkehrungen zur Risikoabwehr, schützen die eigene Privatsphäre;
- „Technische Problemlösung": Schülerinnen und Schüler verstehen die Funktionsweise der verwendeten Geräte, erkennen technische Probleme und können geeignete Problemlösungsstrategien entwickeln; und
- „Computational Thinking": Schülerinnen und Schüler können eigene Anwendungen für die verwendeten Geräte erstellen;

Der endgültige Lehrplan für die verpflichtende Übung steht derzeit noch nicht fest und wird voraussichtlich erst für das Schuljahr 2018/19 durch Verordnung des Bundesministeriums für Bildung festgelegt.

Kompetenzerwerb durch Lehrende

Die Notwendigkeit, den Schülerinnen und Schülern digitale Kompetenz zu vermitteln, bringt zugleich aber neue Herausforderungen für die Lehrenden mit sich, die einerseits selbst digital kompetent und andererseits auch in der Lage sein sollten, neben dem im Rahmen der eigenen Ausbildung erworbenen und vertieften Fachwissen auch entsprechende Kenntnisse und Fähigkeiten im Umgang mit digitalen Medien zu unterrichten. Dabei stellen verschiedene Organisationen und das Bundesministerium für Bildung geeignete Informationsangebote zur Verfügung.

Eine wertvolle Hilfestellung bietet zunächst der Kompetenzrahmen digi.komp[2], der vom Bundeszentrum On-

linecampus Virtuelle PH der Pädagogischen Hochschule Burgenland (kurz „Virtuelle PH“) im Auftrag des Bundesministeriums für Bildung entwickelt wurde. Den Bedürfnissen der unterschiedlichen Altersgruppen entsprechend liefert dieser Kompetenzrahmen nicht nur konkrete Kompetenzmodelle für die einzelnen Schulstufen, sondern enthält mit dem Kompetenzmodell digi.kompP auch einen Leitfaden für den Erwerb digitaler Kompetenz durch Pädagoginnen und Pädagogen. Dabei werden verschiedene Kernkompetenzen unterschieden, die vor, während oder in den ersten fünf Praxisjahren nach der Ausbildung vorhanden sein sollten. Anhand des Kompetenzmodells digi.kompP können Lehrende also feststellen, über welche Kenntnisse und Fähigkeiten sie verfügen sollten, und sofern erforderlich den Erwerb fehlender Kompetenzen planen. Zu diesem Zweck bietet die virtuelle PH auch zahlreiche Fortbildungsveranstaltungen an.

Ebenso verfolgt das Bundesministerium für Bildung mit der Initiative „eEducation Austria“[3] das Ziel, die Vermittlung digitaler Kompetenz in den verschiedenen Schulstufen und Unterrichtsgegenständen zu fördern. Zu diesem Zweck können Lehrende über die Onlineplattform der Initiative auf zahlreiche Informationsangebote zugreifen, die neben didaktisch abgestimmten Aufgabenstellungen und Werkzeugen für bestimmte Schulstufen und Unterrichtsgegenstände („eTapas“) auch digitale Publikationen („eBooks“) zu relevanten Themen umfassen.

Schließlich soll auch die Initiative „IMST – Innovationen machen Schule top“[4] des Bundesministeriums für Bildung die Lehrende bei der innovativen Gestaltung des Unterrichts in den MINDT-Fächern (Mathematik, Informatik, Naturwissenschaften, Deutsch und Technik) unterstützen. Im Rahmen dieser Initiative können unter anderem Förderungen für Schul- oder Unterrichtsprojekte beantragt werden, die sich mit dem Thema kompetenzorientiertes Lernen mit digitalen Medien befassen. Neben einer finanziellen Förderung ist dabei insbesondere auch eine inhaltliche Projektbegleitung durch Schulpraktiker und Wissenschaftlerinnen und Wissenschaftler an Pädagogischen Hochschulen oder Universitäten möglich.

Autor

Manuela Appl, BEd.
Pädagogische Hochschule OÖ – Institut Berufspädagogik
Fachbereich Information und Kommunikation
E-Mail: manuela.appl@ph-ooe.at

[1] *Siehe dazu das Rundschreiben des Bundesministeriums für Bildung vom 20.04.2017, GZ BMB-9.000/0029-II/8/2017*
[2] *Abrufbar auf der Website http://www.digikomp.at*
[3] *Abrufbar auf der Website http://eeducation .at*
[4] *Abrufbar auf der Website http://www.imst.ac.at*

Coding und Robotik im Unterricht - Professionell begleitet durch IMST

Digitale Medien haben im Unterricht eine wichtige Rolle eingenommen, deren methodischer und fachdidaktischer Einsatz gut durchdacht und reflektiert sein muss. IMST bietet dafür eine sehr gute Unterstützung.

IMST ist eine bundesweite Initiative des österreichischen Bildungsministeriums, das innovativen Lehrern/-innen und Schulleitungen Unterstützung bei der Entwicklung ihres Unterrichts bzw. ihrer Institution bietet. Sie wird vom Institut für Unterrichts- und Schulentwicklung an der Alpen-Adria-Universität Klagenfurt geleitet. Das Akronym IMST steht für „Innovationen Machen Schulen Top“.

Im IMST-Themenprogramm „Kompetenzorientiertes Lernen mit digitalen Medien“, unterstützt durch das Team der Autoren/-innen an der PH Linz in Kooperation mit der FH Oberösterreich (Campus Hagenberg, Department für Kommunikation und Wissensmedien) sowie der Uni Linz (Institut für Pädagogik und Psychologie) finden vor allem Projekte in MINT-Fächern[1] und Deutsch Unterstützung, sodass Schüler/-innen die Bedeutung der Naturwissenschaften und der Mathematik in unserem Alltag erkennen und die Auseinandersetzung mit diesen Themengebieten sowie mit Medien als Werkzeuge im beruflichen und privaten Alltag intensiviert wird. Neben dem Fokus auf Coding und Robotik als Teilbereiche einer technischen und informatischen Grundbildung stehen auch medienunterstützte Unterrichtsformen im Mittelpunkt der Auseinandersetzung, wie sie heute bei der Verwendung von Plattformen und Netzwerken allgegenwärtig sind. E-Learning und E-Teaching sowie der Umgang mit den jeweils neuen Medien eröffnen dabei auch Möglichkeiten zur Individualisierung und Differenzierung, zur Visualisie-

[1] *MINT-Fächer: Mathematik, Informatik, Naturwissenschaften und Technik (u. a. DG/GZ, Werken technisch) sowie die verwandten Fächer der berufsbildenden Schulen*

Abb. 1 Vier Aspekte des Lernens mit „neuen" Medien
Foto: 1, 4: Rachbauer Tamara, 2: Brein Christoph, 3: Binder Christine

rung und zur Interaktion durch die Schüler/-innen selbst. So leisten diese Projekte Entwicklungsschritte zu einem kompetenzorientierten Unterricht und für Schulprofile mit E-Learning- oder Medien-Schwerpunkten, wie es auch Ziele der eEducation-Initiative des Bildungsministeriums sind.

Im IMST-Themenprogramm „Kompetenzorientiertes Lernen mit digitalen Medien" reichen engagierte Lehrerinnen und Lehrer aller Schultypen, von der Primarstufe über die Sekundarstufe I bis zur Sekundarstufe II und den Berufsschulen, und aller Schulstufen Projekte ein. Diese Projekte verfolgen vielfältige fachdidaktische Ansätze. Im Zuge der von IMST angebotenen Workshops, dem jährlich stattfindenden IMST-Tag sowie der IMST-Tagung finden auch österreichweite Vernetzungen zwischen den Teilnehmenden statt. Durch diesen Informationsaustausch werden viele neue Ideen und weiterführende Impulse entwickelt, die ein alters- sowie schul- und schultypenübergreifendes Denken und Arbeiten ermöglichen.

IMST-Projektbetreuung

Den Lehrenden steht durch die Teilnahme am Projekt IMST das gesamte Projekt- bzw. Schuljahr ein Betreuungsteam zur Verfügung, das sich aus Vertreter/-innen aus der Wissenschaft (Universitäten, Pädagogischen Hochschulen) und Vertreter/-innen aus der Schulpraxis zusammensetzt. Dieses IMST-Betreuungsteam bietet unabhängig vom Schultyp und vom Unterrichtsfach fachdidaktische, medienpädagogische und technische Beratung und Unterstützung für Lehrer/-innen und Direktor/-innen.

Die IMST-Initiative kann auf eine umfassende praktische Expertise im Bildungssektor über Jahre hinweg zurückgreifen. Reflexive Dokumentationen, Good-Practise-Beispiele, Empfehlungen und fachdidaktisch sowie medienpädagogisch reflektierte Unterrichtsmaterialien sind von allen geförderten Projekten im IMST-WIKI unter http://www.imst.ac.at abrufbar.

Computational Thinking, Coding und Robotik praktisch umgesetzt

Computational Thinking ist im Schulwesen schon lange nicht mehr nur auf HTLs und technische Fächer beschränkt und wird mit unterschiedlichen Unterrichtsbehelfen, die von Hardware-Komponenten bis hin zu Software-Werkzeugen reichen, an Schüler/-innen in allen Altersstufen herangetragen. Es leistet durch die Förderung von Denk- und Strukturierungsprozessen auch einen Beitrag zur fachlichen Allgemeinbildung.

IMST-Projekte zu Coding und Robotik betreffen Schulen der Sekundarstufe II, vorwiegend HTLs sowie technische Berufsschulen, aber auch die Schulen der Primar- und Sekundarstufe I. Sie erfolgen auch schulstufen- und schultypenübergreifend. Beispielhaft sei hier das Projekt „Sensitives Kuscheltier goes Youngsters" herausgehoben.

Die Ausgangssituation des IMST-Projekts „Sensitives Kuscheltier goes Youngsters" war das erste IMST-Projekt von Christoph Brein, in dem HTL-Schüler/-innen Raspberry-Pi-Roboter programmierten, mittels Bekleidung

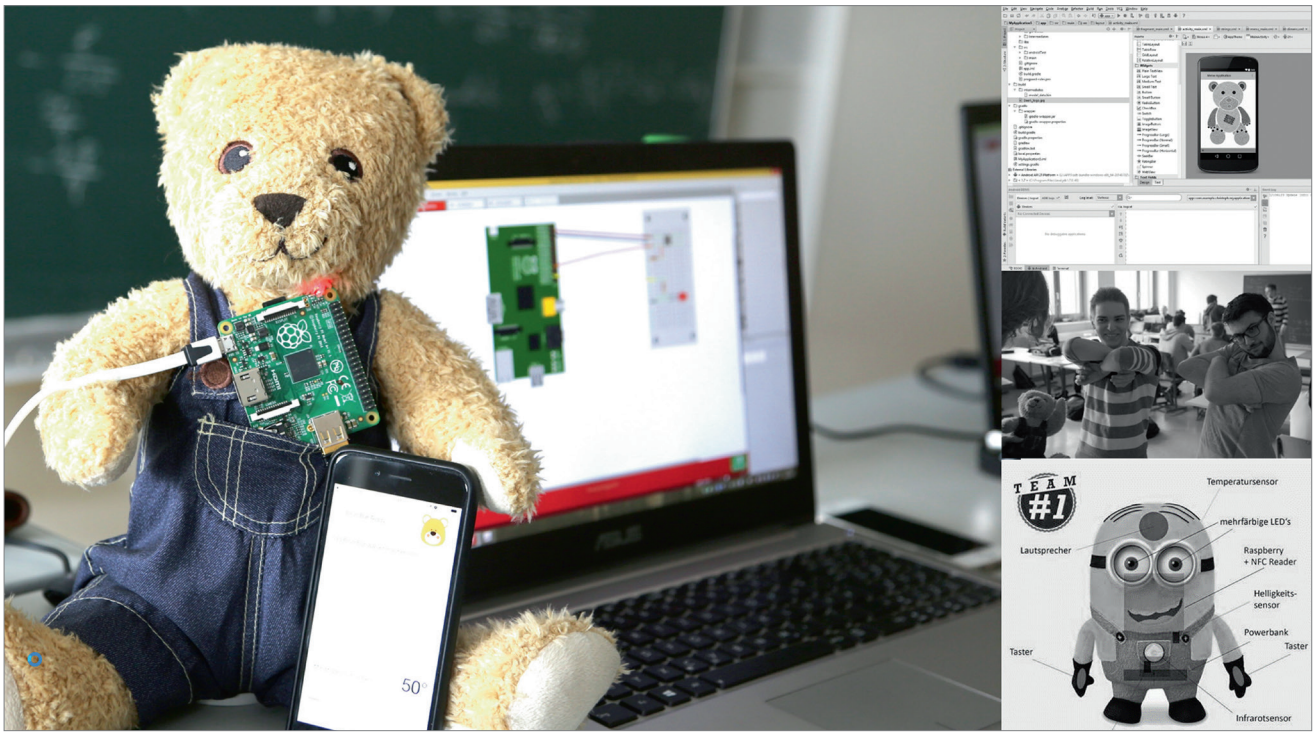


Abb. 2 „Sensitives Kuscheltier" – Raspberry Pi mit Sensoren
Fotos: Brein Christoph – IMST-Projekt ID 1569, ID 1706

personifizierten und mit unterschiedlichen Sensoren, die auf diverse Interaktionen mittels Sprachausgabe reagierten, ausstatteten. Damit wurde eine Initiative in einer technischen Schule gesetzt, welche innovativ die Lernziele von mehreren Fächern umsetzte und den Schüler/-innen viel Spaß und Freude bereitete. Aufbauend auf dieses Projekt nahmen die Schüler/-innen die Herausforderung in Angriff, für eine jüngere Nutzergruppe ein Softwaretool zur Raspberry-Pi-Programmierung zu entwickeln. Die HTL-Schüler/-innen arbeiteten dabei eng mit den jüngeren Kollegen/-innen in einer Volksschule und NMS zusammen. Sie legten ein Interaktionsdesign für die jeweilige Anwendergruppe fest. Basierend auf Personas und Szenarien wurden Usability-Tests mittels Mockups sowie die Thinking-Aloud-Methode durchgeführt und anhand von Eyetracking-Aufnahmen wurden Heatmaps und Cursormaps erstellt und ausgewertet. Anhand dieser professionellen Zielgruppeneinbindung in der Entwicklungsphase wurden die Bedürfnisse und Denkprozesse der Jüngeren sowie deren informatisches Denken berücksichtigt.

Dieses Projekt zeigt - wie zahlreiche andere Projekte zu Lego-Mindstorms, Lego League uvm. -, dass Coding und Robotik in der Schule thematisiert werden und neben der fachlich-inhaltlichen Vermittlung in unterschiedlichen weiteren Fachdomänen (wie beispielsweise Cross-Age-Denken, Peer-Learning, Teamarbeit, Out-of-the-Box-Denken) ein Lernzuwachs bei den Schülern/-innen erreicht wird.

Beide IMST-Projekte „Sensitives Kuscheltier" und „Sensitives Kuscheltier goes Youngsters" von Christoph Brein (IMST-Projekt-ID 1569 sowie ID 1706) sind im IMST-WIKI unter http://www.imst.ac.at nachlesbar.

Kooperation mit Lego® Education Europe in der Primarstufe

In der IMST-LEIS-Kooperation mit Lego® Education Europe werden im Schuljahr 2017/2018 zwölf Volksschulen aus ganz Österreich unterstützt, die ihre innovativen Ideen im Rahmen von IMST durchführen. Zusätzlich werden weitere 21 Volksschulen angesprochen, die im Rahmen eines weiteren Projekts des Bildungsministeriums (Denken lernen – Probleme lösen, kurz DLPL)[2] Interesse zur Teilnahme bekundet haben. Sie setzen in ihren Unterrichtsszenarien bereits in der Primarstufe Akzente zu Computational Thinking, indem sie LEGO® Education WeDo2.0 Baukästen für fachliche Lernziele nutzen. In den teilnehmenden Schulen arbeiten die Schülerinnen und Schüler mittels Tablets und Bluetooth-Verbindung zu den WeDo-Devices in einer Software, die algorithmisches Denken erfordert. So sollen spielerische und haptische Aspekte angesprochen und etwaige Ängste (sowohl bei Lehrpersonen als auch bei Lernenden) vor der Auseinandersetzung mit Technik genommen werden. Schüler/-innen junger Altersstufen werden dabei an die Grundzüge von Computational Thinking herangeführt und deren Wirksamkeit für die

[2] *Das Projekt „Denken lernen – Probleme lösen (DLPL)" widmet sich der Etablierung von Education Innovation Studios (EIS) in Österreich zur Stärkung der informatischen Grundbildung mit Schwerpunkt Primarstufe (vgl. dazu http://zli.phwien.ac.at/projekt/dlpl/ (abgerufen am 17. Juli 2017))*

Bildung der Schülerinnen und Schüler evaluiert und mit Evidenzen belegt.

Vielfältige Aspekte und Lernprozesse werden angesprochen:

- Im Feld des Computational Thinking werden Strategien des algorithmischen Problemlösens, das Erfassen, Planen und Festlegen von Entwicklungs- und Prozessschritten, deren Überprüfung und Testung sowie das Verbalisieren und Dokumentieren von logischen Abfolgen erprobt und erlernt.
- Darüber hinaus wird auf Modellbildung von naturwissenschaftlichen Prozessen sowie die Abstraktion von Phänomenen in den beiden Lernfeldern des Sachunterrichts Natur und Technik eingegangen.
- Weiters werden Grundkompetenzen der Lernenden wie Teamfähigkeit, sprachliche Weiterentwicklung sowie die Haltung im Umgang mit Diversitäten und die Umsetzung von Inklusion in der Schule gefördert.

Dies führt beispielsweise zum schriftlichen Verfassen von Anleitungstexten und der mündlichen Beratung gleichaltriger oder auch jüngerer Schüler/-innen bei der Arbeit und leistet somit einen wichtigen Beitrag zur Entwicklung der Sprech- und Schreibkompetenzen.

Out-of-the-Box-Denken und Peer-Learning

Die Möglichkeit, dass Lernende als Peers und sogar als Cross-Age-Peers agieren, ist ein spannender Ansatz, der sich in den bereits durchgeführten IMST-Projekten als lernförderlich und selbstwertsteigernd erwies. Das Entwickeln von methodischen Schritten, die Umsetzung didaktischer Konzepte und das Hineinversetzen in die Denkmuster, Fachbereiche und Kompetenzen anderer, ein Out-of-the-Box-Denken, sowie das Entwickeln von Anleitungen für eine jüngere Zielgruppe erfordern und bewirken eine intensive Auseinandersetzung mit dem eigenen, selbst erworbenen Wissen. Es bleibt zu beforschen, unter welchen Rahmenbedingungen Coding und Robotik im Unterricht der Primarstufe allgemeingültig möglich ist, welche Lerneffekte nachhaltig erzielt werden und wie Lehrpersonen darauf vorbereitet werden können. Diese IMST-LEIS-Kooperation soll diesbezüglich einen Startpunkt setzen.

Autor

DI (FH) Gudrun HEINZELREITER-WALLNER
Lehramtsstudium für Textverarbeitung, Informations- und Officemanagement, Studium Engineering für Computerbasiertes Lernen an der FH OÖ, Campus Informatik, Kommunikation und Medien in Hagenberg, weitreichende Erfahrung als Lektorin an der FH OÖ, Campus Informatik, Kommunikation und Medien in unterschiedlichen Studiengängen (Software Engineering, Computer Based Learning sowie Kommunikation, Wissen und Medien). Lehrtätigkeit als Professorin für Angewandte Informatik an der Höheren Lehranstalt für wirtschaftliche Berufe in Freistadt vorwiegend im Zweig für Kommunikations- und Mediendesign. Mitarbeiterin im IMST-Themenprogramm „Kompetenzorientiertes Lernen mit digitalen Medien“.

Autor

Mag. Dr. Emmerich BOXHOFER
Lehramtsstudium für Mathematik, Leibeserziehung, Physik und Chemie, Informatik; Studium der Sozialen Verhaltenswissenschaften und der Erziehungswissenschaften; Doktoratsstudium Pädagogik; Leitung des Instituts für Forschung und Entwicklung an der Privaten Pädagogischen Hochschule der Diözese Linz. Wissenschaftlicher Leiter des IMST-Themenprogramms „Kompetenzorientiertes Lernen mit digitalen Medien“.

Autor

Mag. Alfons KOLLER
Lehramtsstudium für Mathematik sowie Geographie und Wirtschaftskunde (GW) an höheren Schulen, Lehrbefähigung für Informatik. Lehrender an der Pädagogischen Hochschule der Diözese Linz im Bereich der Geoinformatik und Fachdidaktik GW sowie in der Lehrerfortbildung. Organisatorischer Leiter des IMST-Themenprogramms „Kompetenzorientiertes Lernen mit digitalen Medien“.

Autor

Mag. Stefan HAMETNER
Lehramtsstudium Biologie und Erdwissenschaften an der Universität Wien, Lehrer am Bischöflichen Gymnasium Petrinum Linz, Lehrtätigkeit im Bereich der LehrerInnenausbildung für Biologie und Umweltkunde an der Pädagogischen Hochschule der Diözese Linz und im Cluster Mitte, Mitarbeit im IMST-Themenprogramm „Kompetenzorientiertes Lernen mit digitalen Medien“.

Scratch – Computerspiele selbst gemacht

Programmieren in der Volksschule. Muss das sein? Was sollen wir den Kindern denn noch alles beibringen? Sollen sie das doch in der NMS oder im Gymnasium lernen.

Diese oder ähnliche Sätze höre ich immer wieder. Die Erfahrungen mit meiner Klasse (dritte Schulstufe) zeigten jedoch, dass mit Hilfe geeigneter Tools und Software Ergebnisse erzielt wurden, die man normalerweise Kindern in der Volksschule nicht zutraut.

Die visuelle Programmsprache Scratch bietet eine Möglichkeit, spielerisch und lustbetont (denken) zu lernen. Ein Computerprogramm entsteht, indem einfach Anweisungen aneinander gereiht werden.

Ausgangspunkt:

Idealerweise arbeiten die SchülerInnen einzeln oder zu zweit am PC oder Tablet. Hilfreich ist ein Beamer mit dem die Lehrperson jeden Schritt vorzeigen kann.

Zuerst muss Scratch (ist kostenlos) heruntergeladen werden (https://scratch.mit.edu/download) oder man steigt direkt in das Programm über die Website (auch in Deutsch verfügbar) ein.

Ich hatte für jedes Kind meiner Klasse ein eigenes Netbook zur Verfügung und die Kinder waren es gewohnt, Programme unter meiner Anleitung selbständig herunterzuladen. Hat man diese Möglichkeit nicht, installiert man Scratch am Schulserver oder Klassen -PC bzw. -Tablet.

Start:

Zu Beginn erklärte ich meinen SchülerInnen (indem ich die Startseite von Scratch mittels Beamer an die Wand projizierte) die Grundfunktionen von Scratch. (Abb. 1) Wie starte ich Scratch, wo sehe ich den Ablauf meines Programms, wie füge ich Figuren und Hintergrund ein, welche Funktionen haben die Programmbausteine, wo finde ich diese und wie und wo starte ich mein kleines Programm und wie speichere ich? (siehe auch http://aufgabensammlung4.digikomp.at/mod/page/view.php?id=392&inpopup=1)

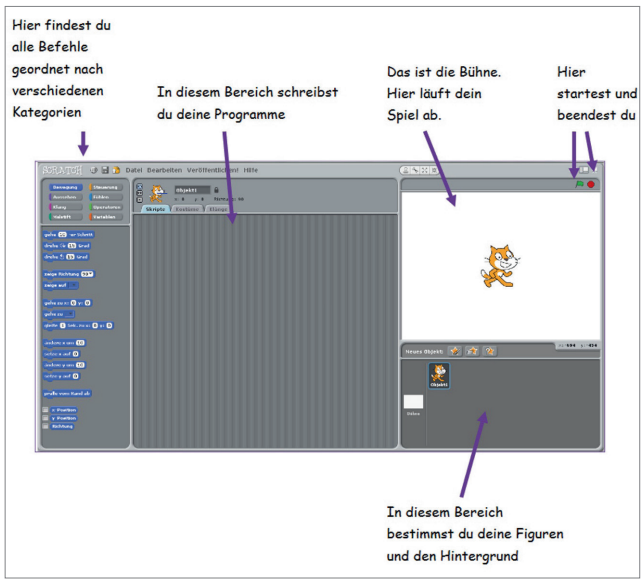


Abbildung 1

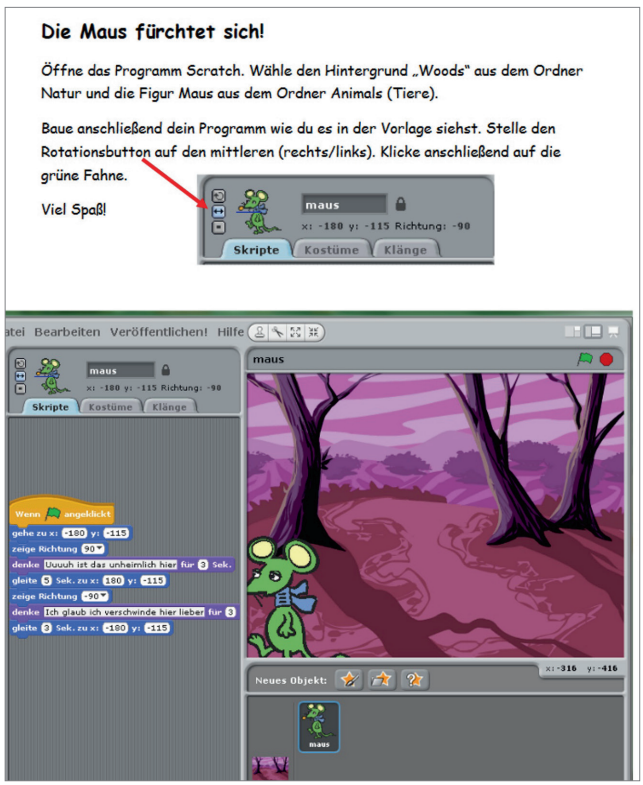


Abbildung 2

Ablauf:

Bewährt hat sich, den SchülerInnen ein fertiges, kleines und einfaches Programm vorzugeben, das sie nachbauen. Dadurch haben sie sofort ein Erfolgserlebnis (z.B. die Maus fürchtet sich; Abb. 2)

Durch die Veränderung der Koordinatenzahlen oder Sekunden bei Bewegungen, sehen die SchülerInnen sofort die Wirkung. So erarbeiten sie sich individuell viele Funktionen. (Figuren sprechen, verschwinden oder erscheinen lassen)

Meine Erklärungen konnten sie an der Projektionsfläche mitverfolgen und gleich nachmachen. Innerhalb kurzer Zeit probierten sie jedoch selbständig aus.

Einige entdeckten, dass man eigene Geräusche oder selbst Gesprochenes aufnehmen kann (Headset wird benötigt!). Sofort wollten das alle ausprobieren und so unterstützten und halfen sie sich gegenseitig.

Indem einige SchülerInnen selbständig oder in Kleingruppen arbeiteten, konnte ich mich in Ruhe denjenigen widmen, die mehr Unterstützung und Hilfe benötigten.

Da die Positionierung der Objekte ein sehr wichtiger Punkt ist, habe ich meinen SchülerInnen das Koordinatensystem etwas genauer erklärt. (Abb. 3+4)

Ergebnisse:

Die SchülerInnen hatten insgesamt acht Unterrichtseinheiten zur Verfügung, um ein Computerspiel zu programmieren. Sie arbeiteten konzentriert mit großem Eifer. Meine Arbeit beschränkte sich darauf, kleine Ratschläge zu geben oder Unterstützung zu bieten, wenn Probleme auftraten. Z.B. programmierte eine Schülerin ein Spiel, in dem Fische in einem Korb unter Wasser gefangen wurden, der an einem Seil hing und nach oben gezogen werden sollte. Das Problem war, dass die Teile (Korb, Fische, Seil) nicht ein zusammenhängendes Objekt bildeten und daher allen Teilen Befehle zugeordnet werden mussten, die zusammenpassten und richtig koordiniert waren. Ein anderer Schüler wollte unbedingt eine Star Wars Lego-Figur einbauen. Hier scheiterte ich jedoch und konnte das Problem nicht lösen. Viele Kinder installierten Scratch auch am Rechner zu Hause und arbeiteten in ihrer Freizeit an ihren Projekten weiter oder kreierten neue.

Am Ende reichte ich alle Projekte der Kinder, so wie sie diese abgaben, bei einem Wettbewerb ein und wir überstanden drei Ausscheidungsrunden.

Ziele:

Folgende Lernziele werden bei Scratch berücksichtigt:

- schöpferisch tätig sein
- rationale Denkprozesse anbahnen
- die praktische Nutzbarkeit der Mathematik erfahren
- Spielerisches Umgehen mit Zahlen und Operationen (Erfinden von Spielen, Durchführen von Strategiespielen)
- Räumliche Positionen und Lagebeziehungen
- Kennenlernen, Erproben und Anwenden von Ausdrucksmöglichkeiten in Bereichen wie Neue Medien, Spiel und Aktion
- Entwicklung des bildhaften Denkens und persönlichkeitsbezogener Eigenschaften wie Offenheit, Flexibilität, Experimentierfreude, Einfallsreichtum, Sensibilität, Konzentrationsfähigkeit, Ausdauer, Kooperationsbereitschaft...

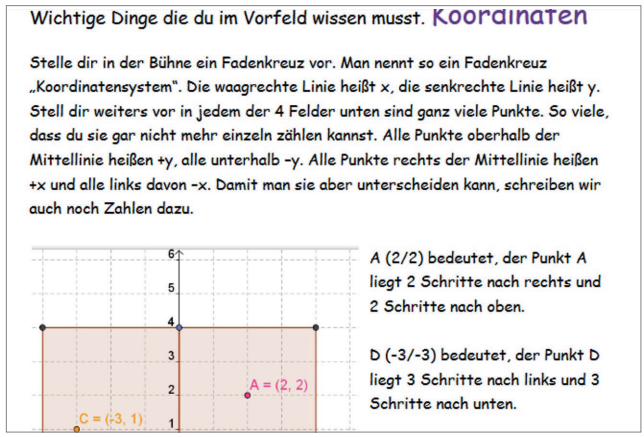


Abbildung 3

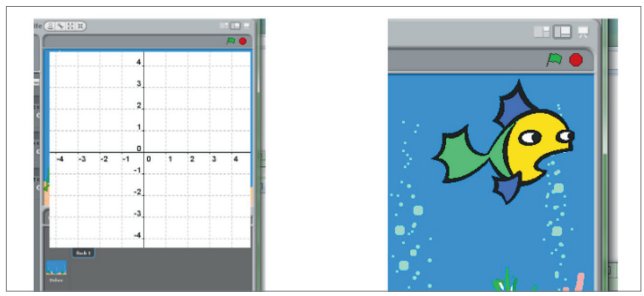


Abbildung 4

Tipps:

Auf der Scratch-Hauptseite gibt es Möglichkeiten, Vorlagen nachzubauen oder Tipps und Anregungen in einer Community auszutauschen. Die ist in Englisch aber man kann auch Deutsch auswählen.
https://scratch.mit.edu/
https://scratch.mit.edu/educators/

Auf der DigiKomp4 Seite http://aufgabensammlung4.digikomp.at/course/view.php?id=7 habe ich einige Videos zur Verfügung, gestellt die auch ganz hilfreich sein können.

Sehr ans Herz legen kann ich Lehrpersonen einen Moodle-Kurs der Virtuellen PH (http://www.virtuelle-ph.at/) um Scratch kennenzulernen und unter einer professionellen Anleitung selber auszuprobieren. Die Kurse sind kostenlos (Registrierung an der Virtuellen PH ist notwendig) und werden mehrmals im Jahr angeboten.

Autor

Prof.in Mag.a Silvia Nowy-Rummel
Referentin für feministische Bildung und Politik. Lehramtsstudium für Volksschulen, Pädagogikstudium; Lehrende an der PH Salzburg Stefan Zweig, am Institut für Unterrichtsentwicklung und Didaktik im Bereich Sachunterricht sowie am Institut Gesellschaftliches Lernen und Politische Bildung im Bereich Medienpädagogik; Mitarbeit im Bundeszentrum für Geschlechterforschung und -pädagogik.

Konsumierst du noch oder programmierst du schon?

Von Scratch bis micro:bit

Unter meinem obigen Motto versuche ich seit mehr als zehn Jahren, SchülerInnen aus der Konsumfalle zu locken und auf interessante, forschende Entdeckungsreise (inquiry-based learning) zu gehen. Und ich bin stolz darauf, dass dies alters- und geschlechtsunabhängig immer wieder gelingt!

Mit den heutigen Webtools ist dies sogar sehr viel einfacher geworden. Sei es das Herstellen eines selbstkomponierten Werbeclips, einer lässigen Animation zur Funktionsweise eines Gerätes / Vorganges, erfinderisches Storytelling statt nur immer Text-Aufsätze zu schreiben, eigene Spiele aus einem Storyboard zu designen (Game-Development) oder das Tüfteln und kreative Problemlösen bei kniffligen Alltagsaufgaben – all dies wäre im Informatik-Unterricht möglich, sofern es dies denn als durchgehendes Pflichtfach in Österreich gäbe.

Die meisten Staaten in der EU und in Übersee haben ein solches kreatives Pflichtfach schon bereits seit der ersten Klasse (Primary School) als Unterrichtsfach etabliert. Dies zeigen die vielen Webtools, Unterrichtsbehelfe und Initiativen der sogenannten Pioneer-Teacher & Sponsoren (z.B. UK mit BBC & micro:bit, MIT-AppInventor zur einfachen und raschen Eigenentwicklung von Apps für Smartphones,…).

Sprachsteuerung (z.B. Siri & Co), Übersetzungshilfen (z.B. Google-Translator), Gemeinschaftsspiele (Teamwork-Onlinespiele), spannende medial aufbereitete Lerninhalte (e-Content) und Online-Tutoren/Assistenten, Robotik und autonom selbstfahrende Autos - sei es der komfortable Rasenmäher oder die Seitwärts-Einparkhilfe auf Knopfdruck – sind längst Alltag für unsere SchülerInnen.

Statt viele dieser interessanten Bereiche im Unterricht wegzulassen oder statt immer nur über unsere inaktiven und leider oft auch inkompetenten PolitikerInnen in Österreich zu klagen, rege ich hiermit an, selbst im Team mit den SchülerInnen diese neuen Bereiche zu erforschen!

Sie werden staunen, wie einfach ein Start mit den überquellend vorhandenen und didaktisch aufbereiteten Materialien und Werkzeugen eigentlich ist. Dieses erforschende Lernen & Programmieren ist in nahezu jedem Unterrichtsfach möglich und wird sogar noch vom Ministerium in Österreich gefördert.

Zögern Sie also nicht, neben dem reinen „Lern-Konsumieren" auch Kreativität über Informationstechnologie in den Unterricht einzubauen – es macht einfach auch Spaß!

Tipps:

- Auch in SCRATCH oder micro:bit werden Kurzanleitungen und Blöcke nun wie üblich rechts in der Arbeitsumgebung eingeblendet. Dem selbstgesteuerten Experimentieren, Erforschen und kreativen Programmieren steht nun nichts mehr im Wege!
- Speziell für die Altersgruppe der 6- bis -10 Jährigen eignet sich „Scratch Jr." Eine nochmals vereinfachte Scratch-Oberfläche ist als spezielle App (iOS & Android) für Tablets & Smartphones kostenlos erhältlich. Ziel: einfach und spielerisch erste Programmiererfahrungen zu sammeln. Kinder können ihre eigenen interaktiven Geschichten und Spiele am Tablet oder PC erstellen. https://www.scratchjr.org/
- Microsoft-/Google-Translator benutzen oder Übersetzung im Browser einschalten, damit Inhalte statt in Englisch in Deutsch angezeigt werden. https://www.bing.com/translator bzw. https://translate.google.at
- Es sind keine Anfragen von Lehrpersonen zur Softwareinstallation auf Schul-PC notwendig, da alle Webtools bereits im Browser lauffähig sind.
- Den Browser (Internet-Explorer, FireFox, Chrome,…) einfach auf Deutsch einstellen, dann wird/werden (so vorhanden) automatisch die Hilfe & Textblöcke in Deutsch statt Englisch angezeigt.
- Speichern der Programme & Projekte als Datei auf den PC ist in all diesen Webtools möglich.
- Eine eigene Arbeitsoberfläche sowie das Teilen und Sichern von Projekten ist mit Online-Account (d.h. kostenloser Registrierung) möglich.

Quellen für weiterführende Unterlagen und Vertiefungen:

- https://www.scratchjr.org/ „Scratch Jr." als Scratch für Volksschulen/Primary Schools
- https://chrome.google.com/webstore/detail/scratchjr/oipimoeophamdcmjcfameoojlbhbgjda
- Scratch Jr Browserversion für Chrome https://scratch.mit.edu/
- http://www.swisseduc.ch/informatik/programmiersprachen/scratch_werkstatt/
- http://scratchx.org/ Experimentelle Erweiterungen zu Scratch: Spotify, micro:bit, Text-to-Speech, Arduino, Robotik Fischer-Technik, Intel RealSense, iRobot Create, Raumstation ISS Tracker,…
- https://llk.github.io/microbit-extension/ Scratchx + micro :bit
- http://microbit.org/de/2017-06-20-na-launch/ mehr als 2 Millionen SchülerInnen mit Micro:bit
- https://makecode.com/ Microsoft MakeCode

Hands-on computing education für MicroBit, Adafruit, Minecraft, Sparkfun uvm.

- https://makecode.microbit.org/ Programmierung micro:bit mit Auswahlmöglichkeit Text-orientiert (für Fortgeschrittene) oder Block-orientiert (für Starter).
- http://microbit.org/de/ Auch ohne Hardwareplatine als Webtool besonders spannend.
- http://microbit.org/de/code/ Verwirkliche deine Träume & Ideen durch Programmieren
- https://scratch.mit.edu/info/ext_download/ Das Scratch Extensions Browser Plugin ermöglicht LEGO WeDo, PicoBoard und andere USB-Geräte mit der Online-Version von Scratch, zu verwenden.
- https://wiki.scratch.mit.edu/wiki/Scratch_API_(2.0) Scratch API zur Erweiterung und Steuerung für Fortgeschrittene
- http://snap.berkeley.edu/ Webtool SNAP von der Universität Berkeley/CA als Vertiefung mit Eigenbau von Blöcken (Funktionen und Unterprogramme). Build Your Own Blocks.
- http://snapapps.github.io/ SNAP Apps für Tablets & Smartphones
- https://education.minecraft.net/get-started/download/
- https://www.kodugamelab.com/
- http://www.ahs-informatik.at
- http://inf.elearningcluster.at
- http://gmk.elearningcluster.at Sammlung Gamemaker, Scratch, Snap, Microbit
- https://www.codeclub.org.uk/ Beispiele und Unterlagen UK Kids online.

Didaktisches Beispiel: Scratch – Katz und Maus

Schulstufe: ab der 4. Schulstufe (VS) und ab der 5. Schulstufe (AHS, SEK 1)

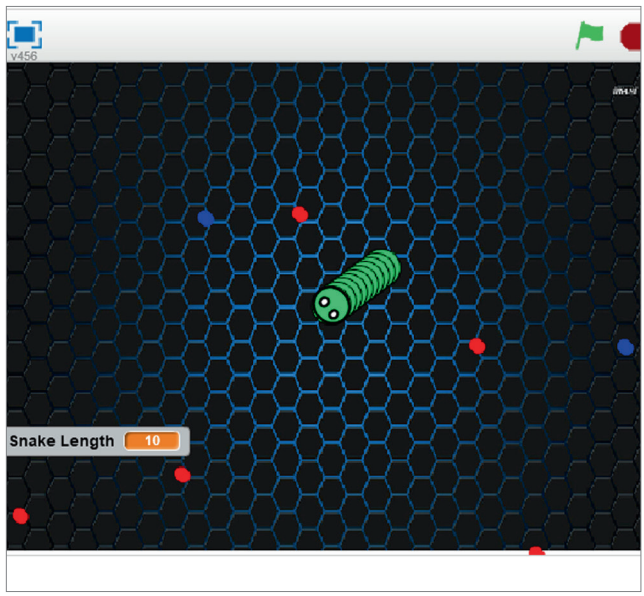
Kompetenzraster INF/IKT/DigiKomp: 4. Konzepte und 4.3 Automatisierung von Handlungsanweisungen
Zeitbedarf: ab 1 - 2 UE

Anmerkungen und Tipps:

- Webtool Scratch: https://scratch.mit.edu/
- Scratch ist ein Webtool, welches geräteunabhängig in jedem Browser wie Internetexplorer, Firefox, Chrome etc. ohne zusätzliche Installation verwendet werden kann. Auf Tablets und Smartphones sind entsprechende Apps (iOS, Android) kostenlos erhältlich.
- Scratch ist ein Projekt der Lifelong-Kindergarten-Gruppe am Media-Lab der Universität MIT in Boston/USA und besteht seit 2007. Als erziehungsorientierte visuelle Programmiersprache für Kinder und Jugendliche inklusive Entwicklungsumgebung und Online-Community-Plattform ist diese in jedem Unterricht weltweit kostenlos einsetzbar.
- Die Sprachauswahl ist ganz unten im Webtool Scratch auf DE einstellbar!
- Überblick Scratch für Lehrkräfte: https://scratch.mit.edu/educators/

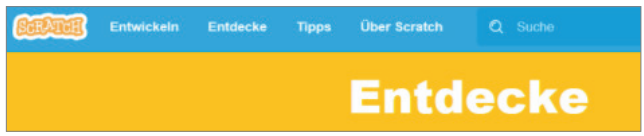
Weiterführende Quellen:

- https://scratch.mit.edu/info/cards/
- http://start-coding.de/tutorials/programmieren-lernen-mit-scratch/
- http://www.funlearning.de/



Aufgabe 1: Entdecken und Starten

1. Starte im Browser die Programmierumgebung SCRATCH: https://scratch.mit.edu/
2. Stelle am unteren Ende des Webtools die Landessprache (DEUTSCH) ein.
3. Lade und erkunde z.B. folgende Programme/Projekte: (unter ENTDECKE oder SUCHE)!

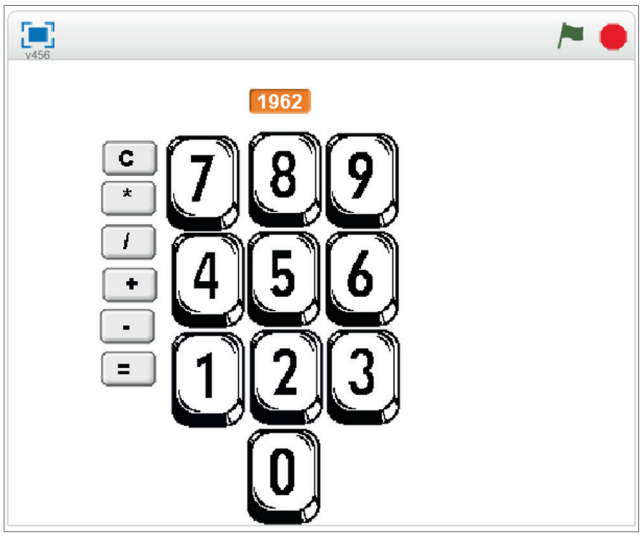


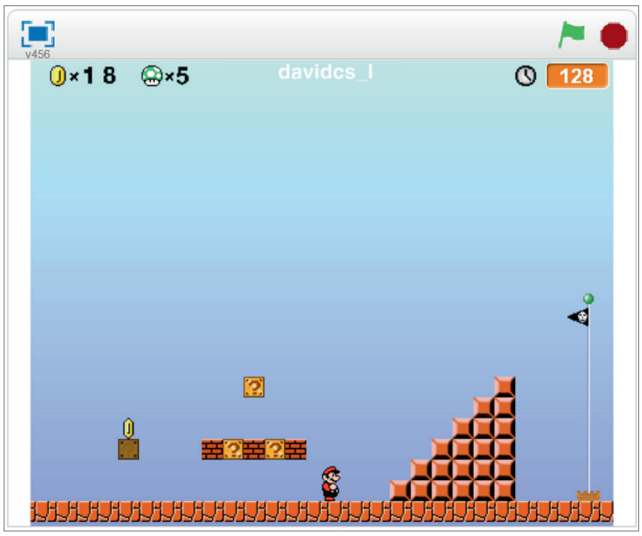
START erfolgt mit grüner Flagge.

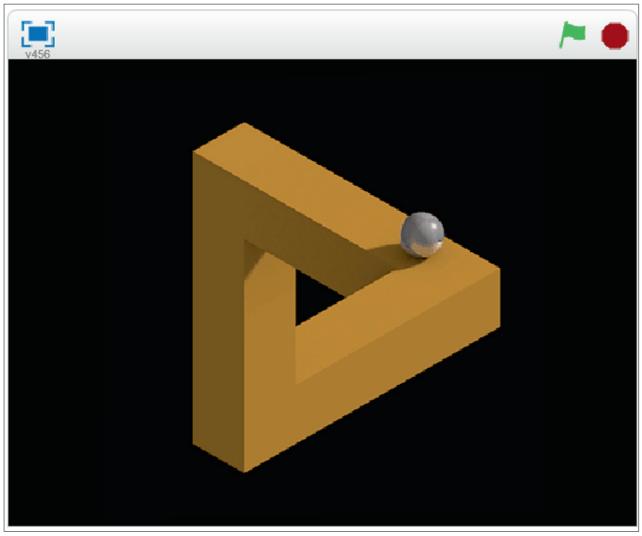
- Slither.io on Scratch: https://scratch.mit.edu/projects/105618680/
- Run, Rudolph Run: https://scratch.mit.edu/projects/89010705/
- Super Mario Bros: https://scratch.mit.edu/projects/167171395/
- Rechner/Calculator: https://scratch.mit.edu/projects/11520972/
- Animations: https://scratch.mit.edu/projects/94311916/

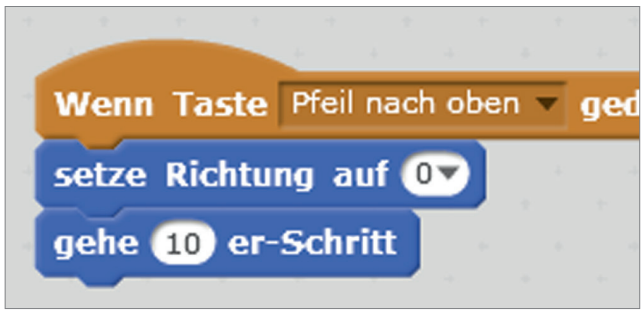
4. Betrachte Aufbau und eingeblendete Arbeitsbereiche der Programmierumgebung (SDK) genauer!

5. Öffne und erforsche weitere Beispiele. Hole dir Tipps und informiere dich über das Scratchprojekt in der obersten Menüleiste.





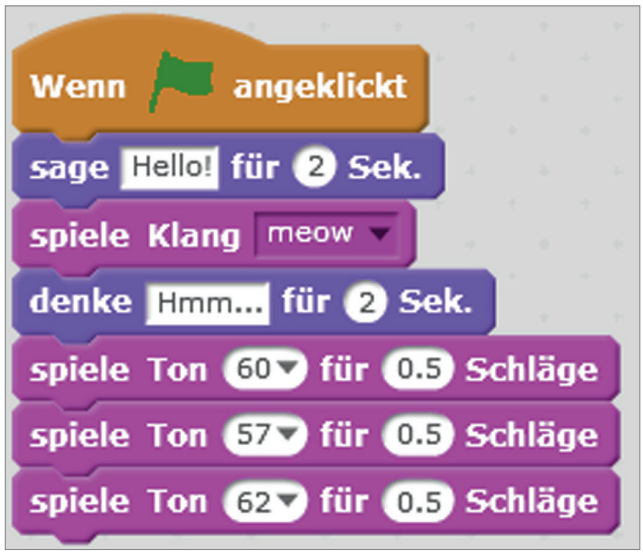




Aufgabe 2: Bewegung

1. Starte ein neues leeres Grundprogramm unter ENTWICKELN!
2. Erstelle im Programmierbereich SKRIPTE mit den Objekten aus der Palette EREIGNISSE, BEWEGUNG und AUSSEHEN nachfolgende Begrüßung und Tastensteuerung:

3. Starte das Programm (Skript) durch Anwählen der GRÜNEN FLAGGE und manövriere durch Betätigen der entsprechenden PFEIL-Tasten!



4. Ergänze dein Programm durch Sound aus der Objekt-Palette KLANG und experimentiere damit ein wenig! In der Abteilung KLÄNGE kannst du vorhandene Klänge und Hintergrundmusik laden, unter SKRIPTE einbauen und beim Programmstart abspielen.

Aufgabe 3: Katz und Maus auf Jagd

1. Erstelle unter SKRIPTE mit der Objekt-Palette STEUERUNG und BEWEGUNG eine fortlaufende Verfolgungsbewegung der Katze in Richtung deines Mauszeigers!
2. Starte und teste dein Programm (Skript) mit der grünen Start-Flagge!
3. Erweitere dein Programm durch Laden einer NEUEN FIGUR aus der Bibliothek um eine Maus.

4. Wähle einen geeigneten Hintergrund unter NEUES BÜHNENBILD aus der Bibliothek.
5. Erweitere dein Programm durch eigene Kreationen, teste diese und speichere im Menü DATEI deine Arbeit auf den PC.

Hinweis 1: Beachte die Werkzeuge zum Löschen und Duplizieren von Programmteilen (Skripts):

Hinweis 2: Jede Figur hat seine eigenen SKRIPTE mit KOSTÜME und KLÄNGE und kann somit als Objekt im Programm agieren! (objektorientierte Programmierung mit Ereignissteuerung und Eigenschaften)

Hinweis 3: Auch ohne Anmelden/Account kann das erarbeitete Programm im Menü DATEI auf den PC gespeichert werden. Teamwork und Sicherung online ist jedoch nur mit (kostenlosem) Account möglich.

Autor

Prof. Mag.rer.nat.
Hubert Egger
AHS-Lehrer für Mathematik, Physik, Informatik, INF-ARGE-Leitung & eLC-V & LehrerInnen-Fortbildung PHV: CEO www.egger.ac und Hobby-Pilot

Programmieren – ein Weg zu Problemlösekompetenz und informatischer Bildung

Mit der zunehmenden Digitalisierung unserer Lebenswelt entstehen neue Herausforderungen, denen sich Bildungsinstitutionen stellen müssen. Informatische Bildung gilt als Bestandteil allgemeiner Bildung für eine verantwortliche Gestaltung der Zukunft in Selbstbestimmung (Dörge 2015, S. 214). Der vielfach beschriebene „Digital Gap", die Kluft zwischen reinen digitalen „Anwendern" und „Gestaltern", kann nicht als unveränderbare Realität hingenommen werden. Die Pädagogische Hochschule Nordwestschweiz fordert auf der Webseite www.imedias.ch, dass die Jugendlichen nicht reine Konsumenten von Informatikprodukten, sondern kompetente und aktive Teilhaber dieser mächtigen Werkzeuge werden.

Andererseits rückt aber auch das kreative Lösen von Problemen in den Mittelpunkt von Bildung. Problemlösekompetenz gilt als eine Schüsselkompetenz für mathematische Bildung. Regina Bruder von der TU Darmstadt merkte dazu im Rahmen einer Mathematiktagung bereits 2003 an: Mathematisches Problemlösen gilt (nicht erst seit TIMSS und PISA) als defizitär, zählt aber […] zu den drei Grunderfahrungen, die den allgemeinbildenden Charakter des Mathematikunterrichts legitimieren. Die mit den Bildungsstandards eingetretene Fokussierung auf Kompetenzerwerb bietet die Chance, dem Problemlöselernen im Fachkontext oder auch fachübergreifend die notwendige Aufmerksamkeit zu widmen (Vgl. Bruder & Collet 2011, S. 2).

Kann eine (schüler/innengerechte) Einführung in die Programmierung neben informatischer Grundbildung auch Problemlösekompetenz fördern?

Was unterscheidet eigentlich ein Problem von einer Aufgabe? Dunker (1935, S.1) schreibt, dass ein Problem entsteht, wenn ein Lebewesen ein Ziel hat und nicht ´weiß´, wie es dieses Ziel erreichen soll. Ein Denkprozess wird dadurch angeregt. Problemlösendes Denken erfolgt, um Lücken in einem Handlungsplan zu füllen, der nicht routinemäßig eingesetzt werden kann (Funke 2003, S. 25). Programmieren, das Entwickeln einer Handlungsanweisung (Algorithmus) zum schrittweisen Übergang von einem Ausgangs- zu einem Zielzustand, ist höchst kreatives, systematisches Problemlösen. Denn Probleme, vor denen eine Programmiererin oder ein Programmierer stehen, haben nie eine vorgegebene Lösung. Beim Entwickeln eines Programms geht es darum, ein Problem zu analysieren und in kleinere Teilprobleme zu zerlegen, diese zu lösen, und die Teillösungen wieder zu einem Ganzen zusammenzufügen. Das erste entwickelte Programm kann so lange überarbeitet und verbessert werden, bis es den eigenen Ansprüchen genügt. Ein Computerprogramm ist gleichsam die in einer speziellen Sprache verfasste Anleitung zum Lösen eines Problems durch einen Computer (Boles 2006, S. 21). Wichtig ist dabei nicht die Verwendung einer bestimmten Programmiersprache, sondern das Entwickeln von Lösungsalgorithmen. Ein Vergleich zum Sport sei hier gestattet: Wer in jungen Jahren die Gelegenheit erhält, vielseitige Bewegungserfahrungen zu sammeln, wird in Extremsituationen ein großes Repertoire an vielfältigen Bewegungsmustern nützen und damit unter Umständen Verletzungen vermeiden können. Analog dazu kann auch für das Entwickeln von Lösungswegen bzw. Algorithmen festgehalten werden: Die selbst entwickelten Denk- und Lösungsmuster werden gleichsam in einer „internen Datenbank" gespeichert. Bei neuen Problemstellungen kann auf diese Vorerfahrungen zurückgegriffen werden.

Grafische Programmierumgebungen sind für den Einstieg in die Programmierung besonders geeignet. Sie bieten den Schülerinnen und Schülern motivierende Aufgabenstellungen in lustigen Programmierwelten und erleichtern die Fehlersuche. Und: Es gibt keine Syntaxfehler. Das manchmal sehr mühsame Erlernen der Besonderheiten und des Befehlswortschatzes einer Programmiersprache kann unterbleiben. Aus der Vielzahl der im Internet zur Verfügung stehenden Angebote sollen jene vorgestellt werden, mit denen ein Einstieg in die Programmierung ohne viele Vorerklärungen begonnen werden kann.

Programmieren mit Lightbot

Für den Computer wird auf der Webseite lightbot.com/flash.html eine kostenfreie, wenn auch eingeschränkte, Version angeboten. Die App für Tablets oder Smartphone ist zwar für alle Betriebssysteme verfügbar, aber leider kostenpflichtig. Das Grundziel in dieser Programmierumgebung ist, einen „Außerirdischen" in seiner Welt so zu steuern, dass mit möglichst wenigen Programmierschritten vordefinierte blaue Kacheln zum Leuchten gebracht werden. Im ersten Abschnitt werden selbsterklärend die

Grundlagen vermittelt. Mit neuen Problemstellungen werden neue Werkzeuge zur Verfügung gestellt, die in dieser Programmierumgebung auch gleich verwendet werden müssen. Mit fünf Werkzeugen (vor, Licht ein, links drehen, rechts drehen, hinauf-/hinunterspringen) müssen acht Problemstellungen gelöst werden.

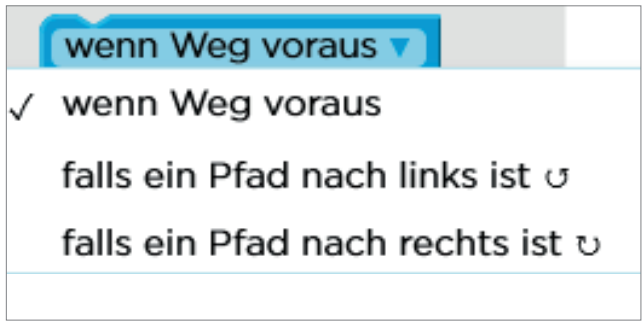
Der zweite Abschnitt ermöglicht mit der Einführung von Prozeduren einen Zugang zu strukturierter Programmierung. Die Lösungswege müssen nun dahingehend analysiert werden, inwieweit sich Programmsequenzen (Muster) wiederholen. Diese Muster können in Prozeduren (Unterprogramme) ausgelagert werden und im Hauptprogramm dann wieder aufgerufen werden. Im letzten Abschnitt werden schließlich Schleifen über die rekursive Programmierung eingeführt. Eine Prozedur wird dabei innerhalb dieser Prozedur aufgerufen. Auf das Festlegen einer Abbruchbedingung wird dabei verzichtet, weil am Rand der „Welt" automatisch gestoppt wird. Für Lehrerinnen und Lehrer werden im Internet auch Lösungen für alle Problemstellungen angeboten.

Links:

- https://lightbot.com/flash.html
- https://lightbot.com/LightbotSolns.pdf - Lösungen für Lehrkräfte

Angry birds – Ein motivierender Zugang

Den „zornigen Vogel" oder andere Tiere bei den Aufgaben zu steuern, macht allen Schülerinnen und Schülern großen Spaß. In einer Scratch sehr ähnlichen Umgebung müssen Aufgaben mit aufsteigendem Schwierigkeitsgrad gelöst werden. Ähnlich wie bei Lightbot werden bei neuen Aufgaben neue Befehle oder Programmierstrukturen vorgestellt. Durch die grafische Darstellung wird, ähnlich wie bei Scratch, das Verständnis für Schleifen und Verzweigungen gefördert. Der Kursaufbau ist selbsterklärend und kann problemlos ohne zusätzliche Lehrerhilfe abgearbeitet werden.



In einer der letzten, etwas anspruchsvolleren Aufgabe soll das Eichhörnchen mit den vorgegebenen Befehlen bzw. Strukturen zur Eichel geführt werden. In dem Listenfeld der Verzweigung können die Bedingungen ausgewählt werden. Ziel ist auch bei diesen Problemstellungen wieder, eine Lösung mit der geringstmöglichen Anzahl an Programmschritten zu entwickeln. Werden die Aufgaben gelöst, kann eine personalisierte Urkunde erstellt und ausgedruckt werden.

Links:

- https://studio.code.org/hoc/1
- Materialien: http://aufgabensammlung8.digikomp.at/course/view.php?id=263

Silent teacher – eine Vorübung zum Coding

Geht es bei den bisher vorgestellten grafischen Programmierumgebungen primär um das Entwickeln von Lösungskonzepten mit den vorhandenen Werkzeugen (Befehlen), so stehen nun die Grundlagen von Programmiersprachen und das Lesen von Programmcode im Fokus. Silent teacher ist eine spielerische Einführung in Programmierkonzepte ohne viele Erklärungen. Folgende Basiskonzepte der Programmierung werden dabei behandelt:

- Verschiedene Variablentypen - Zahlen, Zeichenketten, Boolsche Variable, Listen

```
var a = 1;
a + 3;
? 
```

```
var a = 'gh';
var b = 'gr';
a + b;
? 
```

```
var a = 0;
a = a + 1;
a + 3;
? 
```

- Eigenschaften von Variablen

```
var a = 'eijkr';
a.length;
```

- Unterschied zwischen zuweisendem und vergleichendem Gleichheitszeichen

```
14 !== 2;
? true
```

- Funktionen und Bedingungen

```
function hello (a, b) {
    return a * b;
}

hello(2, 3);
? 
```

```
var a = 3;
if (1 < 3) {
    a = 5;
}
a + 2;
? 
```

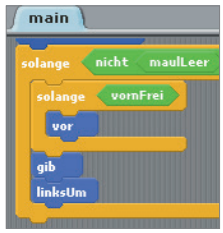
In einer Serie von Fragestellungen können Anfänger/innen die Grundlagen zum Arbeiten mit der Programmiersprache Javascript erlernen. In gleicher Weise kann auch das Lesen von Programmen geübt werden. Dadurch kann ein grundlegendes Verständnis ermöglicht werden. Welchen Ausgabewert wird dieses Programm erzeugen?

```
function hi (a, b) {
    if (a < b) {
        return a + b;
    } else {
        return a * b;
    }
}
```

Als besonderer Bonus kann nach Abschluss des Kurses ein nett gestaltetes personalisiertes Zertifikat ausgedruckt werden. Möchte man diesen spielerischen Einstieg ins „richtige Programmieren" fortsetzen, kann das Java-Hamster-Modell empfohlen werden. In dieser „Welt" wird ein virtueller Hamster gesteuert, der Körner ablegen und aufsammeln kann. Es ist mit diesem Modell aber auch möglich, ab der 8. Schulstufe in die objektorientierte Programmierung mit einer Java ähnlichen Programmiersprache mit ganz wenigen Grundbefehlen, Boolschen Variablen und Grundstrukturen einzusteigen.

```
void main() {
    while (!maulLeer()) {     // lauf solange Körner im Maul
        while (vornFrei()) {  // lauf solange vorne frei ist
            vor();
    }
    gib();
    linksUm();                // drehe dich um 90 Grad nach links
    }
}
```

Schülerinnen und Schüler ab der 5. Schulstufe können den Hamster auch in einer Scratch nachempfundenen Programmierumgebung bei seinen Aufgaben steuern.



Links:

- http://silentteacher.toxicode.fr/hourofcode
- http://www.java-hamster-modell.de/simulator.html

Literatur:

Boles, D. (2006). Programmieren spielend gelernt mit dem Java-Hamster-Modell (3. Aufl.). Wiesbaden: Teubner Verlag

Bruder, R. (2003). www.math-learning.com/files/Tagdma_fulda.pdf

Bruder, R. & Collet, C. (2011). Problemlösen lernen im Mathematikunterricht. Berlin: Cornelsen Verlag Skriptor

Dörge, C. (2015). Informatische Schlüsselkompetenzen. Potsdam: Universitätsverlag

Duncker, K. (1935). Zur Psychologie des produktiven Denkens. Berlin: Julius Springer

Fachhochschule Pädagogische Hochschule Nordwestschweiz: www.imedias.ch

Funke, J. (2003). Problemlösendes Denken. Stuttgart: Kohlhammer.

Autor

Hubert Pöchtrager
ist Lehrer für Mathematik, Informatik, Bewegung und Sport. Neben der Unterrichtstätigkeit betreut er im Bundeszentrum eEducation die oö. Schulen in der Sekundarstufe I in den Bereichen informatische Grundbildung und Einsatz digitaler Medien im Unterricht. An der Pädagogischen Hochschule OÖ leitet er die Landesarbeitsgemeinschaft für Mathematik und koordiniert die Mathematikfortbildung. Überdies ist er Mitarbeiter am Linzer Zentrum für Mathematik Didaktik an der Johannes Kepler Universität.
E-Mail:
hubert.poechtrager@ph-ooe.at

Beebots als niederschwelliger Zugang zu Coding und Robotik

Beebots wirken nicht nur auf Kinder sehr anziehend. Über diese Roboter-Bienen kann auch für wenig affine Pädagoginnen und Pädagogen der Primarstufe der Einstieg ins Thema Robotik & Coding gelingen. Im Rahmen der Lehrveranstaltung "Informatische Bildung" an der Pädagogischen Hochschule Niederösterreich haben Studierende des Lehramts Primarstufe einen Zugang zum Thema gefunden und didaktisch-methodische Szenarien für den Einsatz der Beebots in verschiedenen Unterrichtsfächern entwickelt.

Fachliche und überfachliche Kompetenzen fördern

Längst hat die Digitalisierung alle Lebensbereiche der Gesellschaft erfasst. Schule muss auf diesen Umstand reagieren und unsere Kinder auf die Welt von morgen vorbereiten.

Die Beschäftigung mit Coding und Robotik kann dazu einen wichtigen Beitrag leisten, da die dabei entwickelten Kompetenzen für viele Berufe Voraussetzung sind (Brandhofer, 2016). Als Teil der informatischen Bildung helfen diese nicht nur Kindern und Jugendlichen, sondern auch Erwachsenen, die automatisierte und von Algorithmen gesteuerte Verarbeitung von digitalen Informationen zu verstehen (Zorn et al., 2013).

Zusätzlich werden auch mathematische und logische Fähigkeiten und Fertigkeiten trainiert, sowie das Vorstellungsvermögen angeregt. Schon in der Primarstufe lassen sich die Beebots und andere robotic toys gewinnbringend einsetzen, etwa zur Förderung der Sprache und dem kreativen Lösen von Problemstellungen (Stöckelmayr et al., 2011).

Die Beebots bieten sich an, da sie für Kinder einen besonders einfachen Zugang zum Thema Programmieren darstellen. Die Gestaltung ist kindgerecht und es wird kein zusätzliches digitales Gerät benötigt. Die Programmierung der Roboter-Biene funktioniert über vier auf dem Rücken angebrachte orangefarbene Pfeile, die Bewegungen nach vorne, hinten, links und rechts ermöglichen. Durch das Drücken der Pfeile wird dem Beebot ein Bewegungsablauf eingespeichert, und mit der Taste GO wird das Programm abgespielt. Bis zu 40 Befehle können eingegeben werden. Möchte man ein neues Programm einspielen, drückt man die Taste CLEAR.

Die Beebots sind sehr einfach zu nutzen und ermöglichen Kindern, Jugendlichen und Erwachsenen einen haptischen Zugang zum Thema Coding und Robotik. Durch die sofortige Reaktion in Form von Bewegung und dem Erleben von Kompetenz ("Ich kann das!") kann die Motivation und das Interesse für den Bereich Technik hochgehalten werden (Garcia-Penalvo, 2016). Für Lehrerinnen und Lehrer bietet sich ein breites Feld an Möglichkeiten, wie die Beebots kreativ und konstruktivistisch im Unterricht eingesetzt werden können. Die dafür notwendigen Kompetenzen werden angehenden Lehrkräften an der Pädagogischen Hochschule Niederösterreich im Fach Informatische Bildung vermittelt.

Selbst entwickelte Einsatzszenarien für den fächerübergreifenden Unterricht

Das Fach Informatische Bildung ist im Curriculum des Bachelorstudiums Primarstufe an der Pädagogischen Hochschule Niederösterreich im zweiten Semester angesiedelt. Die Auseinandersetzung mit den Beebots erfolgte im Studienjahr 2016/17 etwa gegen Mitte des Sommersemesters. Nach einer kurzen theoretischen Einführung kam es sehr schnell zur ersten Erprobungsphase durch die Studierenden mit Hilfe des angebotenen

Studierende fertigen Material für den Einsatz von Beebots

Materials: Befehlskarten mit Abbildungen der einzelnen Funktionstasten des Beebots, Raster im Format DIN A4 mit vorgezeichneten Strecken in verschiedenen Schwierigkeitsstufen sowie diverse auf Flipchart-Papier vorgezeichnete Raster mit einfachen Übungsaufgaben.

Im zweiten Schritt entwickelten die Studierenden in Gruppenarbeit eigene methodische Ansätze für den Einsatz von Beebots in den unterschiedlichen Schulstufen der Primarstufe. Konkret sollten didaktische Szenarien entstehen, den Einsatz von Beebots fächerübergreifend an einzelne Themen des Unterrichts anzuknüpfen. Auf diese Weise werden einerseits die durch den Umgang mit Beebots angesprochenen Kompetenzen wie logisches Denken, Vorausdenken und kreative Problemlösung trainiert und andererseits Inhalte aus einzelnen Gegenständen erarbeitet bzw. vertieft.

Jeder Gruppe wurde dafür vom Lehrveranstaltungsleiter eines der vier Unterrichtsfächer Mathematik, Deutsch, Sachunterricht und Musikerziehung zugewiesen. Das konkrete Thema und die Schulstufe wählte jede Gruppe selbst. Als Vorgabe für die zu entwickelnden Materialien und Szenarien galt, dass das Selbstlernen der Schüler/innen im Mittelpunkt steht und Selbstkontrolle bzw. Peer-Kontrolle möglich ist. Eine kleine Auswahl an Aufgaben aus den entstandenen Szenarien:

Mathematik:

Malreihen aus 3 und 4: Auf Kärtchen sind die einzelnen Ergebnisse der Malreihe aus drei und vier aufgeschrieben. Die Lehrperson legt die Kärtchen verteilt auf einen Raster auf Flipchart-Papier mit vier mal fünf Feldern auf. Das Kind soll die Biene so programmieren, dass sie die einzelnen Ergebnisse der Malreihe aus 3 (oder 4) in aufsteigender Reihenfolge besucht. Auf jedem Ergebnisfeld wird eine Pause programmiert. Einfachere Variante: Nur eine Malreihe wird aufgelegt, und auf einige freie Felder werden Hindernisse eingebaut. Schwierigere Variante: Die Ergebnisse müssen in absteigender Reihenfolge besucht werden. Selbstkontrolle: Auf einer Lösungskarte stehen die Ergebnisse der Malreihen in aufsteigender Reihenfolge. Das Kind kann während der Fahrt der Biene die Ergebnisse kontrollieren.

Deutsch:

Reimwörter: Auf einem Flipchart-Raster mit vier mal fünf Feldern sind Reimwörterpaare in die einzelnen Felder geschrieben bzw. gezeichnet. Jedes Reimwörterpaar besteht jeweils aus einer Zeichnung und einem geschriebenen Wort. Die Lehrperson nennt eine Zeichnung. Das Kind setzt die Biene auf dieses Feld und schickt sie von dort aus zum geschriebenen Reimwort. Bei Ankunft liest das

Studierende testen selbst hergestelltes Unterrichtsmaterial

Kind beide Wörter vor. Varianten: Die Biene besucht mehrere Reimwortpaare hintereinander. Die Biene besucht alle Wörter mit einem bestimmten Anfangsbuchstaben/Vokal/etc.

Sachunterricht:

Thema Blumenwiese: Auf einem Flipchart-Raster mit vier mal fünf Feldern sind Blumen sowie ein Bienenstock und diverse Feinde der Biene aufgemalt. In der einfachsten Variante soll der Beebot vom Bienenstock aus eine von der Lehrperson vorgegebene Reihenfolge von Blumen sammeln. Felder mit Feinden muss die Biene dabei meiden. In der mittleren Variante soll die Biene zwei Blumen besuchen, die im Sommer blühen. In der schwierigsten Variante zeigt ein Aufgabenkärtchen eine vorgegebene Befehlszeichenfolge. Das Kind soll nur aus der Entschlüsselung dieser Befehlszeichenfolge aufzählen, welche Blumen die Biene besuchen wird. Die Selbstkontrolle erfolgt durch die Programmierung und die anschließende Fahrt der Biene.

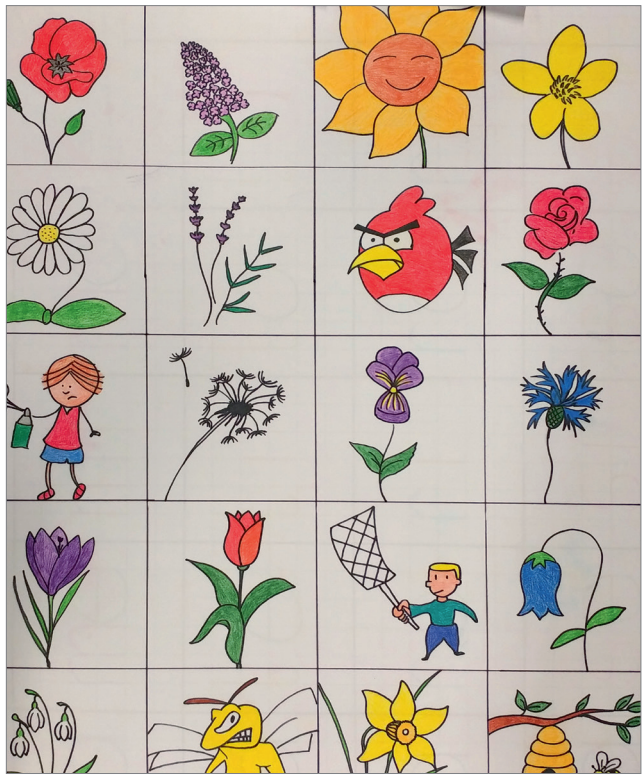
Musik:

Instrumente: Klanginstrumente liegen in der Mitte eines Sitzkreises. Daneben sind auf einem Flipchart-Raster mit vier mal vier Feldern diese Instrumente gezeichnet. Einem Kind werden die Augen verbunden. Ein zweites Kind betätigt eines der Instrumente. Danach wird dem ersten Kind die Augenbinde abgenommen, und es muss die Biene zu jenem Instrument schicken, das es gehört hat. Varianten: Es werden mehrere Instrumente hintereinander betätigt. Ein Kind schreibt eine Befehlszeichenfolge auf, die zu einem bestimmten Instrument führt. Ein anderes Kind erkennt aus dieser Zeichenfolge, bei welchem Instrument die Biene ankommen wird und betätigt daraufhin dieses Instrument. Kontrolle: die Zeichenfolge wird eingegeben und die Biene wird losgeschickt.

Alle entstandenen Lernszenarien können sowohl in der Gesamtgruppe, in Kleingruppen – etwa im Stationenbetrieb – und als Einzelübung zur Individualisierung eingesetzt werden. Alle Szenarien sind mit mehreren einfachen Aufgabevarianten versehen und zusätzlich erweiterbar, also eignen sich auch zur Differenzierung nach Schwierigkeitsgrad und Umfang.

Selbsteinschätzung der Studierenden bezüglich ihrer Kompetenzen im Bereich Coding und Robotik

In einer kurzen Fragebogenerhebung unmittelbar im Anschluss an die letzte Einheit der Lehrveranstaltung gaben 73,3% der Studierenden (n=43) an, vor Beginn der Lehrveranstaltung Informatische Bildung im Bereich Robotik und Coding "völlig unwissend" gewesen bzw.



Szenario Blumen

Szenario Reimwörter

zwar bereits vom Thema gehört, aber noch nicht damit in Berührung gekommen zu sein. 86,6% der Befragten gaben an, dass ihnen die Auseinandersetzung mit Beebots "ausgesprochen gut" bzw. "sehr gut" geholfen habe, die Grundgedanken des Codings zu verstehen. Die Frage, ob ihnen die Auseinandersetzung mit Beebots die Scheu vor dem Thema Coding genommen habe, beantworteten 73,4 % der Befragten mit "ausgesprochen gut" bzw. "sehr gut".

In der sehr offen gestellten und frei zu beantwortenden Abschlussfrage der Erhebung äußerte sich kein/e einzige/r der Studierenden negativ gegenüber den Bodenrobotern. Die überwiegende Anzahl der Aussagen unterstreicht das deutlich positive Ergebnis der quantitativen Erhebung. Ein kleiner Auszug:

Fragestellung: Durch die Auseinandersetzung mit Beebots habe ich...

- ... einen Einblick bekommen, wie man Unterricht interessanter und spielerischer gestalten kann. Natürlich nur für zwischendurch, aber den Kindern macht es sicher mehr Spaß, und sie lernen dadurch auch sicher leichter und lieber.
- ... gelernt, dass man auch auf recht einfache und spielerische Art und Weise die Thematik „Coding" behandeln kann.
- ... gelernt, wie wichtig der Umgang mit neuen Technologien bereits in der Volksschule ist. Ich habe erkannt, dass man sich als Lehrperson ruhig dem Thema annehmen kann und sich nicht davor fürchten muss.
- ... großes Interesse an der Arbeit im Bereich Coding und Robotik in der Schule entwickelt.
- ... erkannt, dass man keinen IQ von 150 braucht, um es zu verstehen und anzuwenden.

Fazit

Die Ergebnisse der Untersuchung zeigen deutlich, dass die Auseinandersetzung mit Beebots auch bisher wenig bzw. gar nicht affinen Studierenden einen einfachen Zugang zum Thema Robotik ermöglicht und Interesse für das Thema Coding erzeugt. Die Bereitschaft zu dieser Auseinandersetzung kann für Studierende des Lehramts Primarstufe durch die Planung eines fächerübergreifenden Einsatzes der Bodenroboter zusätzlich erhöht werden.

Autor

Mag. (FH) Walter Fikisz, BEd.
Ist Lehrender an der Pädagogischen Hochschule Niederösterreich in den Bereichen Informatische Bildung, Kommunikation, E-Learning und digitale Medien im Unterricht der Primarstufe.
Daneben arbeitet er als Referent und Co-Moderator für eLectures an der Virtuellen Pädagogischen Hochschule sowie als Tutor an der Katholischen Medien Akademie.

Autor

Mag. Josef Buchner
Lehrer für Geschichte, Psychologie, Philosophie und IKT. An der Pädagogischen Hochschule Niederösterreich verantwortlich für innovative Lehr- und Lernszenarien, Medienpädagogik und Mediendidaktik. E-Learning Koordinator am Bundeszentrum für lernende Schulen (ZLS), Referent und Online-Tutor an der Virtuellen PH sowie Mitbegründer des Netzwerks Flipped Classroom Austria.

Literatur:

Brandhofer, G. (2016). Coding, Robotik und Making in der Schule. Abgerufen von http://eis.ph-noe.ac.at/coding-robotik-und-making-in-der-schule/

Garcia-Penalvo, F., Rees, A. M., Hughes, J., Jormanainen, I., Toivonen, T., & Vermeersch, J. (2016). A survey of resources for introducing coding into schools. In Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16) (S. 19–26). Salamanca, Spain. Verfügbar unter: https://gredos.usal.es/jspui/bitstream/10366/131973/1/TACCLE-ResourcesCatalog-Preprint.pdf

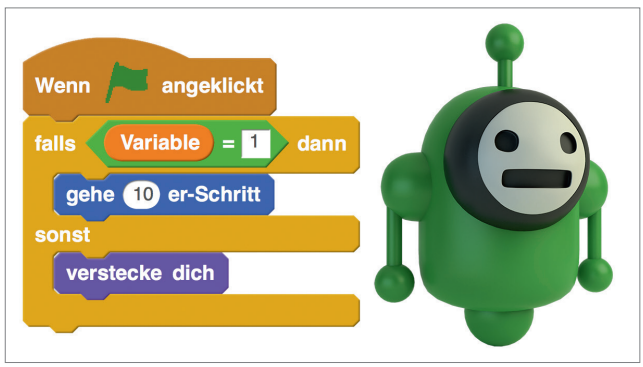
Stöckelmayr, K., Tesar, M., & Hofmann, A. (2011). Kindergarten Children Programming Robots: A First Attempt. In Proceedings of the 2nd International Conference on Robotics in Education (RiE 2011) (S. 185–192). Vienna, Austria. Verfügbar unter: https://pdfs.semanticscholar.org/ad7a/0219089052088d76089c5605fbd8fa8ee65e.pdf

Zorn, I., Stöckelmayr, K., Kohn, T., Derndorfer, C., & Trappe, C. (2013). Interessen und Kompetenzen fördern. Programmieren und kreatives Konstruieren. In M. Ebner & S. Schön (Hrsg.), L3T - Lehrbuch für Lernen und Lehren mit Technologien (2. Auflage). Verfügbar unter: http://www.pedocs.de/volltexte/2013/8371/pdf/L3T_2013_Zorn_et_al_Interessen_und_Kompetenzen.pdf

„Let's teach kids to code"

Tja, so ist das im Leben. Beim Wort „Scratch" denken fast alle Erwachsenen, insbesondere die Baby-Boomer, an den Discjockey, der die schwarze Vinylscheibe mal eben von 33 Umdrehungen pro Minute runterbremst, dann wieder auf den Singlespeed von 45 Runden beschleunigt, um in der nächsten Zehntelsekunde den Plattenteller in die Gegenrichtung zu bewegen. Durch dieses scratchen genannte Stilmittel gibt der DJ einem Hit seine ganz persönliche Note.

Mitch Resnick ist kein Discjockey, sondern Wissenschaftler. Man könnte auch sagen Wissenschaffer. Und er ist Erfinder von „Scratch". Nicht im musikalischen Sinne. Resnicks Scratch ist vielmehr eine Programmiersprache. Der Wissenschaftler, Jahrgang 1956, lehrt „Lernforschung" am berühmten Massachusetts Institute of Technology (MIT). Als Direktor der „Lifelong Kindergarten Group" des MIT-Media Lab gilt sein besonderes Interesse dem Einsatz von Computern im Unterricht. Vor zehn Jahren entwickelte er mit seinem Team die Programmiersprache Scratch. Wie geschaffen für Grundschüler, die nicht einfach nur im Internet rumsurfen, sondern eigene Programme schreiben und anwenden wollen.



Mit Scratch und Raspbotics sind Lernerfolg und Spaß vorprogrammiert.

Schon Achtjährige können mühelos „scratchen"

Auf Scratch baut auch das Raspbotics System auf. Damit lässt sich Coding hervorragend umsetzten. Für den Einstieg in die Welt der Programme – hier meinen wir ausdrücklich nicht ORF, KiKa und Toggolino – und insbesondere für Kinder eignet sich Scratch. Schon Achtjährige schaffen es mühelos, Codeblöcke zusammen zu stecken - ähnlich wie bei einem Puzzle. So lassen sich mit wenigen Mausklicks intuitiv eigene Abfragen, Wiederholschleifen und einfache Anwendungsszenarien programmieren - und das ohne jegliche Vorkenntnisse.

Dabei steht fest, dass Coding schon im Volksschulalter erlernt werden kann und sollte. Es ist für den

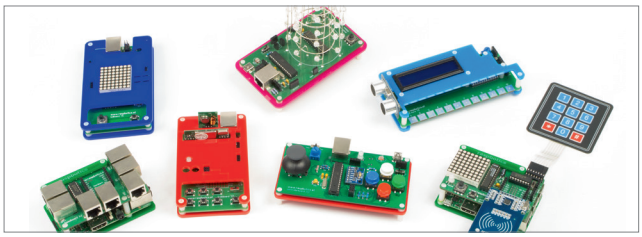
Technische Neugier und kreative Projekte sind Voraussetzungen für bestmögliche Resultate.

Nachwuchs ähnlich zu erlernen wie Lesen, Rechnen und Schreiben. Das Erkennen und Formulieren eines Gesamtproblems, dessen Zerlegen in Teilaufgaben sowie die zielorientierte Recherche sind Fähigkeiten, die während der Ausbildung und im Berufsleben immer wieder gefordert werden. Das Testen von Lösungswegen, die durchaus auch mal als Sackgasse enden, erhöht die Frustrationsgrenze, das heißt bei Fehlschlägen werden Wege raus aus der Sackgasse gesucht.

Momentan ist das Angebot an Lernrobotern, Entwicklungsboards, Minicomputern und ähnlichem ziemlich umfangreich. Man muss technisch versiert sein, um die Vor- und Nachteile, die Stärken und Schwächen der unterschiedlichen Systeme zu erkennen. Zudem muss auch der Preis eines Systems im Verhältnis zu dessen Möglichkeiten stimmig sein. Bei der Wahl des richtigen Systems stoßen also nicht nur Pädagogen schnell an ihre Grenzen.

Der Einfachheit halber wird dann oft das bekannteste System ausgewählt oder eines, das einem von irgendjemand empfohlen wurde, der vielleicht ganz andere Ansprüche hat. Das führt häufig dazu, dass ein teuer gekauftes System nach ein paar Stunden Einsatz in einer Ecke liegt und nicht mehr verwendet wird. Ähnlich dem Lieblingsspielzeug, das Weihnachten unterm Tannenbaum lag und spätestens an Silvester nicht mehr zündet.

Das wird bei dem Codingsystem Raspbotics mit großer Wahrscheinlichkeit nicht passieren. Denn mittlerweile geht die dritte Generation an den Start, die Weiterentwicklung bringt noch mehr Vorteile. Bei Raspbotics handelt es sich um Lernboards, die durch die Umsetzung kreativer Ideen eine Vielzahl an Aufgabenstellungen zulassen. Die Anwendungsbereiche der Boards sind in Kategorien eingeteilt, sodass alle Vorlieben bzw. jeder Bedarf abgedeckt wird:



Produktpalette Raspbotics Lernboards für Einsteiger und Fortgeschrittene.

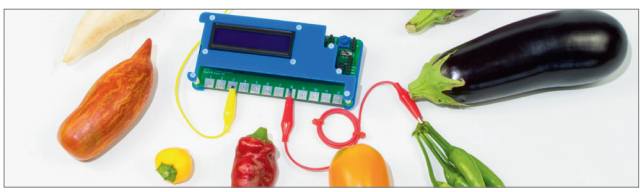
Lernboards für eine Vielzahl an Aufgabenstellungen

Mit dem Controllerboard lassen sich durch die verbauten Taster, Joystick und einer Vielzahl an Sensoren interaktive Steuerungen vornehmen, was sich ideal für die Spieleentwicklung anbietet. Das LED-Board holt den User die Retrozeit zurück, in der das Handydisplay noch aus wenigen Pixeln bestand. Das Programmieren von Snake oder das Zeichnen mit dem verbauten Joystick holt zumindest den Lehrer zurück in die gute alte Zeit.

Mit dem Smartboard zum Smarthome – Programmieranfängern geht ein Licht auf, denn sie können problemlos Steckdosen und Lampen per Funk ansteuern und die ersten Schritte zum intelligenten Wohnen unternehmen. Und wenn es nach dem Öffnen der Kühlschranktür dunkel bleibt, sollte man sein Programm noch mal überarbeiten. Mit dem Touch&more Board werden Alltagsdinge wie Obst, Wassergläser, metallische Gegenstände, leitende Farbe usw. zu Tastern, um Steuerungsaufgaben durchzuführen, mit Obst zu musizieren und vieles mehr. Ein verbautes Display sorgt für das Anzeigen wichtiger Ereignisse.

Schließlich bietet das Raspbotics-System auch handwerkliche Herausforderungen. Beim Pipeboard müssen SchülerInnen eine LED-Röhre aus Leuchtdioden löten und gewinnen dadurch Geschicklichkeit mit dem Lötkolben. Selbstverständlich kann auch diese LED-Pipe, wie alle anderen Boards, programmiert werden. Über Taster, Joystick und Sensoren können individuelle Muster ausgegeben werden.

Fortgeschrittene SchülerInnen können ihr Programmiergeschick am Pythonboard unter Beweis stellen. Es ist mit zahlreichen Sensoren und Modulen vollgepackt und bietet einen nahezu unbegrenzten Programmierspaß mit Python.



Mit dem Touch&more Board kommen auch Vegetarier auf ihre Kosten.

Raspberry Pi - das Herz von Raspbotics

Das Herz von Raspbotics ist ein circa 40 Euro teurer Raspberry Pi. Die Hardware ist kindersicher, Kurzschlüsse und falsches Anstecken sind nicht möglich. Dank zahlreicher Anwendungsbeispiele können auch unerfahrene Lehrer sofort loslegen und mit dem Coding starten.

Für das Aufsatteln von Scratch (für Schüler ab acht Jahren) zu C bzw. Python für fortgeschrittene Anwender ab ca.14 Jahren muss kein neues System angeschafft werden. Die Umstellung fällt sehr leicht, da die erlernte Programmierdenkweise schon geläufig ist und nur noch die Syntax der neuen Programmiersprache gelernt werden muss, ähnlich wie das Vokabellernen einer Fremdsprache.

Und während Erwachsene noch in der Bedienungsanleitung lesen, nehmen die Kids den Joystick in die Hand, probieren rum bis was klappt und leben ihre oft vorhandene Kreativität aus – ganz ohne Vorgaben. Wenn mal ein Experiment nicht hinhaut, hat vielleicht ein Mitschüler eine Lösung parat. Auf diese Weise wird die Teamfähigkeit gestärkt – eine auch im Berufsleben durchaus brauchbare Eigenschaft.

Vom Lehrer für Lehrer

Als Lehrer und Entwickler der Raspbotics Boards sehe ich mich nicht als der typische Hersteller, sondern versuche mit Boards zu unterschiedlichsten Themengebieten den Bedarf von LehrerInnen und SchülerInnen abzudecken. Ähnlich sieht es mit dem Thema Support aus. Einerseits sind zahlreiche Hilfestellungen und Einführungsbeispiele zu Raspbotics online verfügbar, andererseits stehe ich bei anfänglichen Fragen gerne als Ansprechpartner zur Verfügung. Weiters gibt es die Möglichkeit, im Rahmen eines Workshops dieses Lernsystem kennen zu lernen, um für die erste Schulstunde perfekt vorbereitet zu sein.

Link: www.raspbotics.at

Autor

Claus Zöchling
46 Jahre, Lehrer an der PTS18.
Aktuell: Abhaltung von Workshops (Programmieren von Mikrocontrollern, Raspberry Pi). Momentan berufsbegleitendes Studium am Technikum Wien in Embedded Systems.

Ziel: Mit Raspbotics vielen Kindern und Jugendlichen einen raschen und spannenden Einstieg in die Welt der Technik und des Programmierens zu ermöglichen.
E-Mail: zoechling@raspbotics.at

Digitale Kompetenzen für den Bereich Messen-Steuern-Regeln

„Do it yourself" ist das Motto der Maker Szene, in welcher im Zeitalter von Industrie 4.0 neben dem handwerklichen Geschick auch das Verständnis für Sensoren, Motoren und aktuelle Technik aus unserer Umwelt und dem Alltag ausgetauscht und vermittelt werden. Die Auseinandersetzung mit dieser Thematik in der Schule ist für unsere Jugend von hoher Bedeutung. Kritische Stimmen meinen, dass durch den Einsatz neuer Medien im Unterricht Technik vor der Pädagogik stehe – dem muss nicht so sein!

Computer übernehmen in unserer heutigen Welt immer häufiger einfache und komplexe Steuerungsaufgaben. Deshalb ist es zunehmend unverzichtbar, dass Schülerinnen und Schüler Grundkenntnisse in diesem Bereich erwerben. Für das lebensbegleitende Lernen ist der Erwerb von digitalen Kompetenzen im Schulalltag der Sekundarstufe I nicht nur dringend erforderlich, sondern auch in den österreichischen Lehrplänen schon lange festgeschrieben. Verschiedene Offensiven tragen dazu bei, dass diese verbindlichen Vorgaben von der Grundschule bis zur Sekundarstufe 2 verlässlich und praktisch umgesetzt werden können. Als Querschnittsmaterie sollen digitale und informatische Kompetenzen integrativ in allen Fächern vermittelt werden, was wiederum auch im Aus, Fort- und Weiterbildungsangebot der Pädagoginnen und Pädagogen berücksichtigt werden muss. Das Projekt digi.komp-MSR bietet sowohl Pädagoginnen und Pädagogen als auch Schülerinnen und Schülern einen Zugang in die digitale Grundbildung im Kompetenzbereich Messen-Steuern-Regeln und Robotik.

Bei der Anwendung von digi.komp-MSR Unterrichtsideen werden das Verstehen und Lösen unterschiedlicher Problemstellungen aus Schule und Alltag gefördert. Mit einem Blick auf Alltag und Umwelt werden aktuelle Themen, wie z. B. das Smart Home oder Industrie 4.0, aufgegriffen und bearbeitet. Diese Innovationen und „Erleichterungen" in der Arbeitswelt und dem Alltag haben auch Anforderungen an uns Menschen. Um dem gerecht zu werden, steht ein nachhaltiger Erwerb von Fertigkeiten und Fähigkeiten im Umgang und ein solides Basiswissen außer Diskussion. Die methodisch und didaktisch aufbereitete digi.komp-MSR Experimentierbox schafft einen Zugang dazu. Kinder und Jugendliche lernen nachhaltig, wenn sie selbst etwas schaffen oder zusammenbauen. Für Pädagoginnen und Pädagogen soll dies einerseits ein Auftrag für die Wissensvermittlung im Unterricht sein, kann aber andererseits auch für die eigene digitale Bildung und Professionalisierung von Vorteil sein. Durch das eigene Tun und praktisches Arbeiten können wichtige Erfahrungen und Erkenntnisse gesammelt werden.

Abbildung 1: digi.komp-MSR

Digi.komp-MSR ist ein didaktisches Konzept und Angebot zur digitalen Grundbildung und Implementierung dieser Schlüsselkompetenzen in Schule und Unterricht. Das Projekt digi.komp-MSR befasst sich mit Kompetenzen rund um das Messen-Steuern-Regeln und um die Robotik. Dieser Themenbereich kann in unterschiedlichsten Fächern aufgegriffen werden und ist dementsprechend in den österreichischen Lehrplänen der Sekundarstufe I in verschiedensten Bereichen verankert.

Mit den bei digi.komp-MSR eingesetzten Werkzeugen kann auch „kreativ gestaltet" und „selber gemacht" werden. Making Aktivitäten verbinden Handwerk mit digitalen Technologien und Jugendliche zeigen in der Regel Begeisterung dafür. Der Schulalltag zeigt jedoch immer wieder, dass für die inhaltliche Erarbeitung und Umsetzung im Zuge des sehr knapp bemessenen Informatikunterrichts alleine Zeit und Organisation fehlen. Aber im Fächerkanon sind die Zugänge zu informatischen Kompetenzen noch ausbaufähig. Der Themenbereich Messen-Steuern-Regeln erscheint vielen Kolleginnen und Kollegen im ersten Moment als Randbereich, doch spätestens nachdem ein Bewusstsein dafür geschaffen wurde, zeigt sich dieser Kompetenzbereich als sehr praxisnahe, wichtige Basis, experimentelle und handlungsorientierte Möglichkeit, digitale Kompetenzen im Unterricht zu implementieren. Vom algorithmischen Denken bis hin zur kreativen Schaffenskraft, ob fächerübergreifend oder in Projektform, dem nachhaltigen Lernen sind keine Grenzen gesetzt.

Schülerinnen und Schüler werden mit und für Technik begeistert. Gerade die Robotik bietet quer über die Schullaufbahn einen vielseitigen und handlungsorientierten Zugang zu komplexen Aufgabenstellungen und Themenbereichen. Das Kennenlernen und Anwenden unterschiedlichster Hardware sorgt für einen breitgefächerten Zugang. Hohe Kosten und fehlende Erfahrungswerte verhindern hier oft den ersten Schritt in die Umsetzung im Klassenzimmer. Das Konzept von digi.komp-MSR basiert auf Produkten aus dem Alltag und offenen Standards, damit keine Abhängigkeit von Herstellern und keine Vergänglichkeit für das didaktische Konzept entstehen können.

Syntax basierte Programmiersprachen und aufwendige Schaltungen erschweren oft den Einsatz und sind nicht immer einladend für Neueinsteigerinnen und Neueinsteiger. Sie sind fehleranfällig und rauben schnell die Motivation. Die gesamte Umsetzung scheitert oft an Kleinigkeiten, wie z. B. Beistrichfehler im Programmcode oder Wackelkontakte bei der elektronischen Schaltung. Ein fertiges und erprobtes Konzept mit visuellen Programmiersprachen für Kinder, wie z.B. Scratch, offenen Hardwareplattformen wie dem Arduino-Board und selbst gebauten Steckmodulen bilden die Basis der Experimentierbox des Projektes digi.komp-MSR. Dieses didaktische Konzept erleichtert den Kompetenzerwerb im Bereich Messen, Steuern und Regeln. Dadurch ist ein Zugang geschaffen, der nicht nur auf Pädagoginnen und Pädagogen aus dem Informatiksektor beschränkt ist, sondern auch alle daran Interessierten und neugierige Kolleginnen und Kollegen zur Mitarbeit einlädt. Der Aneignung und Vertiefung von Grundkompetenzen sind keine Grenzen gesetzt.

Durch einen umfangreichen Einsatz digitaler Medien kann Unterricht so gestaltet werden, dass die Schülerinnen und Schüler beim Erwerb ihrer digitalen Kompetenzen bestmöglich unterstützt werden. Sie sollen die Individualisierung und Personalisierung von Lernprozessen unterstützen und Unterricht für heterogene Lerngruppen anregen. Jedes Kind kann sein Potential optimal entwickeln. Im Zeitalter von Industrie 4.0 ist der Erwerb digitaler Kompetenzen für das Leben in einer von digitalen Medien und Geräten durchdrungenen Gesellschaft unabdingbar.

Die digi.komp-MSR Experimentierbox

Ein in der Schulpraxis bewährtes Konzept für die didaktische Aufbereitung von Elektronikthemen sind Experimentierboxen. Elektronik Baukästen gibt es auf dem Markt in verschiedensten Ausführungen und oft auch mit Begleitmaterial für Pädagoginnen und Pädagogen. Diese oft nicht pädagogisch aufbereiteten Begleitmaterialien sind qualitativ sehr unterschiedlich. Die Idee von digi.komp-MSR ist, dass sich Pädagoginnen und Pädagogen die Experimentierbox im Zuge einer modularen Fortbildungsserie selber bauen. Nachdem erstes Basiswissen für die Welt der Leuchtdioden, Widerstände und Sensoren erworben wurde, werden, angeleitet von Expertinnen und Experten, Elektronik Bauteile für das Messen, Steuern und Regeln auf Platinen gelötet. Die Funktion jedes Modules für das Arduino-Uno-Board wird am Ende überprüft. Anschließend werden Unterrichtsbeispiele in Scratch for Arduino programmiert. Während dieser Fortbildung können auch der Praxistransfer und die didaktische Umsetzung reflektiert und diskutiert werden.

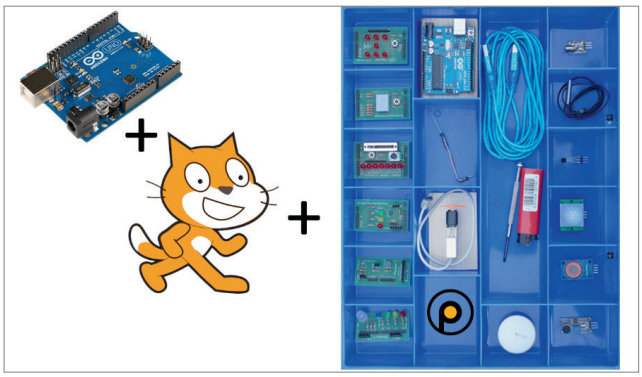


Abbildung 2: digi.komp-MSR Werkzeuge

Steckmodule versus Breadboard

Die Steckmodule können professionell vorgefertigt oder mit einer Lochrasterplatine aufgebaut werden. Der Vorteil dieser Methode ist, dass alle elektronischen Bauteile verlötet werden und somit fix miteinander verbunden sind. Bei der Erarbeitung von elektronischen Schaltungen können für den Erstaufbau auch sogenannte Breadboards verwendet werden. Wenn ausschließlich Breadboards verwendet werden und mehrere Gruppen parallel mit denselben Materialien arbeiten, müssen die Schaltungen immer wieder neu zusammengestellt werden. Neben der Fehleranfälligkeit von Kabeln und Wackelkontakten geht auch wertvolle Zeit für die inhaltliche Erarbeitung verloren.

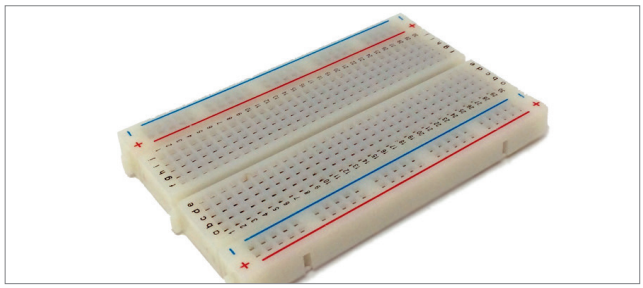


Abbildung 3: Breadboard

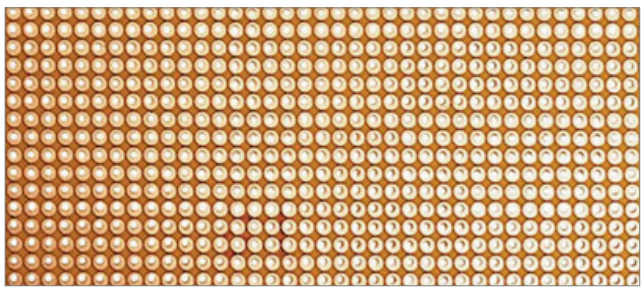


Abbildung 4: Lochrasterplatine

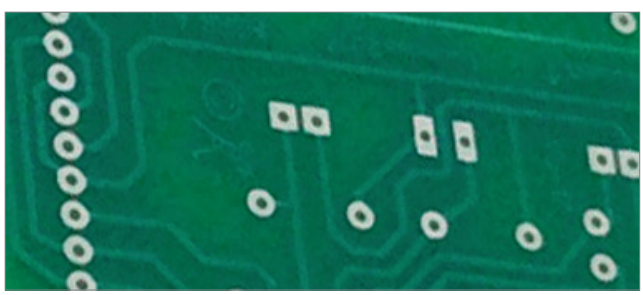


Abbildung 5: Professionell gefertigte Platine

Die inhaltliche Zusammenstellung der digi.komp-MSR Module lädt zum fächerübergreifenden Unterricht ein. Der Gestaltung von Modulen sind keine Grenzen gesetzt, jedoch für den Einstieg und die Fortbildung gibt es vorgegebene Module.

Die Experimentierbox kann derzeit mit folgenden Modulen bestückt werden:

- Steuern 1
- Messen (mit zusätzlichen Sensoren wie Flammen-, Gas-, Bewegungs-, Ultraschallsensor, etc.)
- Thermischer Regelkreislauf
- Motoren Steuerung
- Sieben-Segment Anzeige
- Elektronischer Würfel

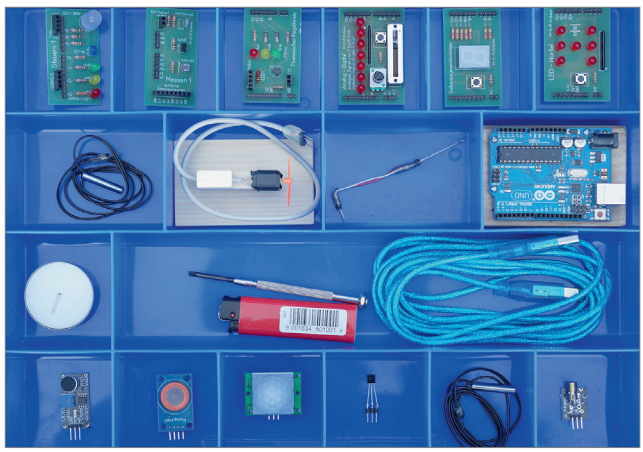


Abbildung 6: digi.komp-MSR Experimentierbox

Bei der Anwendung der Experimentierbox können mittels einfacher Experimente der Unterschied zwischen Analog und Digital aufgezeigt, ein Grundverständnis über verschiedene Sensoren gewonnen und erste Steuerungsaufgaben umgesetzt werden.

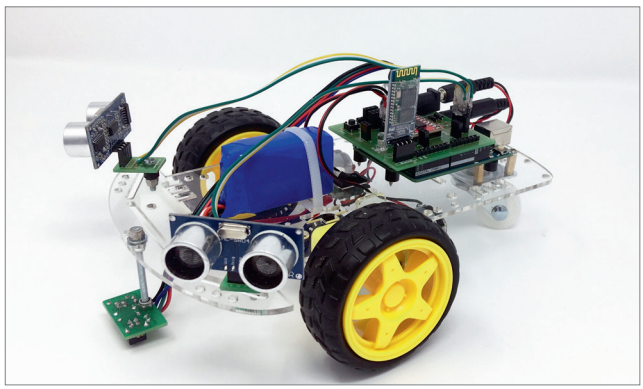


Abbildung 7: MSR-Fahrzeug

Im Rahmen der modular aufgebauten digi.komp-MSR Fortbildungsreihe werden Experimentierboxen für das Arduino-UNO-Board von den Pädagoginnen und Pädagogen selbst gefertigt. Da diese Experimentierboxen im Handel nicht erwerbbar sind, gilt es zu bedenken, dass die gewünschte Anzahl, idealerweise in Klassenstärke, herzustellen ist. Dazu können unterschiedliche Wege gewählt werden. Die einzelnen Module müssen nicht auf einmal für die Box produziert, sondern sie können auch über ein Schuljahr gefertigt werden. Sowohl Lehrerinnen und Lehrer als auch Schülerinnen und Schüler können im Team produzieren. Die digi.komp-MSR Experimentierbox ist ein zeitgemäßes und universell einsetzbares Equipment und kann über längere Zeit im Unterricht genutzt werden. Für den Aufbau werden handelsübliche Bauteile verwendet, das bedeutet, dass die unabhängige Versorgung von Ersatzteilen und Nachschub sichergestellt und nicht an einen Hersteller gebunden ist. Ist das Equipment einmal zusammengestellt, steht einem flexiblen, differenzierten und fachübergreifenden Einsatz im Unterricht nichts mehr im Wege. Bei der Anwendung auf Schülerinnen- und Schülerebene geht es nicht nur um die Technik und die Anwendung einzelner Bauteile, sondern auch um logisches Denken und das Fördern der Problemlösekompetenz. Vernetzter Unterricht wird angeregt. Eine inhaltliche Vertiefung oder die Weiterführung an höheren Schulstufen bietet sich an. Die digi.komp-MSR Experimentierbox dient dem Erwerb der Grundkompetenzen im Bereich Messen, Steuern und Regeln. Darauf Aufbauend kann das Meisterstück, ein sensorgesteuertes Arduino-Fahrzeug, umgesetzt werden. Inhaltlich bietet dieses Fahrzeug interessierten oder auch hochbegabten Schülerinnen und Schülern eine Vertiefungsmöglichkeit, es ist aber auch eine Brücke für die Weiterführung in höheren Schulstufen. Das MSR-Fahrzeug kann einer Linie auf dem Boden folgen, sich mittels Ultraschallsensoren gesteuert frei im Raum bewegen oder via Bluetooth über eine App vom Smartphone gesteuert werden. Die Umsetzung dieses Fahrzeuges in der Altersgruppe der 10 bis 14-Jährigen ist allerdings herausfordernd.

Das Arduino-UNO-Board

Arduino ist eine quelloffene aus Soft- und Hardware bestehende Plattform mit einem Mikrocontroller und analogen und digitalen Ein- und Ausgängen. Durch die unkomplizierte Handhabung des Arduino-Uno-Boards ist diese Open Source Plattform besonders für den Einsatz in der Schule geeignet. Dieser echte Mikrocontroller ist nicht nur in der Maker-Szene beliebt, sondern wird auch von unterschiedlichsten Firmen für Prozesse und spezielle Aufgaben adaptiert und verbaut. Je nach Hersteller und Bezugsquelle gibt es eine große Preisspanne vom Arduino-UNO-Board bis hin zum Arduino-kompatiblen-Board. Optisch sehen die Boards alle sehr ähnlich aus und für die Verwendung mit der digi.komp-MSR Experimenteierbox sollten alle kompatibel sein. Wenn Projekte inhaltlich über das schulische Know-how hinausgehen, kann auf eine weltweite Community zurückgegriffen werden. Je nach Programmiersoftware, kann das Arduino-Uno-Board entweder mit USB am PC angeschlossen oder autark mit einer 9V Batterie betrieben werden.

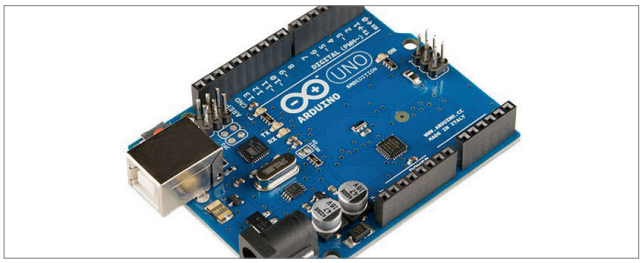


Abbildung 8: Arduino-Uno-Board

Coding mit Scratch & Co

Die Programmierung erfolgt in einer C/C++ ähnlichen Programmiersprache. Da wie schon erwähnt der Einsatz von Syntax Programmiersprachen im Unterricht oft ein schwer überwindbares Hindernis darstellt ist die Verwendung von visuellen Programmiersprachen empfohlen. In Bezug auf die digi.komp-MSR Experimentierbox gibt es derzeit zwei Ansätze.

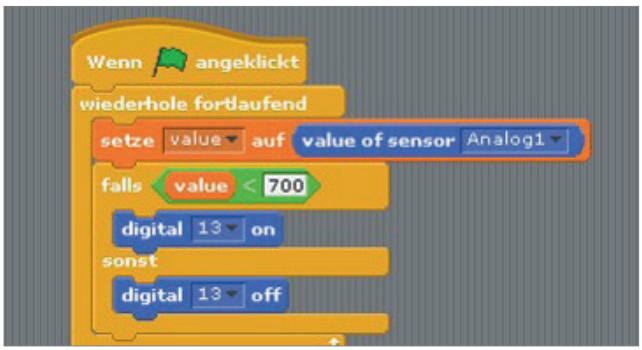


Abbildung 9: Scratch for Arduino Code

Scratch als visuelle und vor allem kindgerechte Programmiersprache schafft einen quasi barrierefreien Zugang nicht nur für Schülerinnen und Schüler. Die Anwenderinnen und Anwender lernen beim Programmieren mit Scratch kreativ zu denken und noch weitere grundlegende Fähigkeiten für das Leben im 21. Jahrhundert. Durch die Hilfestellung grafisch mit Blöcken programmieren zu können, kann das Abstrahieren ohne viel Einarbeitungszeit und somit das Programmieren leicht umgesetzt werden. Blöcke können nur aneinandergefügt werden, wenn sie logisch zusammenpassen. Im Vergleich zu syntaxbasierten Programmiersprachen entstehen hier keine Syntaxfehler z.B. durch einen vergessenen Beistrich im Programmiercode. Eine speziell für das Arduino-UNO-Board adaptierte Version von Scratch mit dem Namen „Scratch4Arduino" ermöglicht die direkte Verbindung zwischen Skript und Microcontroller. Alle Vorteile von Scratch können nun in der realen Welt des Physical Computings genutzt werden. Diese Programmiersprache ist aber nicht kompilierbar. Zum Steuern und Auslesen der analogen und digitalen Ein- und Ausgänge müssen Hard- und Software, also Microcontroller und Computer, dauerhaft mit einer Kabelverbindung verbunden sein. Zusammengefasst können ohne großen Aufwand mit Schülerinnen und Schüler ab der 5. Schulstufe Experimente im Bereich Messen-Steuern-Regeln durchgeführt werden. Erfahrungsgemäß sind die Kinder von dieser Art des handlungsorientierten, kreativen und motivierenden digitalen Kompetenzerwerbs begeistert.

```
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the Uno and
  Leonardo, it is attached to digital pin 13. If you're unsure what
  pin the on-board LED is connected to on your Arduino model, check
  the documentation at http://www.arduino.cc

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

Abbildung 10: Arduino Code

Mit fortlaufendem Schwierigkeitsgrad und steigender inhaltlicher Komplexität von Projekten bedarf es auch eines kabellosen Agierens mit Physical Computing Plattformen. Auch für die Verwendung der Module aus der digi.komp-MSR Experimentierbox und dem Arduino-Uno-Board, ohne stetige Verbindung mit dem Computer, gibt es einen Weg. Ein Erweiterungspaket für das Programm Arduino namens Ardublock ermöglicht visuelles Programmieren, wobei hier das Skript mit Mausklick in die C/C++ ähnliche Programmiersprache umgewandelt, somit kompiliert und auch direkt auf den Microcontroller hochgeladen werden kann. Das Arduino-UNO-Board kann nun auch nur mit Strom von z.B. einer Batterie versorgt, ohne Computer, den Programmcode abarbeiten.

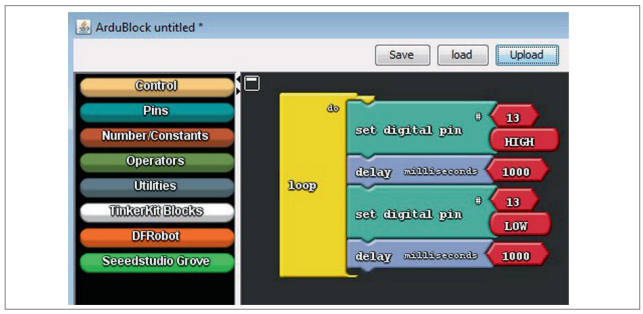


Abbildung 11: Ardublock Code

Das Programmieren zieht inzwischen weite Kreise und ist nun nicht mehr den Softwareentwicklern vorbehalten. Coding fördert das logisches Denken und Problemlösen. Mit dem richtigen und vor allem kindgerechten Zugang, können schon im frühen Kindesalter wertvolle Lernprozesse für das Leben von heute und morgen stattfinden.

Fortbildungsmöglichkeiten

Ohne die kontinuierliche Professionsentwicklung von Pädagoginnen und Pädagogen kann auch die Fülle an Angeboten für einen systematischen Aufbau der digitalen Kompetenzen keine Wirkung erzielen. Das Institut für Digitale Kompetenz und Medienpädagogik der Pädagogischen Hochschule Steiermark steht als Ansprechpartner und Wegbegleiter zur Seite. Neben den Fortbildungsangeboten im Süd-Ost-Verbund gibt es auch österreichweit erreichbare Formate wie eLectures und vor Ort Präsentationen.

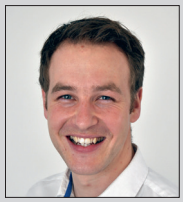
Ich lade neben den Kolleginnen und Kollegen der MINT Fächer alle Interessierten herzlich zu einem Informations-Webinar ein. Unser Ziel ist es, Lehrende mit diesem Konzept zur Vermittlung von Grundkenntnissen im Bereich Messen-Steuern-Regeln und Robotik zu motivieren und sie bestmöglich dabei zu unterstützen aktive Lernprozesse zu gestalten.

Wir freuen uns auf einen Austausch mit interessierten Kolleginnen und Kollegen und wünschen frohes Schaffen!

Weitere Informationen zum Projekt finden Sie unter https://msr.phst.at

Autor

Harald Meyer, MA BEd.
ist am Institut für Digitale Kompetenz und Medienpädagogik an der Pädagogischen Hochschule Steiermark tätig.
E-Mail: harald.meyer@phst.at



Ein offenes Unterrichtskonzept für den Einstieg in die Programmierung mit Hilfe von „Pocket Code“

“Und sie bewegt sich doch!“ – gemeint ist in diesem Fall nicht die Erde, sondern die Schulinformatik, welche sich, langsam aber stetig, in die richtige Richtung bewegt. Anstöße dafür gab es in den letzten Jahren genug. So wurden nicht nur visuelle Entwicklungsumgebungen für die einfache „kinderleichte“ Erstellung von eigenen Anwendungen geschaffen, sondern auch die Entwicklung von programmierbaren und leistbaren Robotern aller Art vorangetrieben, um Schülerinnen und Schülern einen Rollenwechsel zwischen der Rolle der Benutzerin und des Benutzers und der Rolle der Entwicklerin und des Entwicklers zu ermöglichen. Coding und Robotik gewinnen immer mehr an Bedeutung und finden daher auch Eingang in den Lehrplanentwurf für die verbindliche Übung „Digitale Grundbildung“ in der Sekundarstufe 1. Demnach soll „mit Algorithmen“ gearbeitet, „einfache Programme“ erstellt und die „kreative Nutzung von Programmiersprachen“ forciert werden (BMB, 2017).

Lehrerinnen und Lehrer stehen nun vor der Auswahl einer geeigneten Programmiersprache und Entwicklungsumgebung. Danach müssen noch geeignete Lehr- und Lernunterlagen gefunden oder erstellt werden. Eine Möglichkeit stellt der frei zugängliche und exportierbare Moodle-Kurs „Programmieren mit Pocket Code“ dar, der eine praktische Antwort auf die folgende Frage liefert: “Wie kann ein mögliches offenes Unterrichtskonzept für den Einsatz der mobilen App „Pocket Code“ aussehen, welches Schülerinnen und Schüler der Sekundarstufe 1 erste Erfahrungen mit dem Programmieren sammeln lässt?“ (Höllerbauer, 2017). Der Kurs wurde im Rahmen einer Diplomarbeit auf Grundlage von bestehenden Materialien des MOOC „Learning to Code – Programmieren mit Pocket Code“, welcher auf der MOOC-Plattform iMooX.at verfügbar ist und in der Ausgabe von Oktober 2016 vorgestellt wurde, entwickelt (vgl. Janisch et al, 2016).

Die App „Pocket Code“

Pocket Code ist eine im Zuge des Non-Profit-Projektes „Catrobat“ an der Technischen Universität Graz entwickelte mobile App, mit der Programme am Smartphone oder Tablet erstellt werden können (vgl. Slany, 2014). Catrobat bezeichnet auch die gleichnamige blockbasierte Programmiersprache, welche nach dem Vorbild von Scratch entwickelt wurde. Pocket Code erlaubt es den Schülerinnen und Schülern im Gegensatz zu Scratch, jederzeit und jederorts ihre eigenen Smartphones zum Programmieren zu verwenden. Mithilfe der Befehlsbausteine, welche unterschiedlichen Kategorien wie Steuerung, Ereignisse oder Bewegung zugeordnet sind und sich je nach Kategorie farblich voneinander abheben, können einfache Projekte schnell umgesetzt werden. Erfolgserlebnisse sind daher gleich zu Beginn möglich. Nach Seymour Papert, Begründer des Konstruktionismus (Schön et al, 2014) und Entwickler von LOGO, eine der ersten Programmiersprachen für Kinder, sollte sich

eine Programmiersprache durch die folgenden beiden Eigenschaften auszeichnen: Einerseits muss ein einfacher Einstieg in die Programmierung ermöglicht werden („low floor"), andererseits müssen auch komplexe und umfangreiche Projekte umgesetzt werden können („high ceiling"). Mitchel Resnick, ein Schüler von Papert und an der Entwicklung von Scratch beteiligter Professor am MIT, erweiterte diese Definition um die Eigenschaft „wide walls". Demnach soll eine Programmiersprache die Umsetzung von möglichst vielen verschiedenen Projekten ermöglichen, um unterschiedlichen Interessen und Lernmethoden gerecht zu werden (vgl. Resnick et al, 2009). Auch Pocket Code folgt dem „low floor"-„high ceiling"-"wide walls"-Prinzip. Die von den Benutzerinnen und Benutzern erstellten Programme sind ein klarer Beweis dafür, dass Pocket Code von unterschiedlichen Zielgruppen genutzt wird und in allen Unterrichtsfächern, gemäß des im Lehrplan der Sekundarstufe 1 festgelegten fächerintegrativen und fächerübergreifenden Einsatzes von Informations- und Kommunikationstechnologien, verwendet werden kann. Von Storytelling über die Erstellung von Animationen und Computerspielen bis hin zur mathematischen Beweisführung – auf der Catrobat Education Site ist eine Reihe an Unterrichtsbeispielen für den Einsatz von Pocket Code zu finden (siehe weiterführende Links). Neben Pocket Code wird mit Pocket Paint eine weitere App zur Verfügung gestellt, mit der Objekte am Smartphone oder Tablet selbst gezeichnet und anschließend mit Pocket Code programmiert werden können. Beide Apps können über den Google Play Store auf Android-Geräten installiert werden. Wie ein offenes Lehr- und Lernsetting zur Einführung in die Programmierung für Schülerinnen und Schüler von etwa 10-14 Jahren gestaltet werden kann, zeigt der offene Moodle-Kurs „Programmieren mit Pocket Code", der über das Learning-Management-System „TeachCenter" der Technischen Universität Graz aufgerufen werden kann.

Moodle-Kurs „Programmieren mit Pocket Code" für das offene Unterrichtskonzept

Mit dem Ziel, ein offenes Unterrichtskonzept zu entwerfen, das Rücksicht auf das heterogene Vorwissen und das individuelle Lerntempo der Schülerinnen und Schüler nimmt, wurde ein Moodle-Kurs erstellt, der sich in einen Arbeitsplan und in 6 thematische Bereiche („Zoo", „Unendlichkeit", „Sensoren", „Kollisionen und Schwerkraft" und „Rover") gliedert (siehe Abbildung 1).

Jeder dieser Bereiche ist wiederum in mehrere Abschnitte (z. B. Abschnitt Z1 - Z6 im Bereich Zoo) unterteilt, welcher die in Abbildung 2 dargestellten Elemente „Arbeitsblatt", „Videos", „Material" und „Verwendete Bausteine" beinhaltet. Das in jedem Abschnitt verfügbare „Arbeitsblatt" enthält eine offene Aufgabenstellung, die

Abbildung 1: Aufbau des Moodle-Kurs „Programmieren mit Pocket Code"

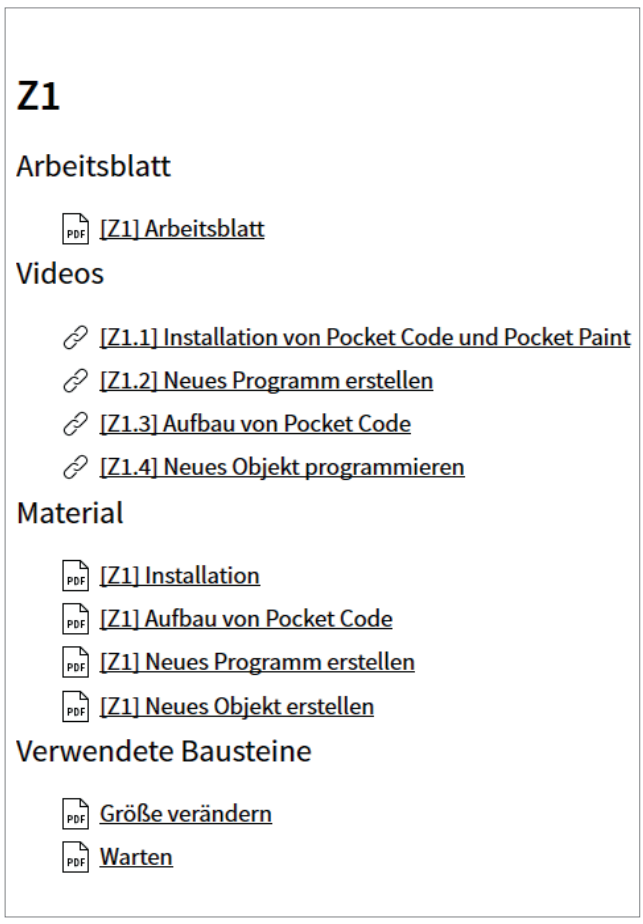


Abbildung 2: Jeder Bereich (z. B. [Z] Zoo) ist in mehrere Abschnitte (z. B. Z1 – Z6) unterteilt.

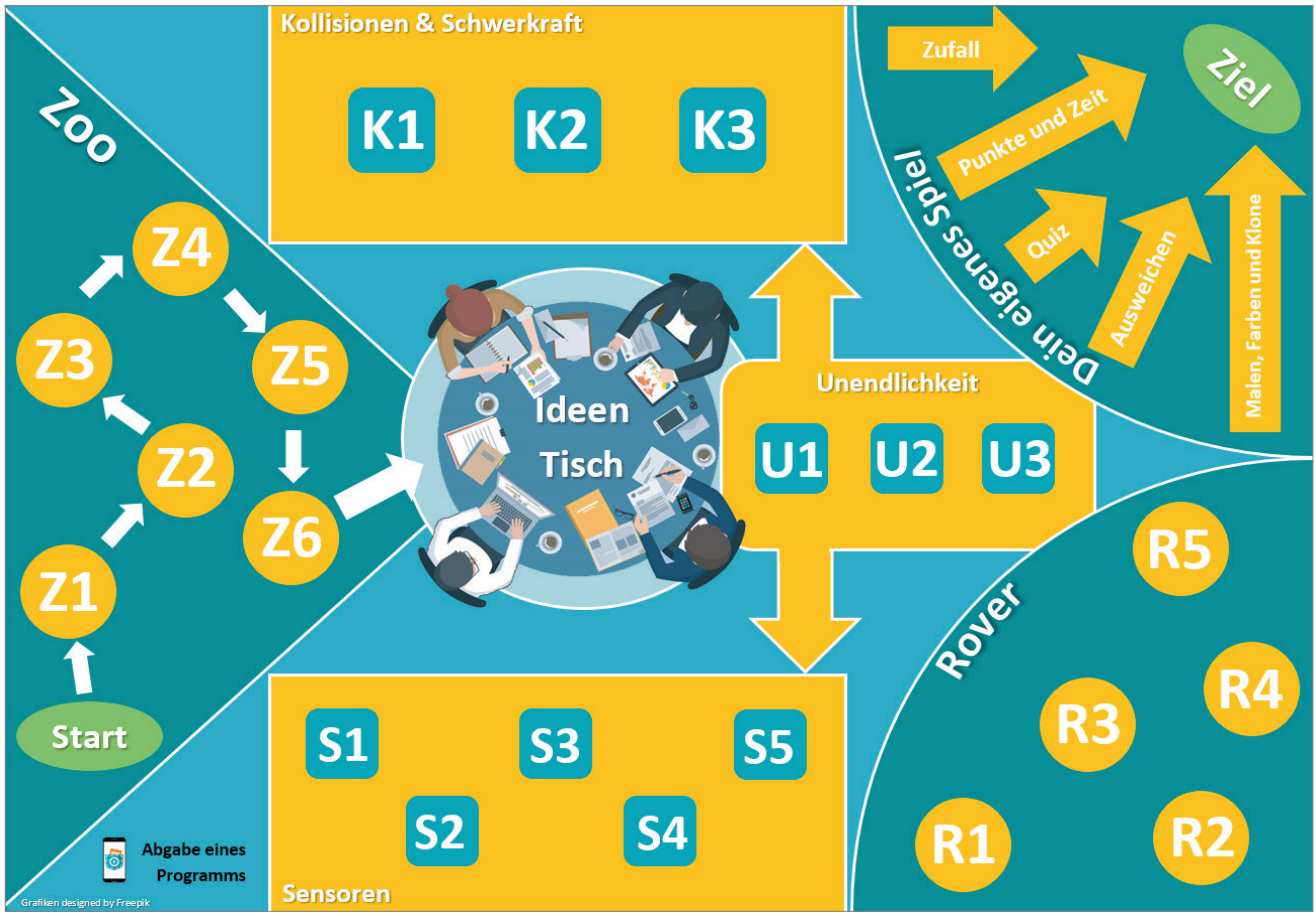


Abbildung 3: Der Arbeits- und Übersichtsplan mit definierten Start- und Endpunkt

mithilfe der Lernvideos oder der bereitgestellten PDF-Dateien gelöst werden kann. Unter „Material" sind neben text- und bildbasierte Erklärungen auch weitere Aufgabenstellungen zu finden. Zudem werden alle (neu) verwendeten Befehlsbausteine unter dem Punkt „Verwendete Bausteine" aufgelistet und beschrieben.

Der Arbeitsplan soll einen Überblick über die zu bearbeitenden Elemente der einzelnen Abschnitte geben und den Schülerinnen und Schülern als Leitfaden dienen (siehe Abbildung 3). Der Startpunkt befindet sich im Bereich „Zoo", wo die wichtigsten Begriffe und Grundlagen im Zusammenhang mit der App „Pocket Code" erklärt werden. Ziel des Kurses ist die Entwicklung eines eigenen Spieles. Den „Lernweg" von Start bis Ziel können die Schülerinnen und Schüler größtenteils selbst, gemäß ihren Vorkenntnissen und Interessen, bestimmen. Die in Abbildung 1 dargestellte Reihenfolge ist lediglich eine Empfehlung, da die Bereiche inhaltlich aufeinander aufbauen. Jedes Feld am Arbeitsplan (z.B. Z3, U1, S5, K2 oder R4) repräsentiert ein Arbeitsblatt, das von den Schülerinnen und Schülern bearbeitet werden kann. Im Bereich „Zoo" ist die Reihenfolge, in der die Arbeitsblätter behandelt werden sollen, vorgegeben. Alle anderen Aufgaben können frei gemäß dem offenen Unterrichtskonzept abgearbeitet werden. Da es sich zusätzlich um einen öffentlich zugänglichen Kurs handelt, ist die Bearbeitung sowohl im Unterricht als auch von zu Hause aus möglich. Am Arbeitsplan befindet sich auch der sogenannte „Ideen-Tisch". Dieser soll im Präsenzsetting als Anlaufstelle für verschiedene Fragen an die Lehrperson und als ein Ort zum Austausch von Ideen umgesetzt werden.

Rückmeldung des Lernfortschritts

Um den Lernfortschritt der Schülerinnen und Schüler besser verfolgen zu können, wird die Verwendung eines Punkte-Systems empfohlen. Auf jedem Arbeitsblatt ist in der linken unteren Ecke eine Punkteanzahl vermerkt (siehe Abbildung 4). Um diese Punkte zu erhalten, müssen die Schülerinnen und Schüler der Lehrperson das Arbeitsblatt zur Kontrolle vorlegen. In diesem Zusammenhang kann auch eine Mindestpunkteanzahl definiert werden, die von den Schülerinnen und Schülern am Ende erreicht werden soll. Ebenso ist eine Umsetzung als Gruppenarbeit möglich, bei der jedes Gruppenmitglied Punkte für die Gruppe sammelt. Das Punkte-System kann somit auch zur Motivierung beitragen.

Bereiche und Lernziele

In Abbildung 5 werden die für jeden Bereich definierten Lernziele angegeben. Im Bereich „Zoo" geht es darum,

die „Basics“ im Umgang mit Pocket Code zu erlernen. Der Bereich „Unendlichkeit“ behandelt Schleifenstrukturen und die Erstellung von einfachen Animationen. Im Bereich „Sensoren“ wird gezeigt, wie die einzelnen Sensoren des Smartphone oder Tablets in verschiedenen Kontexten, beispielsweise zur Steuerung eines Objektes im Rahmen eines Spieles, verwendet werden können. Der Bereich „Kollisionen und Schwerkraft“ umfasst die Erstellung von „physikalischen“ Objekten, die den Gesetzen der Physik, wie beispielsweise der Schwerkraft, folgen. Im Bereich „Rover“ wird die Kommunikation zwischen Objekten näher beleuchtet und Einsatzmöglichkeiten des kostengünstigen „Bigtrak rover“ von Zeon Tech vorgestellt (siehe weiterführende Links). Der Kurs endet mit dem Bereich „Dein eigenes Spiel“, wo die Umsetzung typischer Spielelemente, wie beispielsweise die Verwendung eines Punktezählers oder Countdown, besprochen wird.

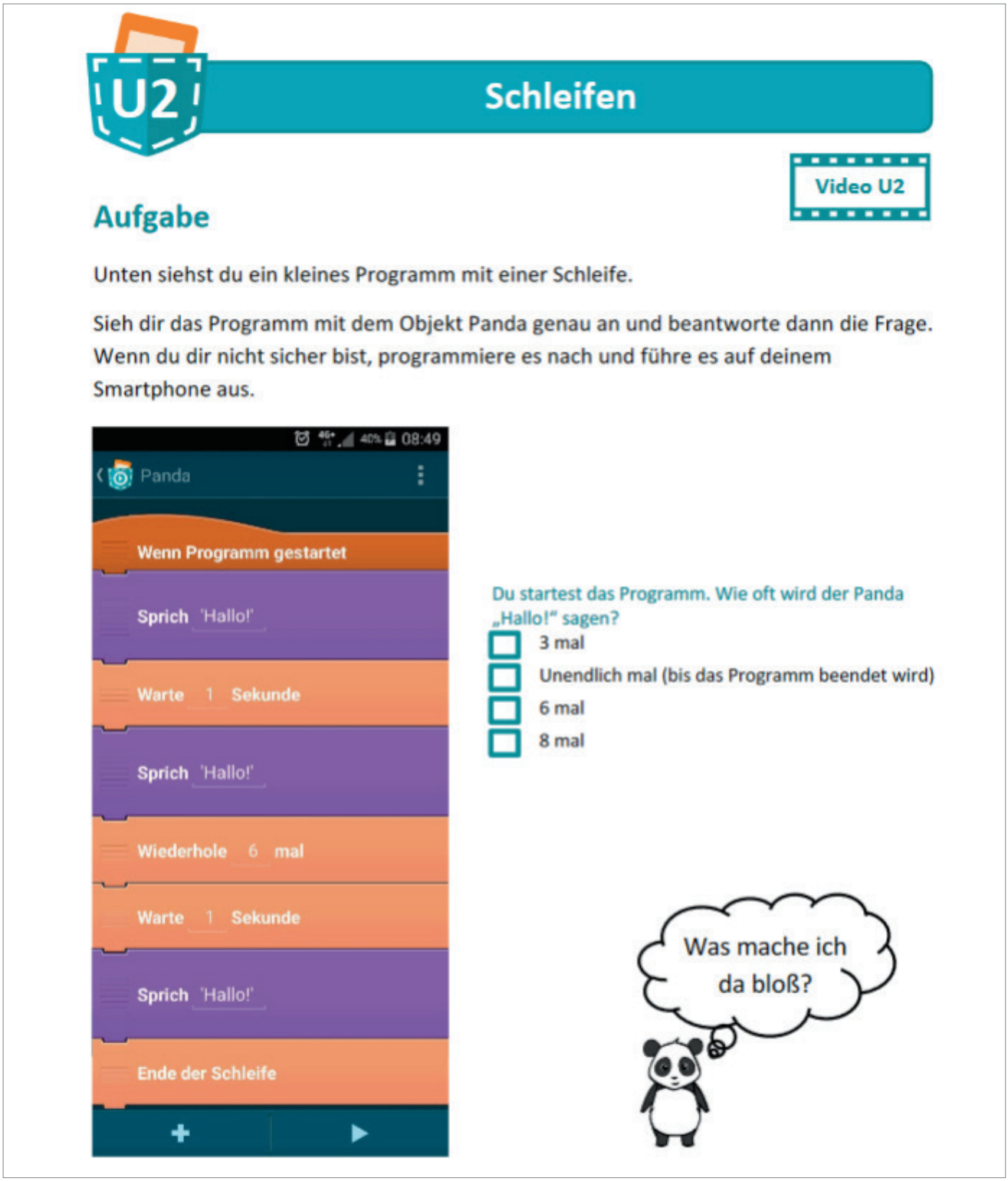


Abbildung 4: Beispiel für ein Arbeitsblatt mit 5 erreichbaren Punkten

Vorbereitungen für den Praxis-Einsatz

Lehrpersonen, welche den Kurs im Unterricht einsetzen möchten, können diesen in das schuleigene Learning-Management-System importieren oder den Kurs über das TU Graz TeachCenter aufrufen. Für das Arbeiten mit der App Pocket Code werden Smartphones oder Tablets mit einem Android-Betriebssystem benötigt. Zudem muss eine Möglichkeit für den Ausdruck der Arbeitsblätter gegeben sein. Zum Anschauen der Lernvideos sollten, zusätzlich zu den mobilen Android-Geräten, Notebooks oder Desktop-PCs und Ohrhörer zur Verfügung stehen. Sollte Interesse am Einsatz des Rovers bestehen, muss dieser rechtzeitig bestellt und auch das nötige Zubehör, wie beispielsweise Batterien, besorgt werden. Für den „Ideen-Tisch" muss in geeigneter Platz eingerichtet werden.

Erfahrungswerte

Die Durchführung des Kurses in einer 2. Klasse einer Neuen Mittelschule im Ausmaß von 12 Unterrichtseinheiten bewirkte eine „Verbesserung des logischen und analytischen Denkens" der Schülerinnen und Schüler (Höllerbauer, 2017). Die Schülerinnen und Schüler gaben an, dass die Aufgabenstellungen verständlich formuliert waren und dass sie mit dem Lehr-/Lernsetting gut zurechtkamen. Auch der Ideen-Tisch wurde von den Schülerinnen und Schülern öfters aufgesucht, um Hilfestellungen zu erhalten. Außerdem zeigte sich, dass im Laufe des Kurses nur mehr die zur Verfügung gestellten Lernvideos zum Lernen herangezogen wurden und auf das Lesen der text- und bildbasierten Beschreibungen verzichtet wurde (vgl. Höllerbauer, 2017).

Coding im Unterricht – Geht nicht gibt's nicht!

Der hier vorgestellte Kurs für ein offenes Unterrichtskonzept ist nur eine von vielen Möglichkeiten, um den Schülerinnen und Schülern grundlegende Programmierkenntnisse auf eine kreative, spannende und vor allem altersgerechte Weise zu vermitteln. Weitere Projektideen im Zusammenhang mit Pocket Code finden Sie auf der Catrobat Education Site oder auf der folgenden Website mit Themen zur informatischen Grundbildung: https://learninglab.tugraz.at/informatischegrundbildung/.

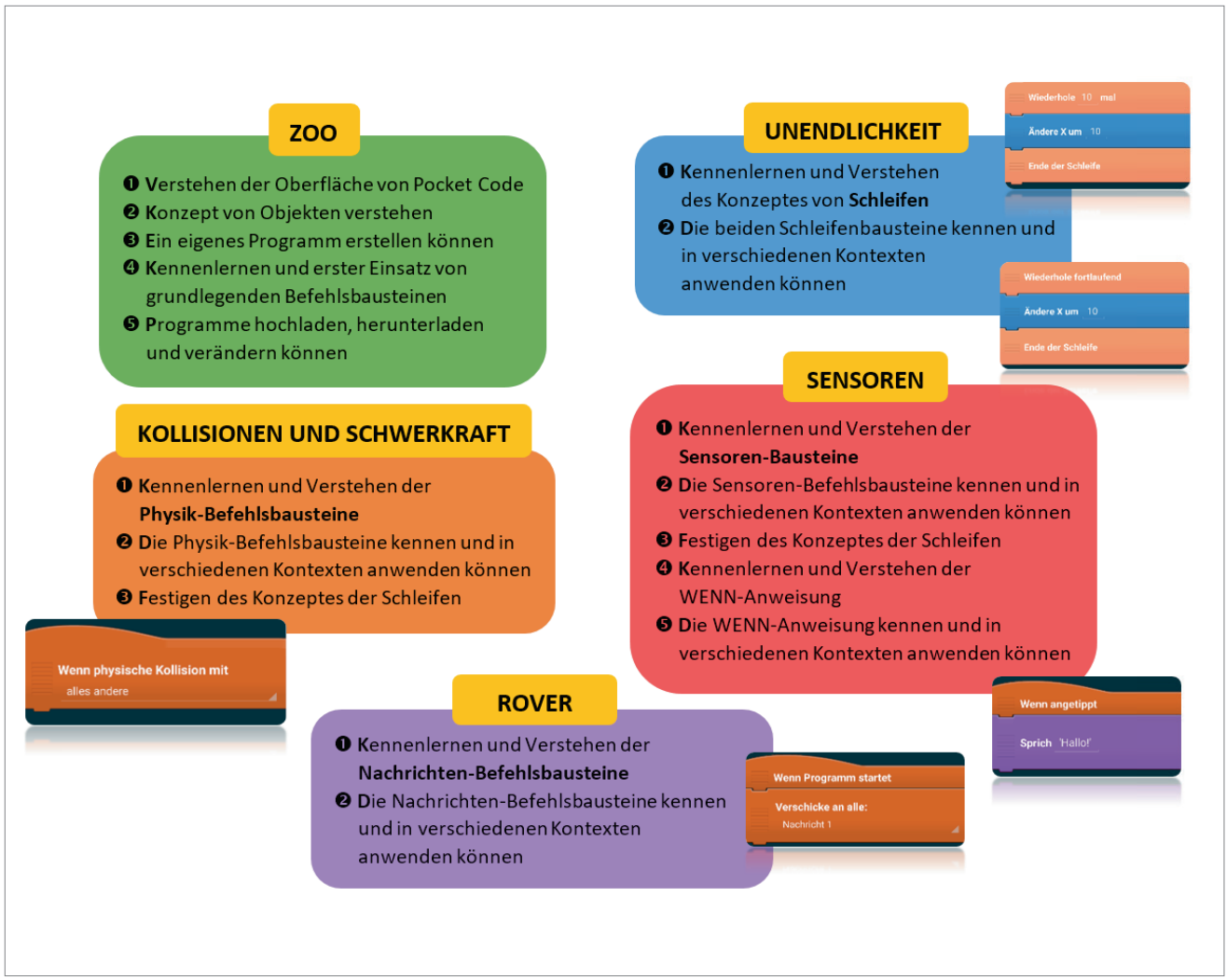


Abbildung 5: Definierte Lernziele (Höllerbauer, 2017)

Literatur:

Bundesministerium für Bildung (2017)
Österreichweite Aussendung an die Direktionen der allgemein bildenden höheren Schulen über die Pilotierung der verbindlichen Übung „Digitale Grundbildung“ im Schuljahr 2017/18.

Janisch, S., Ebner, M., Slany, W. (2016)
Pocket Code – freier Online Kurs für Kinder. In: SCHULE AKTIV, Sonderheft des BMB, Ausgabe Oktober 2016, S. 43-46. CDA Verlag.

Resnick, M., Maloney, J. et al. (2009)
Scratch: Programming for all. In: Communications of the ACM, Vol. 52, 11, S. 60-67.

Höllerbauer, B. (2017)
Schülerinnen und Schüler hacken: Der Einsatz von Pocket Code in einem offenen Unterrichtskonzept. Book on Demand, Norderstedt. http://o3r.eu

Schön, S., Ebner, M., Kurma, S. (2014)
The Maker Movement. Implications of new digital gadgets, fabrication tools and spaces for creative learning and teaching. In: eLearning Papers, 39, July 2014, pp.14-25., URL: http://www.openeducationeuropa.eu/en/article/Learning-in-cyber-physical-worlds_In-depth_39_2?paper=145315

Slany, W. (2014)
Pocket Code: a Scratch-like integrated development environment for your phone. In: Conference: the companion publication of the 2014 ACM SIGPLAN conference

Weiterführende Links:

MOOC “Learning to Code – Programmieren mit Pocket Code”
Start: Oktober 2017
https://imoox.at/wbtmaster/startseite/

Offener Moodle-Kurs “Programmieren mit Pocket Code”
(TU Graz TeachCenter)
https://tc.tugraz.at/main/course/view.php?id=1415

Band 14 der O3R-Reihe
„Schülerinnen und Schüler hacken: Der Einsatz von Pocket Code in einem offenen Unterrichtskonzept“
http://o3r.eu

Catrobat Education Site
Frei lizenzierte Lehr- und Lernunterlagen zum Arbeiten mit Pocket Code
https://edu.catrob.at/

Bigtrack Rover
http://www.rover.bigtrakxtr.co.uk/what.html

Aktivitäten der TU Graz zur Förderung informatischer Bildung
https://learninglab.tugraz.at/informatischegrundbildung/

Autor

Maria Grandl
Mag. Maria Grandl ist tätig am Institut für Interactive Systems and Data Science an der TU Graz und promoviert zum Thema informatorische Grundbildung. Im Rahmen ihres Lehramtsstudiums hat sie verschiedene Coding-Workshops mit Pocket Code abgehalten. Im Moment arbeitet sie u.a. an einem offenen Schulbuch für das Unterrichtsfach Informatik.

Autor

Bettina Höllerbauer
Mag. Bettina Höllerbauer absolvierte ihr Lehramtsstudium mit den Fächern Informatik und Mathematik an der TU Graz bzw. KFU Graz und entwickelte im Rahmen ihrer Diplomarbeit den im Artikel beschriebenen Moodle-Kurs.

Autor

Martin Ebner
Priv.-Doz. Dr. Martin Ebner leitet die Organisationseinheit Lehr- und Lerntechnologien an der Technischen Universität Graz und zeichnet für alle E-Learning-Belange der Universität verantwortlich. Als solches ist das Thema Making mit Kindern und Jugendlichen ein wichtiges Thema für ihn. Mehr können sie auf seinem Weblog nachlesen: http://elearningblog.tugraz.at

Autor

Wolfgang Slany
Univ.-Prof. Wolfgang Slany ist am Institut für Softwaretechnologie an der Technischen Universität Graz tätig und für die App „Pocket Code“ verantwortlich, welche im Rahmen des Catrobat-Projektes entwickelt wird. http://www.ist.tugraz.at/wolfgang_slany

Mobiles Klassenzimmer – Coding For Kids

Als führendes Technologieunternehmen will Samsung jungen Menschen dabei helfen, die digitalen Technologien der Gegenwart zu verstehen und damit die Zukunft selbst zu beeinflussen. In einer Roadshow von Mai bis Oktober ist das „Mobile Klassenzimmer" quer durch Österreich unterwegs. Kinder können dabei auf spielerische Art und Weise die Welt des Programmierens und der Robotik kennenlernen.

Software ist das Herz der Wirtschaft und durchdringt auch zunehmend unser aller Privatleben. Beim Mobilen Klassenzimmer lernen Schulkinder, sich kreativ und aktiv in den neuen Technologien auszudrücken, die unser aller Zukunft massiv beeinflussen werden. Konkret bringen ausgebildete IT-TrainerInnen den Kindern im Rahmen von Workshops das Programmieren von Handy-Apps und Robotern bei – und zwar anfängertauglich, mit garantiertem Riesenspaß für die Kinder, und völlig kostenlos. Inhaltlicher Partner von Samsung für Expertisen ist das Institut für Softwaretechnologie der TU Graz.

Didaktisches Konzept

Programmieren gehört zu den Kulturtechniken unserer Zeit. Das Mobile Klassenzimmer ermöglicht es Schulkindern, Produzenten eigener digitaler Inhalte und Medien zu werden. Programmieren fördert die Problemlösungskompetenz der Kinder und schärft ihr logisches Denken.

In den Workshops lernen Kinder, wie sie Spiele und andere Apps direkt auf ihren eigenen Handys selbst erstellen, Roboter steuern und abstrakte Befehle mit konkreten Ergebnissen verbinden können. Dabei verwenden sie eine intuitiv verständliche, Lego-artige visuelle Programmiersprache, die in Graz seit 2010 in Zusammenarbeit mit dem Scratch Team des MIT Media Labs sowie Freiwilligen aus 50 Ländern entwickelt und mit vielen Kindern speziell für kleine Handy-Bildschirme optimiert wurde: Pocket Code.

Die Gratis-App, die schon in über 40 Sprachen übersetzt wurde, wurde von Univ.-Prof. Dr. Dipl.-Ing. Wolfgang Slany und seinem Team am Institut für Softwaretechnologie der TU Graz mit dem Verein Catrobat entwickelt. Das Catrobat Projekt hat viele Erfolge vorzuweisen, unter anderem den österreichischen Staatspreis für Multimedia und e-Business im Bereich Innovation.

Das Mobile Klassenzimmer wird im Rahmen des EU Projekts „No one left behind" wissenschaftlich begleitet und danach so aufbereitet, dass das Modell unter Berücksichtigung der Erfahrungen von interessierten LehrerInnen aufgegriffen und unabhängig durchgeführt werden kann.

Bei den teilnehmenden Schülerinnen und Schülern kommen die Workshops sehr gut an. Für viele ist es

Abb.: Das Mobile Klassenzimmer soll auch im kommenden Schuljahr wieder durch Österreich touren

faszinierend, das Smartphone nicht nur zum Telefonieren oder Spielen zu nutzen, sondern selber einfache Spiele zu kreieren. Besonders viel Freude haben die Nachwuchs-EntwicklerInnen beim Programmieren der kleinen Roboter.

Auch Lehrerinnen und Lehrer begrüßen die Möglichkeiten, die die kostenlosen Workshops bieten. Im Unterricht bleibe kaum Zeit, um moderne Hilfsmittel einzusetzen und Dinge einfach auszuprobieren, so der Tenor. Die Workshops werden als guter Einstieg in die Welt des Programmierens begrüßt und das kreative, lustbetonte Lernen als gute Motivation für den Unterricht gelobt.

Als innovatives Technologie-Unternehmen investiert Samsung seit Jahren in die digitale Kompetenz der Kinder. Dabei wird größter Wert darauf gelegt, auch abseits der Ballungsräume digitale Fertigkeiten junger Menschen zu fördern, um der digitalen Kluft zwischen Stadt und Land entgegenzuwirken. Speziell auch Mädchen sollen frühzeitig motiviert werden, ihr Potenzial auszuschöpfen und ihr Selbstvertrauen im Umgang mit Technik zu stärken.

Rückblick

Wegen des hohen Zuspruchs und des regen Interesses der Schulen seit 2016 führt Samsung die erfolgreiche Workshop-Reihe auch im Jahr 2018 fort.

Insgesamt programmierten bisher mehr als 2.800 Kinder im Alter von 8 bis 14 Jahren bei den Coding for Kids Workshops. Das Mobile Klassenzimmer machte 36 Mal Station und ließ Schülerinnen und Schüler aus über 100 Schulen aller Bundesländer in einer kreativen Lernumgebung in die Welt des Programmierens eintauchen.

Damit setzt sich Samsung als Innovationstreiber für den sinnvollen und produktiven Umgang mit Technologie ein und bringt jungen Menschen jene Fertigkeiten näher, die sie in einer zunehmend digitalisierten Welt brauchen.

Links:

Projekt: http://coding.digitalebildung.at
Catrobat Projekt: http://www.catrobat.org/
Digitale Bildung Samsung: http://www.digitalebildung.at

Melden Sie Ihre Schule für die Tour 2018 bereits jetzt an unter digitalebildung@samsung.at!

Autor

Mag. Martina Friedl, MSc
Corporate Citizenship & Public Affairs Manager bei Samsung Electronics Austria GmbH; Jg. 1975, Mutter zweier digital Natives; tourt mit „Coding for Kids“ durch ganz Österreich, um Kindern Programmieren und Steuern von Robotern am Handy beizubringen; findet, dass Virtual Reality der neueste Trend in der Bildung werden sollte.

Acodemy – eine Programmierschule als Erfolgsstory

Digitale Medien verändern unsere Welt, sodass zeitgemäße Bildungs- und Arbeitsprozesse ohne die Nutzung digitaler Technologien und ohne entsprechende Kompetenzen nicht mehr möglich sind. Die Anfang 2017 von Bildungsministerin Hammerschmid präsentierte Digitalisierungsstrategie zeigt, wie aktuell und wichtig Themen wie Technologie, Programmieren und Problemlösen sind.

Wie dies in der Praxis funktioniert, zeigt seit 2016 das Start-Up Unternehmen acodemy (www.acodemy.at), gegründet von zwei Technikerinnen Anna Relle Stieger und Elisabeth Weißenböck. Die beiden Gründerinnen haben sich zum Ziel gesetzt, Kindern im Alter von 5-12 Jahren durch das Programmieren eigener Computerspiele/eigener Geschichten digitale Kompetenzen und informatisches Grundverständnis zu vermitteln.

1580 begeisterte Kinder, 1074 absolvierte Programmierstunden und 810 kleine Codingprojekte – das ist die Bilanz von einem Jahr acodemy!

Doch wie viel Arbeit dahintersteckt, mag man nur erahnen. Die zwei Gründerinnen des Unternehmens erzählen ihre Geschichte mit viel Begeisterung: „Wir haben uns vor einem knappen Jahr bei einer Tagung der Österreichischen Computer Gesellschaft zum Thema „Computational Thinking and Coding at Schools“ kennengelernt. Durch Zufall haben wir festgestellt, dass wir beide dieselbe Vision haben: eine österreichweite Programmierschule für Kinder und Jugendliche. Trotz einer nur sehr kurzen Kennenlernphase beschlossen wir, diese Idee gemeinsam in die Tat umzusetzen. Gerade durch unsere Diversität in Alter und Erfahrung erwies sich dieser mutige Schritt als der richtige.

Das Unterfangen sollte so schnell wie möglich in die Tat umgesetzt werden, da wir noch im gleichen Schuljahr Schulklassen kostenlose Programmierworkshops anbieten wollten!

„Dazu brauchten wir Flyer. Und für Flyer braucht man ein Logo. Für ein Logo braucht man allerdings einen Namen. Also entstand in einer langen Nacht mit Sekt und Brainstorming der Name „acodemy". Zusammengesetzt aus den Wörtern academy und coding.

Danach ging es in die Konzeptionierungsphase: Wie können Kinder und Jugendliche für das Programmieren begeistert werden? Indem man sie bei dem abholt, was sie sowieso gerne machen: Spiele-Apps und Computerspiele üben eine große Faszination auf Kinder aus. Wir greifen diese Faszination auf und lassen die Kinder selbst eigene Inhalte gestalten. Wichtig ist für uns dabei auch der Bezug zum Lehrplan der jeweiligen Altersgruppe: Ganz nebenbei werden Grundkenntnisse in Mathematik gefestigt, logisches und strukturiertes Denken geschult und natürlich erste Informatikkenntnisse und Problemlösungskompetenzen erlangt. Die altersgerechte Aufbereitung dieser Inhalte liegt uns dabei genauso sehr am Herzen wie die Freude am selbständigen Gestalten. Wir verwenden die beiden Programmiersprachen ScratchJr und Scratch, die sehr viel Gestaltungsspielraum lassen: So kann der Fantasie freien Lauf gelassen werden, von Feen, Drachen, Sportlern oder Fahrzeugen, die Spiele und Geschichten der Kinder sehen sehr unterschiedlich aus. Jedes Spiel/jede Geschichte hat einen individuellen Charakter, jedes Kind kann seine Vorlieben einbringen und hat dadurch einen persönlichen Bezug zur Thematik."

„Parallel zur Konzeptionierung ging es auf TrainerInnensuche, da aufgrund der Schulworkshops das Interesse an unseren laufenden Kursen immer größer wurde. Wir sind sehr stolz auf unsere TrainerInnen! Sie müssen kein Informatikstudium absolviert haben, uns ist es wichtig, dass sie informatische Konzepte kindgerecht weitergeben können (das lernen sie bei uns) und Erfahrung mit Kindern mitbringen.

Die erste Institution, bei der wir unterrichten durften, war ein Kindergarten. Auf 5-Jährige waren wir zu Beginn nicht vorbereitet, entwickelten also ein besonderes Konzept (Computational Thinking-Aufgaben ohne Computer, spielerisches Programmieren am Tablet mit ScratchJr, usw.). Die Freude der Kleinen war groß und 5-Jährige, die am Tablet eine Geschichte zusammenstellen und sich die Logik dahinter überlegen, sind etwas ganz Besonderes!

Im Laufe des Jahres kamen immer mehr Kooperationspartner dazu (öffentliche Schulen, Privatschulen, Kindergärten) die mit Begeisterung sehen, wie viel Spaß das Programmieren den Kindern macht und wie sinnvoll und notwendig es ist, digitale Kompetenzen zu erlernen.

Wir sind sehr glücklich darüber, auf welch reges Interesse unser Programmierunterricht bei Kindern, Eltern und PädagogInnen stößt und wie viel Freude und Wissen wir vermitteln können!

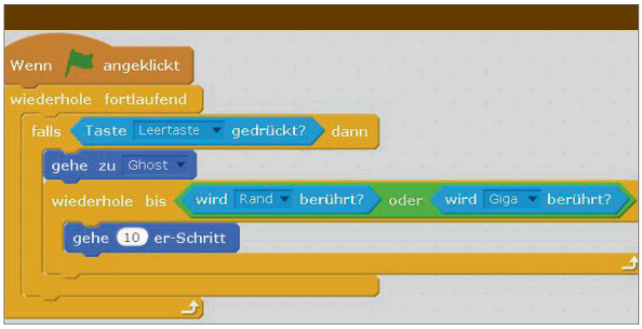
Praktisches Beispiel zur komplexen Problemlösung anhand von Scratch: Eine Figur soll einen Ball schießen.

Für Kinder sind gewisse Handlungen in Computerspielen/Apps selbstverständlich, wie etwa das Steuern von Figuren, Hüpfen usw. Dass dies zuvor programmiert werden muss, ist den Kindern oft nicht klar.

Ein Beispiel: Ich möchte, dass in meinem Computerspiel eine Figur einen Ball mehrmals werfen kann, wenn ich die Leertaste drücke. Ich weiß, wie die Lösung aussehen soll: Die Figur schafft es, einen Ball mehrmals zu schießen. Der Weg dorthin wird mittels Programmierung in Scratch gelöst.

Problem identifizieren:
Es gibt leider keinen Befehl, mit dem ich ganz einfach einen Ball schießen kann. Daher muss ich das Problem in kleinere Teile unterteilen:

Problem Nr. 1: Damit ich auch im realen Leben einen Ball schießen kann, muss ich ihn zuerst in meinen Händen halten. Sobald ich die Leertaste drücke, muss der Ball in die Hände meiner Figur gehen. Dies wird mit dem Befehl „gehe zu jener Figur, von der der Ball wegfliegen soll" gelöst.



Problem Nr. 2: Der Ball muss sich bewegen. Dazu gibt es mehrere Lösungswege:

1. Ich kann den Ball einfach so oft um ein Stück bewegen, bis er die gewünschte Stelle erreicht hat.

2. Ich bewege den Ball solange, bis er entweder den Rand der Bühne (sprich: er hat vorbeigeschossen) oder die andere Figur berührt (der Ball hat sie getroffen).

3. Der Kreativität sind keine Grenzen gesetzt, solange die Lösung sich mithilfe der Programmiersprache ausdrücken lässt!

Lösungsansatz bewerten und entscheiden: Ich entscheide mich für die zweite Variante, da ich nicht lange ausprobieren muss, wie weit der Ball fliegen soll. Die Weite des Fliegens ist an die Figur und an die Bühnenrand gekoppelt. Nun ist die Aufgabe gelöst, die Figur kann einen Ball schießen!

Kontakt und Information:

Acodemy sitzt im 3. Bezirk in Wien. Das Angebot besteht aus Nachmittagskursen, Programmierworkshops für Schulklassen, Kooperationen im Rahmen der Nachmittagsbetreuung, und vielem mehr. Wir sind auch für Kooperationen mit Schulen in anderen Bundesländern offen.

Näheres unter:
www.acodemy.at oder per E-Mail an info@acodemy.at.

Autor

Elisabeth Weißenböck
ist geschäftsführende Gesellschafterin der acodemy GmbH. Sie hat Informatikmanagement und Informatikdidaktik studiert und beschäftigt sich seit Jahren mit der Wissensvermittlung im Bereich der Informatik.

Autor

Anna Relle Stieger
ist geschäftsführende Gesellschafterin der acodemy GmbH. Sie hat Elektrotechnik und postgraduate Rechts-, Betriebs- und Wirtschaftswissenschaften studiert und hat über 20 Jahre Erfahrung im Management und Unternehmensaufbau.

Warum ich Python mag

Nein, das ist nicht der missionarischer Versuch, Python als die einzig seligmachende Programmiersprache anzubieten. Diskussionen über Programmiersprachen (vor allem über deren Einsatz im Unterricht) verlaufen oft wie religiöse Diskussionen: Nur die eigene Ansicht ist richtig, alle anderen müssen falsch sein (auch und vor allem), wenn man sich gar nicht ernsthaft damit beschäftigt hat. Die Geschichte ist voll von Beispielen dafür: von Echnaton, der den Monotheismus eingeführt hat, über den Dreißigjährigen Krieg bis zu den Auseinandersetzungen in der Gegenwart.

Nein, das wird hier kein neuer Kampf um Sprachen, sondern eine Einladung, etwas anderes anzusehen und sich eine eigene Meinung zu bilden. Ich bewundere die Entwickler von Python, die eine vollwertige und leicht zu lernende Sprache geschaffen haben - daher dieser Beitrag. Also, auf zu Neuem!

Python kann von der Webseite https://www.python.org/ für unterschiedliche Betriebssysteme kostenlos geladen werden. Derzeit (Juli 2017) sind die Versionen 2.7.13 und 3.6.2 aktuell. Ja, es gibt ein Python 2 und ein Python 3. Welche wählen wir? Python 3 ist sicher die zukunftsweisende Wahl. Wer aber ein ganz bestimmtes Programmpaket verwenden will, muss möglicherweise Python 2 einsetzen.

Python ist ein Interpreter. Damit eröffnet sich ein besonders einfacher Zugang zur Sprache. Befehle können immer gleich ausprobiert und umgesetzt werden. Wie wäre es, statt der üblichen „Hello World"-Programme einmal Python als Taschenrechner einzusetzen? Die grundlegenden Rechenoperationen sind rasch erklärt und können immer sofort ausprobiert werden:

```
>>> 25+38
63
```

Über das gedankliche Modell „Taschenrechner“ führt der Weg rasch zu Variablen.

Einige Beispiele sollen nun das Interesse wecken.

Zwei Zahlen werden eingelesen und deren Summe ausgegeben.

```
a = int(input('Bitte die Zahl a eingeben: '))
b = int(input('Bitte die Zahl b eingeben: '))
print (f'Die Summe von {a} und {b} ist {a+b}.')
```

Zum Vergleich dieselbe Aufgabe in Java:

```
import java.util.Scanner;
public class Sum {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("Bitte die Zahl a eingeben: ");
        int a = scan.nextInt();
        System.out.println("Bitte die Zahl b eingeben: ");
        int b = scan.nextInt();
        System.out.format("Die Summe von %d und %d
                                ist %d.", a, b, a+b);
    }
}
```

Welcher Aufwand ist wohl notwendig, um Begriffe wie import, public, class, static, void, main, String, args, System, out, format usw. zu erklären? Oder lautet das pädagogische Konzept hier „Schreibt es einmal ab, ich erkläre es später“?

Zurück zu Python: Elektrotechniker freuen sich darüber, dass das Rechnen mit komplexen Zahlen keine speziellen Bibliotheken erfordert, sondern Teil der Sprache ist.

Wer sich aber lieber der Sprache „grafisch“ nähern will, findet auch in Python ein Turtlegrafik-Paket. Mit den folgenden Zeilen entsteht ein hübscher Stern mit 36 Zacken.

```
import turtle
for i in range(36):
    turtle.forward(200)
    turtle.right(170)
```

Eine besondere Stärke von Python zeigt sich beim Umgang mit Zeichenketten, Tupeln und Listen: auch sind sie Teil der Sprache und können ohne Bibliotheksaufrufe verwendet werden. Die Vereinfachungen, die sich daraus ergeben, sind in der Tat dramatisch. Carl Friedrich Gauss wird eine originelle Lösung für die Berechnung der Summe der natürlichen Zahlen von 1 bis 100 nachgesagt. In Python wird daraus ein schmucker Einzeiler:

```
print(sum(range(1, 101)))
```

Selbstverständlich gibt es auch Klassen für die objektorientierte Programmierung. Da Python ja ein Interpreter ist, können Instanzen einer Klasse auch zur Laufzeit verändert werden.

Python ist als Sprache zum Einstieg in das Programmieren sehr gut geeignet. Das typische „Hello World“-Programm besteht wirklich nur aus einer Zeile. In Java oder C# wären mindestens sechs Zeilen notwendig und es wären Elemente enthalten, die zu Beginn der Programmierausbildung nicht im Details erklärt werden können.

Andererseits ist Python eine vollwertige Programmiersprache, mit der auch sehr umfangreiche Projekte realisiert werden können. Dazu kommen umfangreiche Programmbibliotheken, die für alle üblichen Standard-Aufgaben Programmpakete bereithalten: kein Wunder, verwendet doch Python einfach die C-Bibliotheken und stellt sie über eine Schnittstelle als Python-Funktionen zur Verfügung.

Ein Projekt, das vollständig in Python geschrieben ist, ist die Initiative „InfoSMS“ (http://www.infosms.org): Es geht um die Verbesserung der Kommunikation zwischen Schule und (vor allem) den Eltern mit elektronischen Mitteln. Schauen Sie sich doch die Webseite an. Wenn Sie Interesse an einer Probeinstallation haben, schreiben Sie mir eine E-Mail.

Python gewinnt aber auch bei maschinennaher Programmierung immer mehr an Boden:

micro:bit

Natürlich ist es sehr motivierend, auch einmal ein wenig Hardware anzusteuern – und wenn es für's erste nur ein paar LEDs sind, die im Rhythmus einer Verkehrsampel blinken. Der Raspberry Pi ist ein ausgezeichneter Kandidat für ein derartiges Vorhaben, vor allem, da es auch sehr gute Programmbibliotheken samt Aufbauanleitungen gibt. Wer aber schon einmal versucht hat, in einer Gruppe an zehn Computern mit zehn Steckbrettern ein paar LEDs samt Vorwiderständen korrekt mit den Ein- und Ausgängen eines Microcontrollers zu verbinden und auch das Betriebssystem samt Programmiersprachenineterpreter fehlerfrei zu bedienen, wird sich über eine Entwicklung der BBC freuen. Der micro:bit ist ein Einplatinen-Computer, der über einen Micro-USB-Anschluss mit einem PC (oder Laptop) verbunden wird. Die Programmierung ist sogar via Smartphone möglich. Der

micro:bit enthält nicht nur 25 LEDs zur Ausgabe, sondern auch Tasten und Sensoren zur Datenein- und -ausgabe, und kann über Krokodilklemmen oder eine Steckverbindung mit weiterer Peripherie verbunden werden.

Vom micro:bit wurden mehr als eine Million Exemplare kostenlos verteilt. Der Microcontroller ist auch im Handel erhältlich. Warum konnte so ein Computer für Lehrzwecke nicht in Österreich erfunden werden? Vielleicht weil die Erfindung von gemischten Bundes-Landes-Behörden („Bildungsdirektionen") bereits als großer Erfolg gefeiert wird?

Raspberry Pi

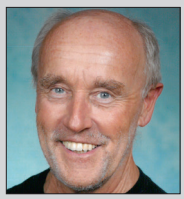
Wer schon die ersten Erfahrungen (Erfolgserlebnisse, aber auch Misserfolge) hinter sich gebracht hat, kann sich an Größeres wagen. Der Raspberry Pi (oder sein chinesischer Nachbau, der Orange Pi) können für anspruchsvolle Projekte eingesetzt werden. Und wieder bietet sich Python mit einer umfangreichen Programmbibliothek an.

Ja – soweit meine Einladung, sich doch einmal mit Python zu beschäftigen. An Stelle einer umfangreichen Literaturliste, die ja doch niemand anschaut, vier Hinweise bzw. Links zum Weiterlesen.

1. Statistik der Programmiersprachen: Der monatliche TIOBE-Index der Programmiersprachen: https://www.tiobe.com/tiobe-index/ nennt im Juli 2017 Python an der vierte Stelle: Java, C, C++, Python, C#, PHP...
2. Der PYPL http://pypl.github.io/PYPL.html reiht Python im Juli 2017 auf Stelle 2: Java, Python, PHP, C#, Javascript, ...
3. Ein etwas älterer Beitrag https://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-tea ching-language-at-top-u-s-universities/fulltext erklärt Python zu der am weitesten verbreitete Sprache in Einführungskursen an den wichtigsten Universitäten in den USA.
4. Und schließlich ist die Magisterarbeit über „Die Eignung von Python für Einführungskurse in das Programmieren im Vergleich zu anderen Programmiersprachen" von Yildiz Yazicioglu, http://katalog.ub.tuwien.ac.at/AC12689747 sehr lesenswert: Unter anderem werden an mehreren Beispielen Programmiersprachen verglichen.

Autor

Martin Weissenböck
Univ.-Lektor Dipl.-Ing. Mag. Dr.phil., Lehrer und ehem. Direktor der HTL Wien 3 Rennweg, Leiter der Arbeitsgemeinschaft für Didaktik, Informatik und Mikroeleketronik, Obmann des Vereins SCHUL.InfoSMS, Vorstandsmitglied der OCG, Autor von Büchern, Unterrichtsmitteln und Artikeln
E-Mail: martin@weissenboeck.at



Minecraft mit Python neu entdecken

Das Computerspiel Minecraft begeistert viele Kinder. Bei Minecraft geht es um das Bauen mit würfelförmigen Blöcken und das Erleben von Abenteuern. Statt nur zu spielen, eignet sich dieses Spiel wunderbar, die Programmiersprache Python zu erlernen. So wird Minecraft ein völlig neues Erlebnis. Ein Erfahrungsbericht.

Minecraft eröffnet viele Freiheiten und Möglichkeiten, das Spiel zu gestalten. Der Spieler besitzt von Spielbeginn an ungehinderte Bewegungsfreiheit und entscheidet selbst, was er wann und wie in der Spielewelt erkunden möchte. Bei Minecraft kann man die Spielewelt beliebig bearbeiten und gestalten. Die Spieler können Rohstoffe abbauen (mine) und diese zu anderen Gegenständen weiterverarbeiten (craft), so können komplexe Bauwerke entstehen. Minecraft ist deshalb so erfolgreich, weil es nicht nur ein Spiel ist, sondern eine kreative Plattform, mit der sich die verschiedensten Ideen umsetzen lassen.

Minecraft wurde in Schweden entwickelt und 2014 von Microsoft gekauft. Seitdem unternimmt der neue Eigentümer große Anstrengungen, um Minecraft an Schulen zu etablieren. Sichtbares Ergebnis dieser Bemühungen ist die „Minecraft Education Edition", eine spezielle Version des Spiels für den Einsatz im Unterricht. Für die Minecraft Education Edition wurde erst Ende Mai 2017 eine Erweiterung namens „Minecraft Code Builder" vorgestellt. Diese soll Schülern in spielerischer Weise den Zugang zur Programmierung eröffnen. Anfänger lernen mittels „Minecraft Code Builder" das Programmieren per Drag-and-Drop, Fortgeschrittene können auch in die Javascript-Sicht umschalten und textuell programmieren.

Alternative zur Minecraft Education Edition

Die Minecraft Education Edition und der Code Builder sind aber nicht Voraussetzung für die Minecraft-Program-

mierung. Die klassische Programmierung von Minecraft mit Python über eine Programmierschnittstelle (Application Programming Interface oder kurz API) ist nach wie vor populär. Die OCG hat schon mehrere Minecraft/Python Workshops abgehalten. Diese Workshops eignen sich hervorragend, um die Schüler von einem beliebten Spiel hin zum Erlernen einer vollwertigen Programmiersprache zu führen.

Wer auf traditionelle Weise eine Programmiersprache erlernt, muss sich erst mühsam in die Anwendungsbeispiele einarbeiten, um deren Zweck zu verstehen. Mit der Kombination von Minecraft und Python finden sich aber sofort praktische und anschauliche Einsatzmöglichkeiten Mit den so selbst erstellten Skripts kann man dieselben Tätigkeiten durchführen, die auch manuell möglich sind. Der Vorteil ist, dass die Durchführung mit Python weitaus schneller und weniger mühevoll ist. Man kann den Skripts bei der Arbeit zusehen und sieht die Ergebnisse der Skripts in Echtzeit in der Minecraft Welt, was für angehende Entwickler interessant ist, denn so kann man das dynamische Verhalten von Programmen beobachten. Das ist ein weiterer Vorteil der Kombination von Python und Minecraft. Bei unseren Workshops, die wir für Kinder ab 12 Jahren abhalten, zeigte sich, dass dieses Erfolgserlebnis die Freude am eigenen Programmieren sichtlich beflügelte.

Erfolge mit eigenen Skripts

Dass man mit einfachen Python-Skripts Aufgaben erledigen kann, die sonst viel Zeit und Mühe in Anspruch nehmen würden, ist für die Lernenden eine wichtige und unmittelbare Erfahrung. So erlernten die Teilnehmer bei der OCG eigene Skripts zu erstellen, die ein fertiges Haus in die Spielewelt platzieren oder eine lange Allee von Bäumen pflanzen. Wir erarbeiteten auch Skripts, die Blumen regnen lassen oder jeden Block, der betreten wurde, in Gold verwandeln.

Python ist modern, weit verbreitet, leicht zu erlernen und eignet sich deshalb hervorragend als Unterrichtssprache. Für den Workshop verwendeten wir das sehr empfehlenswerte Buch „Python programmieren lernen mit Minecraft“ als Lernunterlage. Das Buch enthält eine Vielzahl von Codebeispielen, die von den Teilnehmern ausprobiert, variiert und weiterentwickelt werden können. Das bringt schnellere Erfolgserlebnisse aber auch mitunter schnellere Lernerfolge, als wenn die Programme völlig neu erstellt werden müssten.

Bei den Workshops wurden Python-Grundkenntnisse über Variablen, Verzweigungen, Schleifen, Listen und Funktionen vermittelt. Codebeispiele wurden intensiv durchbesprochen und die Kursteilnehmer konnten in Kleingruppen selbst ausgedachte oder vorgeschlagene Projekte umsetzen. Ein von den Schülern umgesetztes Projekt erstellt in Minecraft eine alternative Uhr (vgl. Abb.2)

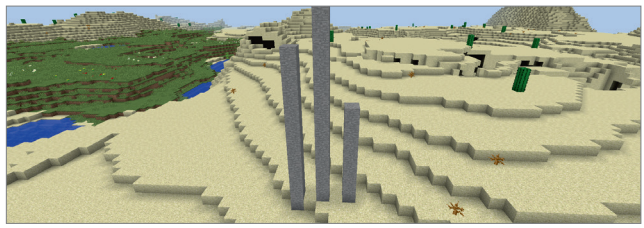


Abb.2: Die Uhrzeit wird vom Betriebssystem in einer Schleife abgefragt. Stunden, Minuten und Sekunden werden als unterschiedliche Säulen dargestellt. Das Skript wird so gestartet, dass es dauerhaft ausgeführt wird. So entsteht eine alternative Uhr in Minecraft, die aus drei Säulen besteht. An diesen drei sich ständig verändernden Säulen kann man die Uhrzeit ablesen.

Abb.1 zeigt ein von Minecraft mittels Skript erstelltes Haus. Leicht lässt sich mit dem Skript auch gleich eine ganze Stadt erstellen. Mittels Zufallsparameter könnte man die Häuser so variieren, dass kein Haus dieser Stadt einem anderen gleicht.

```
from mcpi.minecraft import Minecraft
import datetime
import time
mc = Minecraft.create()
pos = mc.player.getTilePos()
x = pos.x
y = pos.y
z = pos.z
height1 = mc.getHeight(x,z)
height2 = mc.getHeight(x+2,z)
height3 = mc.getHeight(x+4,z)
while True:                 #keep running until terminated
  t = datetime.datetime.now()
  m = t.minute
  h = t.hour
  s = t.second
  print(str(h) + ":" + str(m) + ":" + str(s))    #for debugging only
  if h == 0:                #reset hour column
    mc.setBlocks(x, height1, z, x, height1 + 12, z, 0) #blockid 0 is air
    print(".")
  elif m == 0:              #reset minute column
    mc.setBlocks(x+2, height2, z, x+2, height2 + 59, z, 0)
    print(".")
  elif s == 0:              #reset second column
    mc.setBlocks(x+4, height3, z, x+4, height3 + 59, z, 0)
    print(".")
  mc.setBlocks(x, height1, z, x, height1 + h, z, 1) #blockid 1 is stone
  mc.setBlocks(x+2, height2, z, x+2, height2 + m, z, 1)
  mc.setBlocks(x+4, height3, z, x+4, height3 + s, z, 1)
  time.sleep(0.8)
```

Skript zu Abb.2

Die Kursteilnehmer experimentierten auch mit folgendem aus dem Buch stammendem Skript. Es erstellt mittels einer Schleife eine Pyramide bei der Spielerposition (vgl. Abb.3). Die Kinder freuten sich über die Ersparnis an Zeit und Mühe. Solche unmittelbaren Erfolgserlebnisse sind motivierend und bestärken die Kinder, sich weiter mit Programmierung zu beschäftigen.

Abb.3: Die mittels Python programmierte Pyramide

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()


block = 24  # sandstone
height = 10
levels = reversed(range(height))

pos = mc.player.getTilePos()
x, y, z = pos.x + height, pos.y, pos.z

for level in levels:
    mc.setBlocks(x - level, y, z - level, x +
    level, y, z + level, block)
    y += 1
```

Skript zu Abb.3

Alle Schüler waren mit Begeisterung bei der Sache, denn sie konnten die neuerworbenen Python-Kenntnisse sofort praktisch erproben. Die Workshops waren auch für uns eine sehr positive Erfahrung, weil wir auf der Popularität von Minecraft aufsetzend einen sehr lebendigen Programmiersprachenunterricht gestalten konnten.

Buchtipp:

Craig Richardson, Python programmieren lernen mit Minecraft. dpunkt.Verlag. Juli 2016. 392 Seiten. 2017 ist eine unveränderte Neuauflage erschienen.

Inhaltsverzeichnis, Codebeispiele und Leseproben können gratis auf der Verlag-Website heruntergeladen werden.

Deutsch: https://www.dpunkt.de/buecher/12511/9783864903731-python-programmieren-lernen-mit-minecraft.html
Englisch: https://www.nostarch.com/programwithminecraft

Autor

DI Wilfried Baumann
Jahrgang 1966, hat an der TU Wien Informatik studiert und ist seit einigen Jahren für die Österreichische Computer Gesellschaft tätig, er gehört zum Team Forschung & Innovation. Baumann betreut u.a. gemeinsam mit Univ.-Prof. Gerald Futschek (TU Wien) auch den von der OCG durchgeführten „Biber der Informatik"-Wettbewerb und macht Coding-Workshops für Schüler aller Altersklassen. Zu seinen Hauptinteressen gehören Softwareentwicklung, technische und naturwissenschaftliche Themen, Informatik und Didaktik der Informatik.

Coding4you.at - die Coding-Plattform für alle

Coding4you.at ist eine von Netidee.at 2014 geförderte Plattform der Österreichischen Computer Gesellschaft (OCG), die in Österreich bereits bestehende Initiativen zum Thema Coding vernetzt. Eine kommentierte Sammlung von Online-Ressourcen zum Thema Coding komplementiert das Angebot. Einige diese Ressourcen für die Volksschule und die Sekundarstufe 1 werden hier beispielhaft wiedergegeben. Viele weitere Ressourcen können auf der Plattform aufgerufen werden.

1) Coding-Konzepte ohne Computer lernen WIZIK - Wiener Zauberschule der Informatik

Das Projekt WIZIK (Wiener Zauberschule der Informatik), das von der Wirtschaftsagentur Wien unterstützt wird, bringt Kindern die Informatik auf spielerische Weise näher und gibt erste Einblicke in die Informationstechnologie. Dazu werden bereits existierende österreichische Methoden, wie z.B. aus dem „Biber der Informatik"-Wettbewerb, herangezogen und daraus werden optimale Aufgabenstellungen für Kinder im Volksschulalter entwickelt.
https://www.ocg.at/de/wizik-wiener-zauberschule-der-informatik

2) Einstieg in Coding Code.org

Auf Code.org befinden sich einige Online-Kurse, die es Lernenden, Lehrenden und Codern ermöglichen, programmieren zu lernen.
https://code.org/

3) Coding-Apps für den Einstieg Lightbot

Lightbot™ ist ein englisches Spiel (auch für Handys), das für Kinder und Jugendliche entwickelt wurde, um ihnen einfaches logisches Verständnis beizubringen. Die Spielenden lernen je nach Schwierigkeitsgrad des erreichten Levels unterschiedliche Aufgaben zu bewältigen und bauen somit ein Gefühl für Logik auf.

Es gibt auch eine Version, die für Kleinkinder zwischen 4 und 8 Jahren geeignet ist.
http://lightbot.com/

4) Visuelle Programmiersprachen ScratchMaths – Mittels Programmierung mathematischen Wissen vertiefen

ScratchMaths ist ein für Großbritannien für die 5. und 6. Schulstufe entwickeltes Curriculum, bei dem wichtige mathematische Konzepte mittels Programmierung in Scratch vermittelt werden. Der erste Teil dient der OCG als Grundlage für die Erstellung des Syllabus für das geplante ECDL endorsed Module „Junior Coder".
https://www.ucl.ac.uk/ioe/research/projects/scratch-maths

5) Weiterführende Programmiersprachen TigerJython

TigerJython besteht aus einem Online-Lehrmittel und einer speziell für den Unterricht entwickelten Entwicklungsumgebung für die Programmiersprache Python. Das Online-Lehrmittel setzt bei der Turtlegrafik ein, führt aber weiter zu Themen wie der Programmierung von Lego-Robotern, Multimedia, Computerspielen, bis hin zu Datenbanken und stochastischen Simulationen. Zusammen mit dem modularen Aufbau und den zahlreichen Beispielen und Übungen eignet sich TigerJython sowohl für den Einsatz im Unterricht wie zum Selbststudium.

Script	Number in repeat block	Number of degrees in turn block	Total number of degree Tile sprite turned
repeat 8 / move 50 steps / turn 45 degrees / stamp	8	45 degrees	360 degrees
repeat 4 / move 50 steps / turn 90 degrees / stamp	4	90 degrees	360 degrees
repeat 10 / move 50 steps / turn 36 degrees / stamp	10	36 degrees	360 degrees

Abbildung: Erzeugung von geometrischen Mustern mit Scratch

TigerJython ist in Österreich als Lernumgebung für das neue Modul ECDL Computing vorgesehen.
http://www.tigerjython.ch/

6) Coding mit einfachen Microkontrollern micro:bit

Der BBC Micro:Bit ist ein sehr günstiges Microkontroller-Board, das 11- bis -12 jährigen britischen Schülern kostenlos zur Verfügung gestellt wird. Damit soll die Lust auf hardwarenahes Programmieren gefördert werden.

Auf der Platine befinden sich eine LED-Matrix (5 × 5), zwei Taster, eine Bluetooth-Schnittstelle, ein Beschleunigungs- und ein Magnetfeldsensor. Für die im Webbrowser erfolgenden Programmierung stehen mehrere visuelle Code-Editoren zur Verfügung, aber auch die textbasierte Programmiersprache Python (MicroPython).
http://microbit.org/

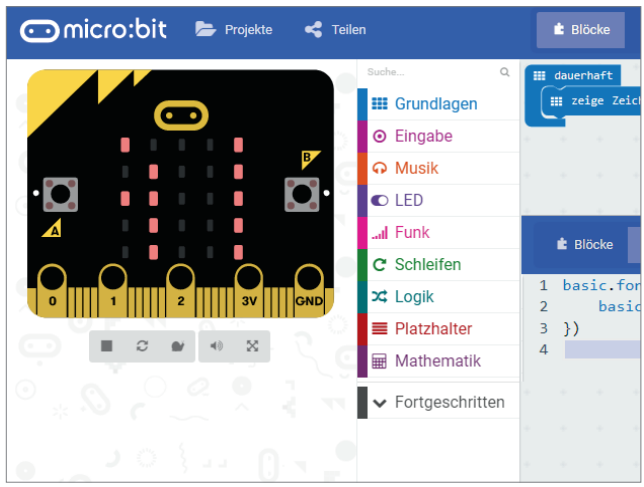


Abbildung: micro:bit. Der Block-Editor kann in die JavaScript-Darstellung umgeschaltet werden.

Autor

Mag. Dr. Johann Stockinger
Ist seit 2001 Mitarbeiter der Österreichischen Computer Gesellschaft (OCG) und leitet derzeit den Bereich Forschung & Innovation. Ein großes Anliegen ist es ihm, einfache Einstiege in das informatische Denken zu fördern. Er hat zum Beispiel die Wiener Zauberschule der Informatik (WIZIK) initiiert (https://www.ocg.at/de/wizik-wiener-zauberschule-der-informatik). Für die TU Wien arbeitet Stockinger am Erasmus+ Projekt „Head in the Clouds" vorwiegend im Bereich Educational Robotics mit. Stockinger hat Kultur- und Sozialanthropologie, formale Logik, Wissenschaftstheorie- und Wissenschaftsforschung an der Universität Wien studiert.

OCG forciert „Bildung 4.0“

Digitalisierung gehört auf den Stundenplan – jedes Kind soll künftig ein Grundverständnis in Informatik und Programmieren in der Schule lernen. Dazu bietet die OCG ab Herbst im Zuge des Europäischen Computer Führerscheins (ECDL) auch zwei Coding-Module an.

Die Österreichische Computer Gesellschaft (OCG) bemüht sich seit vielen Jahren, das Thema Informatik und informatische Bildung besser im heimischen Schulsystem zu verankern. „Gerade im Zeitalter der Digitalisierung ist es ein Gebot der Stunde, dass auch digitale Bildung flächendeckend im österreichischen Schulwesen vermittelt wird“, betont OCG-Präsident Wilfried Seyruck, „wir müssen die Schulen und alle daran Beteiligten dazu jetzt rasch ins Boot holen“. Bereits im September 2016 hat die Österreichische Computer Gesellschaft ihr umfassendes Konzept „Bildung 4.0“ (verfügbar unter: www.ocg.at/download) präsentiert.

Unter „Bildung 4.0“ versteht die OCG ein Drei-Säulensystem für die Ausbildung: Informatik und informatisches Denken, IKT-Anwendungskompetenz sowie Medienbildung. „Alle drei Säulen müssen in der Schule altersgerecht vermittelt werden“, fordert Seyruck. Die OCG agiert mit diesem Konzept auch als Partner des Bildungsministeriums und dessen Offensive „Schule 4.0“, die von der OCG unterstützt wird.

„Es geht nicht darum, dass jedes Kind jetzt Programmierer werden soll, sondern alle Schülerinnen und Schüler sollen ein grundlegendes Verständnis in punkto digitaler Bildung erwerben“, erklärt der OCG-Präsident. Wichtig sei dabei, an alle notwendigen Faktoren zu denken. Hardware und Infrastruktur sind wichtige Voraussetzungen, darüber hinaus muss auch an Lehrunterlagen, Lehrangebote und die Lehrerausbildung gedacht werden. Dazu bietet die OCG den Schulen bereits jetzt ein vielfältiges Angebot: http://www.ocg.at/ocg4schools

ECDL als modernes IKT Zertifikat

Bereits seit 20 Jahren (!) ein wichtiger Beitrag zur IKT Anwendungskompetenz ist der ECDL, der Europäische Computer Führerschein, der von der OCG sowohl für die Schulen als auch in der Erwachsenenbildung angeboten wird. Für die Schulen wird von der OCG derzeit vor allem der ECDL Standard (sechs Pflichtmodule und ein Wahlmodul) sowie der ECDL Advanced für Fortgeschrittene offeriert http://www.ecdl.at/schule. Eine Erweiterung in Richtung neuer Module für Computing/Coding zum Erwerben grundlegender Programmierkenntnisse und informatischen Denkens (Computational Thinking) ist gerade in Ausarbeitung. Konkret wird die OCG ab Herbst die neuen ECDL-Module ECDL Junior Coder (Scratch) sowie ECDL Computing (Python) anbieten (siehe Kasten).

Das gute Zusammenspiel von Experten (Informatik-Didaktikern, Bildungswissenschaftlern, Pädagog. Hochschulen), Lehrerinnen und Lehrern, IT-Praktikern und Fachangestellten aus der Industrie sowie natürlich den Schüler und Schülerinnen selbst ist sicher für die Zukunft von Bedeutung, damit digitale Bildung gelingt“, ist Seyruck überzeugt. Die OCG trägt gerne mit ihrer langjährigen Expertise und dem gesamten Mitglieder-Netzwerk zu diesem Prozess bei. Einen Überblick über bestehende Coding-Initiativen und Angebote bietet die OCG-Plattform http://www.coding4you.at. Die OCG bietet auch Informatik-Schul-Wettbewerbe an. Für alle Kinder ab der dritten Klasse Volksschule gibt es den Biber-der-Informatik http://www.ocg.at/biber, für die Teilnahme bei diesem Online-Wettbewerb ist keinerlei informatisches Vorwissen erforderlich. Der

Impressum:
Verleger: CDA Verlags- und Handelsges.m.b.H, **Herausgeber:** MinR Dr. Anton Reiter, Abteilung II/8 – IT Didaktik und Digitale Medien, Minoritenplatz 5, 1010 Wien, **Redaktionsanschrift:** A-4341 Arbing, Bundesstr. 9, Tel.: (+43) 07269/60220, Fax: (+43) 07269/60220-44, **E-Mail:** redaktion@cda-verlag.com, **Internet:** http://www.cda-verlag.com

Manuskripte und Programme: Es wird keine Haftung für unverlangt eingesandte Manuskripte übernommen. Die Einsendung von Manuskripten jeder Art gilt als Zustimmung des Verfassers zum Abdruck in den vom Verlag herausgegebenen Publikationen. Der Verlag behält sich das Recht vor, eingesandte Manuskripte nicht zu veröffentlichen. Eine Gewähr für die Richtigkeit der Veröffentlichung kann nicht übernommen werden. Für den Inhalt der Anzeigen haftet ausschließlich der Inserent, eine Prüfung seitens des Verlags erfolgt nicht!

Biber-Wettbewerb läuft immer im November, heuer von 6. bis 17.11.2017. Im Jugendbereich organisiert die OCG zudem den ccw (computer creativ wettbewerb) sowie die Teilnahme von Schülern an der nationalen und internationalen Informatik-Olympiade.

OESTERREICHISCHE COMPUTER GESELLSCHAFT ®
AUSTRIAN COMPUTER SOCIETY

Autor

Mag. Dr. Christine Wahlmüller-Schiller
leitet seit Mai 2016 den Bereich Marketing & Kommunikation der Österreichischen Computer Gesellschaft (OCG) und schreibt als IT-Fachjournalistin u.a. für die Zeitschriften Computerwelt und it&tbusiness.
Seit 2003 war die Kommunikationsexpertin als IT-Fachjournalistin, vor allem für den Bohmann Verlag und die IT-Fachzeitschrift Monitor, im Einsatz. Zuvor war die Oberösterreicherin als Marketing und PR-Chefin bei namhaften IT-Unternehmen tätig. Wahlmüller hat Publizistik, Spanisch, Geschichte und Internationale BWL an der Universität Wien und der London School of Media studiert.

Neue ECDL Coding Module

1. ECDL Junior Coder
Als Einstieg in die Welt des Programmierens und Codings wird von der OCG das Modul ECDL Junior Coder angeboten. Basis dabei ist die die visuelle Programmiersprache Scratch, die den spielerischen Zugang zur Informatik eröffnet. Zahlreiche Schüler-Scratch-Workshops in der OCG haben gezeigt, dass Scratch besonders geeignet ist, bereits in der Volksschule eingesetzt zu werden. Gleichzeitig werden die mathematischen Kompetenzen gestärkt.
Zielgruppe: 5. bis 6. Schulstufe

2. ECDL Computing
Beim zweiten Coding-Modul, ECDL Computing, sollen die Schülerinnen und Schüler zunächst Computational Thinking erlernen. Zweitens geht es darum, einfache Computerprogramme selbst zu schreiben und ein tieferes Verständnis für die Welt der IKT und der Informatik zu entwickeln. Die konkrete Umsetzung erfolgt in der Programmiersprache Python und der Entwicklungsumgebung TigerJython.
Zielgruppe: 7. bis 9. Schulstufe