PIXEL BIT

Nº 62 SEPTIEMBRE 2021
CUATRIMESTRAL

e-ISSN:2171-7966I
SSN:1133-8482

Revista de Medios y Educación

In Memoriam
Dr. Ángel Pío González Soto

CALIDAD REVISTAS CIENTÍFICAS ESPAÑOLAS 2021
FECYT-166/2021
Fecha de certificación: 06 de octubre de 2014 (4ª convocatoria)
Válido hasta: 30 de julio de 2022

Scopus
CLARIVATE ANALYTICS EMERGING SOURCES CITATION INDEX INDEXED

PIXEL-BIT

REVISTA DE MEDIOS Y EDUCACIÓN

Nº 62 - SEPTIEMBRE - 2021

https://revistapixelbit.com

EDITORIAL
UNIVERSIDAD DE SEVILLA

Píxel-Bit. Revista de Medios y Educación. 2021 - ISSN: 1133-8482. e-ISSN: 2171-7966.

PIXEL-BIT
REVISTA DE MEDIOS Y EDUCACIÓN
SEPTIEMBRE 2021

Nº 62

Píxel-Bit. Revista de Medios y Educación. 2021 - ISSN: 1133-8482. e-ISSN: 2171-7966.

Píxel-Bit. Revista de Medios y Educación. 2021 - ISSN: 1133-8482. e-ISSN: 2171-7966.

FACTOR DE IMPACTO (IMPACT FACTOR)

SCOPUS (CiteScore Tracker 2021: 2.8) - Journal Citation Indicator (JCI). Posición 400 de 722 revistas Puntuación: 44.67 (Q3) - FECYT: Ciencias de la Educación. Cuartil 2. Posición 16. Puntuación: 39,80- DIALNET MÉTRICAS (Factor impacto 2019: 1,355. Q1 Educación. Posición 11 de 230) - REDIB Calificación Glogal: 29,102 (71/1.119) Percentil del Factor de Impacto Normalizado: 95,455- ERIH PLUS - Clasificación CIRC: B- Categoría ANEP: B - CARHUS (+2018): B - MIAR (ICDS 2020): 9,9 - Google Scholar (global): h5: 42; Mediana: 42 - Journal Scholar Metric Q2 Educación. Actualización 2016 Posición: 405ª de 1,115- Criterios ANECA: 20 de 21 - INDEX COPERNICUS Puntuación ICV 2019: 95.10

Píxel-Bit, Revista de Medios y Educación está indexada entre otras bases en: SCOPUS, Fecyt, Iresie, ISOC (CSIC/CINDOC), DICE, MIAR, IN-RECS, RESH, Ulrich's Periodicals, Catálogo Latindex, Biné-EDUSOL, Dialnet, Redinet, OEI, DOCE, Scribd, Redalyc, Red Iberoamericana de Revistas de Comunicación y Cultura, Gage Cengage Learning, Centro de Documentación del Observatorio de la Infancia en Andalucía. Además de estar presente en portales especializados, Buscadores Científicos y Catálogos de Bibliotecas de reconocido prestigio, y pendiente de evaluación en otras bases de datos.

EDITA (PUBLISHED BY)

Grupo de Investigación Didáctica (HUM-390). Universidad de Sevilla (España). Facultad de Ciencias de la Educación. Departamento de Didáctica y Organización Educativa. C/ Pirotecnia s/n, 41013 Sevilla.
Dirección de correo electrónico: revistapixelbit@us.es . URL: https://revistapixelbit.com/
ISSN: 1133-8482; e-ISSN: 2171-7966; Depósito Legal: SE-1725-02
Formato de la revista: 16,5 x 23,0 cm

Píxel-Bit. Revista de Medios y Educación. 2021 - ISSN: 1133-8482. e-ISSN: 2171-7966.

PIXEL-BIT
REVISTA DE MEDIOS Y EDUCACIÓN

Nº 62
SEPTIEMBRE 2021

índice

Píxel-Bit. Revista de Medios y Educación. 2021 - ISSN: 1133-8482. e-ISSN: 2171-7966.

e-ISSN: 2171-7966
PIXEL-BIT
REVISTA DE MEDIOS Y EDUCACIÓN
MEDIA & EDUCATION JOURNAL

Coding, robotics and socio-emotional learning: developing a palette of virtues

Codificación, robótica y aprendizaje socioemocional: cómo desarrollar una combinación de habilidades

Marina Umaschi Bers, PhD.
Professor, Eliot-Pearson Dept. of Child Study & Human Development; Dept. of Computer Science; Director, DevTech Research Group
Tufts University, EE.UU.

Received: 2021/02/19; **Revised:** 2021/02/27; **Accepted:** 2021/06/25; **Preprint:** 2021/07/22; **Published:** 2021/09/01

ABSTRACT
This paper describes a pedagogical approach, Coding as Another language (CAL) to teach programming and computational thinking in early childhood. The CAL curriculum connects powerful ideas from the discipline of computer science with ideas from literacy in a way that is developmentally appropriate for children 4-8 years of age. CAL is free and can be used with two widely available programming environments for young children: the free on-screen ScratchJr app and the KIBO robotics kit that doesnt require keyboards or screens. Through 24 lessons centered on books, CAL emphasizes creative play and self-expression by positioning the learning of programming as the mastering of a new symbolic language. In addition, CAL provides opportunities for socio-emotional development in the context of a collaborative play-based learning environment, a coding playground, in which there is purposeful exploration of ethical and moral values and intentional promotion of positive behaviors and chrachter strenghs.

RESUMEN
Este artículo describe un enfoque pedagógico, Codificacion en otro lenguaje (CAL), para enseñar programación y pensamiento computacional en la primera infancia. El plan de estudios CAL conecta ideas importantes de la disciplina de la informática con ideas de la alfabetización de una manera apropiada para el desarrollo de los niños de 4 a 8 años de edad. CAL es gratuito y puede utilizarse con dos entornos de programación disponibles para los niños más pequeños: la aplicación gratuita ScratchJr y el kit de robótica KIBO, que no requiere teclados ni pantallas. A través de 24 lecciones centradas en libros, CAL hace hincapié en el juego creativo y la autoexpresión, situando el aprendizaje de la programación como el dominio de un nuevo lenguaje simbólico. Además, CAL proporciona oportunidades para el desarrollo socio-emocional en el contexto de un entorno de aprendizaje basado en el juego colaborativo, un juego de codificación, en el que hay una exploración intencional de los valores éticos y morales y la promoción intencional de comportamientos positivos y sus fortalezas.

KEYWORDS · PALABRAS CLAVES
Computational thinking; early childhood education; programming; robotics; coding.
Pensamiento computacional; Educación infantil; programación; robótica; codificación.

1. Introduction

The teaching of computer science has been growing in popularity all over the world, with a special focus on starting in the early years. In parallel, there is growing research and increasing interest in how to integrate computational thinking (CT) not only into computer science classes, but also througout all areas of learning. While there are multiple definitions of CT, there is agreement that CT is a set of cognitive processes typically exercised in computer sceince (computer programming), but is also applicable to other disciplines (Wing, 2006; Chen, et al., 2017; Lye & Koh, 2014; Tang, et al., 2020; Zhang & Nouri, 2019). Little is known about the factors that influence learning and development of this important skill set in young children or how to suppprt it. However, the learning environment, the technology and the curriculum used, all play an imporant role in the development of CT.

In the United States, women and certain minority groups are under-represented among those who choose to pursue a degree in Computer Science (CS) (National Center for Women and Informational Technology, 2020; Google & Gallup, 2016) and stereotypes regarding who is good at CT and at coding, and who is not, start to show early in life (Özyurt & Özyurt, 2015; Sullivan & Bers, 2016; Cheng, 2019). In addition, access to technology may vary as a function of race, socio-economic status and other factors (Google & Gallup, 2016). A 2021 international study found that students from lower SES have lower level CT skills on a task-based assessment than those from economically advantaged backgrounds (Karpiński, et al., 2021). Little is currently known about whether and how these and other factors influence the acquisition of CT skills in young children. However, research shows that both from an economic and a developmental standpoint, educational interventions that begin in early childhood have lower costs and durable effects (Cunha & Heckman, 2007). This paper presents a pedagogical approach, Coding as Another Language (CAL), that has been succesfully used to introduce creative computer programming and to promote the development of CT in early childhood. CAL is unique as it understands the aquisition of CT and CS skills as a new literacy of the XXIst century.

The CAL curriculum connects powerful ideas from the discipline of CS with ideas from literacy in a way that is developmentally appropriate for children 4-8 years of age. CAL is free and can be used with two widely available programming environments for young children: the free ScratchJr app and the KIBO robotics kit (Bers, 2018).

The CAL curriculum not only focuses on CS, but offers oppprtunities to promote a growth mindset through the practice of coding. Dweck (2006) defined a growth mindset as the belief that talents can be developed, through hard work, good strategies, and input from others. CAL takes this idea forward. The ultimate goal is not only to promote computational talents and skills, but also virtues and values. That is, CAL is a vehicle for building character, and to develop social relationships and emotional strengths.

2. The Coding as Another Language curriculum

The CAL Curriculum was developed by the DevTech Research Group at Tufts University and is freely available here: https://sites.tufts.edu/codingasanotherlanguage/.

This curriculum is designed to teach coding through developmentally apropriate tools such as KIBO robotics and ScratchJr, and integrates the teaching of creative coding skills and CT with literacy skills (Hassenfeld et al., 2020). This pedagogical approach emphasizes creative play and self-expression by positioning the learning of programming as the mastering of a symbolic language to communicate (Bers, 2018; 2019). In addition, CAL provides opportunities for socio-emotional development in the context of a collaborative play-based learning environment: a coding playground (Bers, 2018).

The coding playground engages children in both on-screen and off-screen activities. When programming, children put together a sequence of logical instructions and translate those instructions into a symbolic system of representation that the computer or the robot can understand: a programming language. Thus, programming positions the child as an agent, as someone who can make things happen, and as someone with a voice (Resnick & Siegel, 2015). As the child codes, she develops technical skills and CT. She can problem-solve and deal with abstraction; she can sequence, understand patterns and use variables and conditionals.

Each unit contains 24 45 minutes lessons, centered on coding projects about books, both fiction and non-fiction. For example, fictional storybooks include Where the Wild Things Are by Maurice Sendak or There Was an Old Lady Who Swallowed a Fly by Simms Taback; non-fiction books tell the story of a pioneer woman in computer science, such as Ada Lovelace or Grace Hopper. Teachers are encouraged to substitute any of these books with their own favorite books, as long as they have a clear sequencing of events. Children create their own endings for their books and learn how to re-tell the stories in creative ways using either KIBO robotics or ScratchJr animations.

The CAL curriculum is organized around a scope and sequence of seven powerful ideas from computer science that are age-appropriate and that promote CT (Bers, 2019; 2018): hardware/software systems, algorithms, modularity, control structures, representation, debugging, and design process (Table 1).

Table 1.

The seven powerful ideas, associated concepts, and examples from the CAL-KIBO curriculum

Powerful Idea	Associated Concepts	Example from CAL-KIBO Curriculum
Algorithms	Sequencing/order, logical organization	Child learns to program KIBO in a specific sequence to dance the “Hokey Pokey”
Modularity	Breaking up larger task into smaller parts, instructions	Students break up the “If You're Wild and You Know It” song into smaller components that KIBO can be programmed to perform
Control Structures	Recognizing patterns and repetition, cause and effect	Children learn to trigger sound sensors using “wait for clap” command
Representation	symbolic representation, models	Child learns that each programming block translates into a unique KIBO action.
Hardware/Software	Smart objects are not magical, objects are human engineered	Children play a game about what is and isn't a robot and learn that you must give the KIBO robot a program in order for it to perform
Design Process	Problem solving, perseverance, editing/revision	Children are tasked with creating a final “Wild Rumpus” KIBO project in which they plan, code, test and revise with peer sharing and feedback
Debugging	Identifying problems, problem solving, perseverance	Children identify problems in either hardware or software of KIBO and brainstorm solutions to fix it

The CAL curriculum is designed to be flexible. The timing can be adjusted to make lessons longer or shorter to better suit the curricular needs of different schools. Each lesson follows a similar structure: warm up games to playfully introduce computational ideas, coding activities to solidify skills, structured challenges to practice, creative explorations to tinker and expand skills, off-screen unplugged games to promote social interactions and movement, reading and writing activities, and technology circles to share and reflect (Sullivan & Bers, 2017; 2015). The curriculum is comprised of individual, small group and whole classroom activities.

While the content is organized in terms of the seven powerful ideas of computer science (e.g. algorithms, design process, representation, debugging, control structures, modularity, and hardware/software systems), explicit connections are made in each of the units to early childhood literacy (e.g. the writing process, recalling, summarizing and sequencing, using illustrative and descriptive language, recognizing literary devices such as repetition and foreshadowing, and using reading strategies such as predicting, summarizing, and evaluating). Furthermore, low-tech games or unplugged activities aimed at promoting computational thinking and alphabetical literacy are also incorporated (Bers, 2018).

Throughout the CAL lessons, one curricular domain is used to leverage the other (Strawhacker & Bers, 2019). For example, when children encounter algorithmic thinking they also explore sequencing and storytelling, when they engage in the design process, active connections to the writing process are made, and when they set to debug their ill-functioning programs, they tap into revising strategies that share similarities with the systematic editing of their writing.

There are significant differences between using programming languages and natural languages for expressing ourselves (Fedorenko et al, 2019). CAL doesn't ignore these. However, as an integrated curriculum, the focus is on shared practices (Hassenfeld et al, 2020; Hassenfeld & Bers, 2020): the creation of projects, either through coding or through writing, the creative design process involved in making these projects, the need to revise and fix them at each step of the way, and the sharing of final products with others as a way to express our individuality, our interests, passions and identities.

At its simplest level, computer programming is the activity of putting together a sequence of instructions. In the process of making this sequence, the programmer engages in abstract, logical thinking (Kazakoff & Bers, 2014). Thinking is facilitated by language. As Soviet psychologist Lev Vygotsky (1978) wrote: “Thought development is determined by language, i.e., by the linguistic tools of thought.” Thus, early childhood educators strive to help children develop one of the most powerful tools for thinking: natural written languages.

CAL positions the teaching and learning of programming as the study of a socially situated symbolic system of representation with communicative and expressive functions to promote human encounters. It proposes that programming, as a new literacy of the 21st century, engages new ways of thinking, communicating and expressing ideas -- not only new ways of problem-solving. The goal of literacy is to master the syntax and grammar of a language, but also the meanings and uses of the systems of representation (Vee, 2017). A literate person knows that reading and writing are tools for interpretation and, in time, tools of power. (Bruner, 1983; Wolf, 2007) Echoing Brazilian educator Paulo Freire, literacy is a tool for critical comprehension, for understanding the world and for actively changing it. This is the same with coding.

The CAL curriculum also supports socio-emotional development. Children work hard individually and in teams, and are proud to share their projects with others in the community. They develop gratefulness to their peers and teachers for providing help and support during the hard process of learning to code. They understand the need of determination, persistence and patience to complete their work. They are honest with themselves and choose to keep problem-solving when a project is not exactly what they hoped for. They also learn to forgive themselves for being slow and for not getting it right all the time. They understand that coding involves a constant process of iteration and revision in which flexibility and open-mindedness are needed. The CAL curriculum helps create an environment for practicing human values through learning how to use a programming language; for understanding that our actions, like the actions of anyone who creates, have consequences.

2.1. The KIBO robot

The KIBO robotics kit utlized in the CAL-KIBO curriculum was developed by the DevTech research group at Tufts University with funded by the National Science Foundation (grant # NSF DRL-1118897) and is commercially available by KinderLab Robotics.

KIBO is programmed using tangible wooden programming blocks (Figure 1). The child creates a sequence of instructions (a program) using the wooden blocks and KIBO reads the barcodes with an embedded scanner. With the press of a button children watch the robot come alive. The KIBO programming language contains unique blocks, sensors and actuators leading to endless creative possibilities.

Each wooden block represents an action that the robot performs when read by the scanner embedded on the robot. The combination of KIBO's blocks, sensors, modules, and art platforms gives children a unique opportunity to not only explore programming concepts but also to use their creativity to create personally meaningful projects.

A decade of research has been conducted on KIBO, involving thousands of children, teachers and families from schools around the US and the world (Albo-Canals et al., 2018; Bers 2018, 2019a, 2019b, 2020; Elkin, Sullivan, & Bers, 2016, 2018; Govind & Bers, 2020; Strawhacker & Bers, 2018; Sullivan, Bers, & Mihm, 2017; Sullivan, Elkin, & Bers, 2015). This research has found that KIBO supports learning of computational thinking skills such as sequencing, iterative design, debugging, and more (Bers, 2019a, 2020; Kazakoff & Bers, 2012). Further, this learning has been demonstrated in children from a range of backgrounds and in various learning settings, including: PreK-2nd graders in US classroom settings (Bers 2019b, 2020; Elkin, Sullivan, & Bers, 2016); international early childhood students in countries such as Singapore (Bers, 2020; Elkin, Sullivan, & Bers, 2018), Argentina (Bers, 2020), and Denmark (Strawhacker & Bers, 2018); children in informal settings like makerspaces, libraries, and family centers (Govind & Bers, 2020; Strawhacker & Bers, 2018); and children on the Autism spectrum (Albo-Canals et al., 2018). In addition, the learning of discipline-specific content learning in foundational areas of math and literacy can be supported through KIBO (Bers, 2019a, 2020; Kazakoff & Bers, 2012).

Figure 1.

KIBO robotics

In one of the earliest studies of KIBO and its coding language ("CHERP"), a controlled experimental trial involving 54 Kindergarten children from 2 schools, results showed that programming with KIBO's tangible blocks was linked to significant improvements in children's story-based sequencing skills, a critical component of foundational literacy education (Kazakoff & Bers, 2012). The same study called for more research into teacher's STEM preparation, and another study responded to this call by exploring professional development with the KIBO prototype (originally called "KIWI") (Bers, Seddighin, & Sullivan, 2013). This research involved 32 early childhood educators who participated in an intensive three-day professional development (PD) workshop. Results showed a statistically significant increase in the teachers' level of knowledge about robotics, programming, and engineering after the PD, as well as significant increases in several aspects of technology self-efficacy and attitudes toward technology (Bers, Seddighin, & Sullivan, 2013). These successful PD practices are still used today, in the Early Childhood Technology (ECT) graduate certificate program at Tufts University. A quasi-experimental longitudinal study carried out in multiple schools in a Virginia school district found that the CAL-KIBO curriculum significantly improved coding and CT skills in young children (Relkin et al., 2020; Hassenfeld et al., 2020; Relkin et al., 2021). As part of this study, participating teachers received intense professional development, completed surveys about their teaching experience and CS background and took an assessment specific to the KIBO robotics platform that evaluates coding/ CT skills (Relkin, 2018; Relkin & Bers, 2019).

KIBO is designed to match the following criteria: 1) robotics parts are physically and intuitively easy to connect, 2) programming the robot requires minimal computer equipment, and 3) children can attach a variety of crafts and recycled materials to the core robotic parts (Sullivan, Elkin, & Bers, 2015). Throughout development and after commercialization, KIBO has remained screen-free, in line with research conducted at DevTech that found tangible interfaces may be more beneficial for engaging young children in early coding experiences (e.g. Pugnali, Sullivan, & Bers, 2017; Strawhacker, Sullivan, & Bers, 2013). This screen-free approach has made KIBO accessible U.S. K-2 classroom, and a range of other audiences as well. KIBO's design was informed by a theoretical framework that promotes positive uses of technology in the context of playful learning environments: Positive Technological Development.

3. Positive Technological Development: an applied theoretical framework

The Positive Technological Development (PTD) framework (Bers, 2012) is inspired by the field of Positive Youth Development (Lerner, 2007) and provides a theoretical lens to capture psychosocial behaviors in the context of using technology. As an applied research framework that aims to help those interested in educational interventions, PTD provides guidelines for designing and evaluating technological programs to promote character strengths through six positive behaviors:

- Content Creation: The act of coding involves using an artificial language to create. In this journey, the child engages in a series of interrelated steps that might or might not be linear: the design process. To create her own project, she learns to ask questions, identify a goal, formulate an action plan, make an initial attempt, test, evaluate, and revise her ideas by assessing what went wrong and what could be done better. At the end of the creation process, she has a sharable project.

- Creativity: The ability to transcend traditional strategies to imagine and create original projects supports personally meaningful expression. A creative child can frame problems in innovative ways and find divergent approaches and solutions. However, creativity requires training and hard work (Resnick, 2017). Contradicting some popular myths, the creative child is not necessarily the one who wakes up one morning saying “Eureka!” but the one who is disciplined in her work, takes risks and can find new connections.

- Choices of Conduct: Anytime we do something, we make choices and must assume consequences. This process, when authentic, builds character. On the one hand, our character strengths inform the choices we make. On the other hand, those choices have an impact on our character. We are surrounded by news about people choosing to use their coding skills in positive or negative ways, to help or to harm society. Coding is a tool and, like any other tool, can be used for good or bad. Like a hammer, it can build or destroy.

- Communication: Language socialization plays a key role in cognitive development, as well as personal, social and emotional growth. Children engage in conversations, with either themselves or with others, to externalize ideas and thoughts. However, most programming languages do not have a built-in feature to promote communication. Thus, the curriculum must provides explicit communication mechanisms and strategies to support the formation and sustainment of positive bonds through coding.

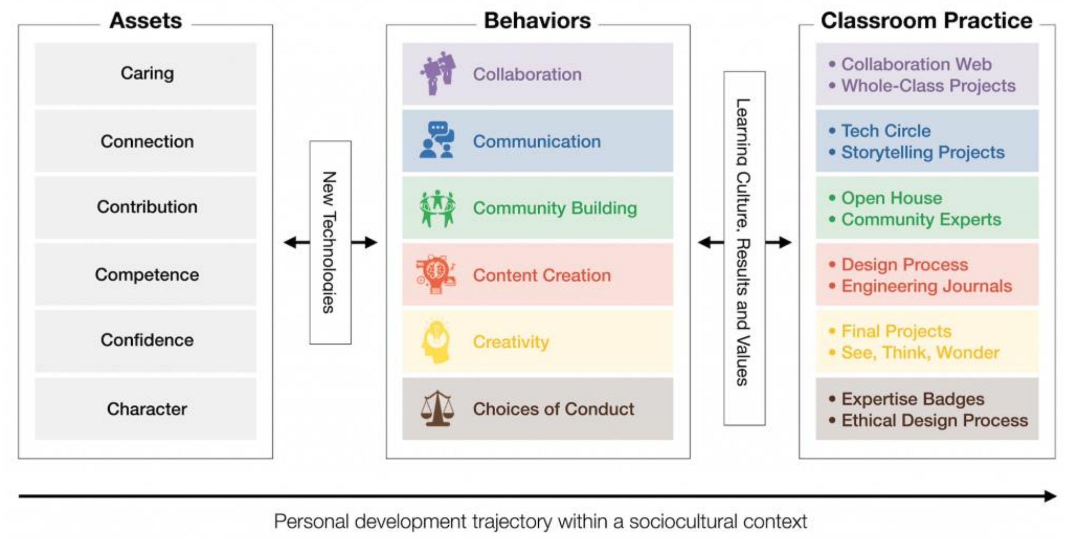
- Collaboration: Two or more people working on a team is not the same as collaboration. For collaboration to happen, there is a need for a shared goal and cooperation on a common task. This can be challenging in early childhood; for a typically developing young child, the turn-taking, self-control, and self-regulation required to effectively collaborate on a project is difficult, thus the curriculum must suppprt this.

- Community Building: The establishment and sustainment of social relationships in the learning environment is crucial and can be achieved by putting together mechanisms for giving back to others, and contributing to our communities. For example, open houses and family coding nights in which children demo their coding projects are an authentic opportunity to share and celebrate the processes and products of learning with parents, family and friends.

These six behaviors, the 6 C's, are value neutral. We can create a video game to practice shooting skills or to learn the ABC's, we can communicate in dysfunctional ways to harm others or to praise, or we can choose to include others in our teams or exclude them. Thus, a coding playground needs guiding values, and not only behaviors. While different cultural contexts might have a diversity of values, programming in a culture in which the act of creative production is rewarded, lends itself to values such as curiosity, determination and persistence. Next, I will describe the most salient values, or virtues, that can be found in a coding playground.

Figure 1.

Positive Technological Development (PTD) Framework



Source: Bers (2018)

3.1. Values in the coding playground

The PTD framework described above can guide the design of a learning environment that promotes both cogntive and socio emotional development: a coding playground. The CAL curriculum was designed upon this framework and pays special attention to the operationalization of the last C, Choices of Conduct. CAL invites teachers to create a coding playground in which children can make behavioral choices based upon a palette of virtues consisting of: Curiosity, perseverance, patience, open-mindedness, optimism, honesty, fairness, generosity, gratitude, and forgiveness.

The term *palette of virtues* refers to the metaphor of the color palette used by the artist. She chooses colors and creates her own palette. She mixes and matches. She adds new colors. There is no absolute right and wrong; it depends on the context of how the colors are used and their relationships. This flexibility reflects the intentionality of working with values in the coding playground. CAL proposes a palette of virtues so the codign playground can serve as another educational space to promote character development and positive behaviors.

For instance, while some teachers might focus on turn-taking, taking care of materials, and learning how to work collaboratively with others, others might pay attention to learning how to be patient when trying to problem-solve or how to help others debug. Some might use mindfulness for helping children work through the frustration of trying to debug with little success, and others might use thank you cards to acknowledge the generous spirit of helping each other problem-solve.

Mitch Resnick uses the imagery of the paintbrush for describing the activity of coding. In 2006, he wrote: “In my view, computers will not live up to their potential until we start to think of them less like televisions and more like paintbrushes.” (Resnick, 2006). He was referring to the creative and expressive potential of paintbrushes. I am extending the metaphor. The paintbrush by itself is not enough. It needs colors. In the coding playground, the child is the artist who learns to code. The paintbrush is the programming language that supports creativity. The colors are the values the child learns and expresses while coding.

The coding playground becomes an art studio for practicing a palette of virtues while developing problem solving, computational thinking and technical skills. Children learn by doing, by experimenting, by trial and error, by collaborating with others and solving social conflicts, by feeling overwhelmed with the challenge ahead, and by learning how to manage frustration. In the social interactions, character strengths are developed, and values are put to practice. In the coding playground, socio-emotional development does not take the back seat. Creative programming is a pathway for character development. Understanding coding as another language facilitates this process. When we learn a new language, we also must learn how to use it in responsible ways. Languages can create and can destroy.

4. Final thoughts

While programmers have been around for a century, philosophers have existed long before programmers. Amongst other things, their work was concerned with how to translate human language into a structured argument with consistent logic in its premises and conclusions. My work, by focusing on learning to code as learning another language, embraces this. As an heir to Aristotle’s logical systems, programming can serve as a gateway to critical thinking, not only about technical problems, but also about societal issues.

Today, more than ever, we need a critical mindset. The rapid acceleration of new discoveries and technological innovations, coupled with the unparalleled access to information “anytime, anywhere” through the Internet, has created new sets of problems (Resnick & Rusk, 2020). The intellectual tools to think about these problems have been around since the days of Socrates and Aristotle: understanding the structure of an argument, and translating human language into the premises and conclusions that make up the basis for logical analysis. These intellectual tools allow us to judge information, to evaluate evidence, and to make decisions by applying formal rules. Coding adds the ability to create new realities through novel systems and processes.

Programming is a verb. It involves actions, and not only thinking. Will the determination of a child who keeps debugging her program, even when outside recess is called, apply grit in every aspect of her life? Can the generosity of a child who chooses to slow down and help another, instead of programming his own robot, translate outside the coding playground? Can the creative ways in which children debug while coding, transfer to solving social problems that impact equity and justice in the world? How about a child who chooses to share his KIBO robot with a child who has none, instead of using it by himself? Will this child also display positive choices of conduct when faced with more complex decisions?

Although most grade schools do not teach formal logic, its practical application in the form of structured thinking through an artificial language, that is computer programming, is being learned by more students today than ever before. However, if we limit its application to the growth of a STEM (Science, Technology, Engineering and Math) career, we will be missing the great opportunity envisioned by the early philosophers of ancient Greece: to form the ethical character of future citizens who can grow as autonomous individuals capable of thinking systematically and independently, problem-solve when needed, and act towards the good of self and society.

Willingly or unwillingly, everyone who teaches, teaches values. That is part of the hidden curriculum. The coding playground makes values visible by offering an initial palette of virtues to work with. Most of these values and characteristics are usually displayed by successful programmers and cultures of innovation. Different traditions, societies and groups might want to add or remove some of them. Others might want to mix and match or prioritize some values and character strengths over others. The intentional teacher makes her own palette, with universal and particular elements. In the coding playground, by understanding coding as another language -- that is, by situating the activity of programming as a vehicle for expression and communication -- children can experience values and practice virtues in the context of forming human relationships. They can develop ways of thinking and behaving associated with the discipline of computer science: Problem-solving, persistence, and open-mindedness are required to break a complex problem into simple processes; the disposition to work with others is necessary because programming involves working with a system created by another human being. The coding playground is an opportunity to put to use the values in our palette and further develop them.

Today, there is a growing push for STEM in schools all over the world. The focus is mostly on technical knowledge and skills. While those are important, cultivating character virtues alongside is crucial. The Coding as Another Language approach involves much more than preparing students for STEM careers. It is about new ways of thinking, relating, and behaving. It highlights creative expression, communication and problem-solving. It underscores that coding, when conceived as a language, situates us in the social world of relationships: with ourselves, with others and with the world.

Note

This article includes excerpts from the upcoming book “Beyond Coding: How to Teach Human Values through Programming” by Marina Bers to be published by The MIT Press in Spring 2022.

References

Albo-Canals, J., Barco, A., Relkin, E., Hannon, D., Heerink, M., Heinemann, M., Leidl, K., & Bers, M. (2018). A Pilot Study of the KIBO Robot in Children with Severe ASD. *International Journal of Social Robotics, 10*(3), 371-383. https://doi.org/10.1007/s12369-018-0479-2

Bers, M. U. (2012). *Designing Digital Experiences for Positive Youth Development: From Playpen to Playground*. Routledge.

Bers, M. (2018). *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom.* Routledge. https://doi.org/10.4324/9781315398945

Bers, M. U. (2018). The Seymour test: Powerful ideas in early childhood education. *International Journal of Child-Computer Interaction*, 14, 10-14. https://doi.org/10.1016/j.ijcci.2017.06.004

Bers, M. U. (2019). Coding as another language: a pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, 6, 1-30. https://doi.org/10.1007/s40692-019-00147-3

Bers, M. U. (2019). Coding as another language In C. Donohue (Ed.), *Exploring key issues in early childhood and technology: Evolving perspectives and innovative approaches* (pp. 63–70). Routledge.

Bers, M. U. (2020). *Coding as a playground: Computational thinking and programming in early childhood*. Routledge.

Bers, M.U., Seddighin, S., & Sullivan, A. (2013). Ready for robotics: Bringing together the T and E of STEM in early childhood teacher education. *Journal of Technology and Teacher Education, 21*(3), 355-377.

Bruner, J. (1983). *Child's Talk: Learning to Use Language.* W. W. Norton & Company.

Chen, G. & Shen, J. & Barth-Cohen, L. & Jiang, S., Huang, X. & Eltoukhy, M. (2017). Assessing elementary students’ computational thinking in everyday reasoning and robotics programming. *Computers & Education*, 109, 162-175. https://doi.org/10.1016/j.compedu.2017.03.001

Cheng, G. (2019). Exploring factors influencing the acceptance of visual programming environment among boys and girls in primary schools. *Computers in Human Behavior*, 92, 361-372. https://doi.org/10.1016/j.chb.2018.11.043

Cunha, F. & James, H. (2007). The Technology of Skill Formation. *American Economic Review*, 97(2), 31-47.

Dweck, C. S. (2006). *Mindset: The new psychology of success*. Random House.

Elkin, M., Sullivan, A., & Bers, M. U. (2016). Programming with the KIBO Robotics Kit in Preschool Classrooms. *Computers in the Schools, 33*(3), 169-186. https://doi.org/10.1080/07380569.2016.1216251

Elkin, M., Sullivan, A., & Bers, M. U. (2018). Books, Butterflies, and 'Bots: Integrating Engineering and Robotics into Early Childhood Curricula. In L. English and T. Moore (Eds.), *Early Engineering Learning* (pp. 225-248). Springer. https://doi.org/10.1007/978-981-10-8621-2_11

Fedorenko, E., Ivanova, A., Dhamala, R., & Bers, M. U. (2019). The language of programming: A cognitive perspective. *Trends in cognitive sciences, 23*(7), 525-528. https://doi.org/10.1016/j.tics.2019.04.010

Google Inc. & Gallup Inc. (2016). Trends in the State of Computer Science in U.S. K-12 Schools. http://goo.gl/j291E0

Govind, M., Relkin, E., & Bers, M. U. (2020). Engaging Children and Parents to Code Together Using the ScratchJr App. *Visitor Studies.* https://doi.org/10.1080/10645578.2020.1732184

Hassenfeld, Z. R., Govind, M., de Ruiter, L. E., & Bers, M. U. (2020). If You Can Program, You Can Write: Learning Introductory Programming Across Literacy Levels. *Journal of Information Technology Education: Research*, 19, 65-85. https://doi.org/10.28945/4509

Hassenfeld, Z. R., & Bers, M. U. (2020). Debugging the Writing Process: Lessons From a Comparison of Students' Coding and Writing Practices. *The Reading Teacher*, *73*(6), 735-746. https://doi.org/10.1002/trtr.1885

Kazakoff, E., & Bers, M. (2012). Programming in a robotics context in the kindergarten classroom: The impact on sequencing skills. *Journal of Educational Multimedia and Hypermedia, 21*(4), 371-391.

Kazakoff, E.R. & Bers, M. U. (2014). Put your robot in, Put your robot out: Sequencing through programming robots in early childhood. *Journal of Educational Computing Research, 50*(4). https://doi.org/10.2190%2FEC.50.4.f

Karpiński, Z., Di Pietro, G., & Biagi, F. (2021). Computational thinking, socioeconomic gaps, and policy implications. IEA Compass: Briefs in Education Series (12). https://bit.ly/3wWbA2M

Lerner, R. (2007). *The Good Teen.* Three Rivers Press.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51– 61. https://doi.org/10.1016/j.chb.2014.09.012

National Center for Women and Information Technology. (2020). NCWIT's women in IT: By the numbers. https://bit.ly/3rtp44S

Özyurt, Ö., & Özyurt, H. (2015). A study for determining computer programming students' attitudes towards programming and their programming self-efficacy. *Journal of Theory and Practice in Education, 11*(1), 51–67.

Pugnali, A., Sullivan, A., & Bers, M.U. (2017). The Impact of User Interface on Young Children's Computational Thinking. *Journal of Information Technology Education: Innovations in Practice, 16,* 172-193.

Relkin, E. (2018). *Assessing young children's computational thinking abilities* [Master's tesis]. Retrieved from ProQuest Dissertations and Theses database. (UMI No. 10813994)

Relkin, E. & Bers, M. U. (2019). Designing an Assessment of Computational Thinking Abilities for Young Children. In L.E. Cohen & S. Waite-Stupiansky (Eds.), *STEM for Early Childhood Learners: How Science, Technology, Engineering and Mathematics Strengthen Learning* (pp. 85-98). Routledge.

Relkin, E., de Ruiter., L., & Bers, M.U. (2020). TechCheck: Development and Validation of an Unplugged Assessment of Computational Thinking in Early Childhood Education. *Journal of Science Education and Technology*, 29, 482–498. https://doi.org/10.1007/s10956-020-09831-x

Relkin, E., de Ruiter, L.E., Bers, M.U. (2021). Learning to Code and the Acquisition of Computational Thinking by Young Children. *Computers & Education,* 169*, 104222.* https://doi.org/10.1016/j.compedu.2021.104222

Resnick, M. (2006). Computer as Paintbrush: Technology, Play, and the Creative Society. In D. Singer, R. Golikoff, R. & K. Hirsh-Pasek (eds.), *Play = Learning: How play motivates and enhances children's cognitive and social-emotional growth* (pp. 1-16). Oxford University Press.

Resnick, M., & Siegel, D. (2015). A Different Approach to Coding. *International Journal of People-Oriented Programming*, *4*(1), 1-4.

Resnick, M., & Rusk, N. (2020). Coding at a Crossroads, Communications of the ACM, vol. 63, no. 11, pp. 120-127.

Resnick, M. (2017). *Lifelong Kindergarten: Cultivating Creativity through Projects, Passions, Peers, and Play*. MIT Press.

Sullivan, A., & Bers, M. U. (2017). Dancing robots: Integrating art, music, and robotics in Singapore's early childhood centers. *International Journal of Technology and Design Education, 28*, 325-346. https://doi.org/10.1007/s10798-017-9397-0

Sullivan, A. & Bers, M. U. (2016). Girls, boys, and bots: Gender differences in young children's performance on robotics and programming tasks. *Journal of Information Technology Education: Innovations in Practice, 15,* 145-165.

Sullivan, A., & Bers, M. U. (2015). Robotics in the early childhood classroom: Learning outcomes from and 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education* 26, 3-20. https://doi.org/10.1007/s10798-015-9304-5

Sullivan, A., Bers, M.U., Mihm, C. (2017). *Imagining, Playing, & Coding with KIBO: Using KIBO Robotics to Foster Computational Thinking in Young Children*: *Proceedings of the International Conference on Computational Thinking Education.* Wanchai, Hong Kong.

Sullivan, A., Elkin, M., & Bers, M. U. (2015). *KIBO Robot Demo: Engaging young children in programming and engineering*: *Proceedings of the 14th International Conference on Interaction Design and Children (IDC '15),* Medford, MA, June 21-25. ACM.

Strawhacker, A. and Bers, M. U. (2018). Promoting Positive Technological Development in a Kindergarten Makerspace: A Qualitative Case Study. *European Journal of STEM Education, 3*(3) 09. https://doi.org/10.20897/ejsteme/3869

Strawhacker, A., & Bers, M. U. (2019). What they learn when they learn coding: investigating cognitive domains and computer programming knowledge in young children. *Educational Technology Research and Development, 67*(3), 541-575. https://doi.org/10.1007/s11423-018-9622-x

Strawhacker, A., Sullivan, A., & Bers, M.U. (2013). *TUI, GUI, HUI: Is a bimodal interface truly worth the sum of its parts?*: *Proceedings of the 12th International Conference on Interaction Design and Children (IDC '13)* (pp. 309-312). ACM.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: a systematic review of empirical studies. *Computers & Education*, 148, 103798. https://doi.org/10.1016/j.compedu.2019.103798.

Vee, A. (2017). *Coding Literacy: How Computer Programming is Changing Writing*. The MIT Press.

Vygotsky, L.S. (1978) *Mind in society: The Development of Higher Psychological Processes*. Harvard University Press.

Wing, J. (2006). Computational Thinking. CACM, 49(3), 33-36. https://doi.org/10.1145/1118178.1118215

Wolf, M. (2007). *Proust and the Squid: The Story and Science of the Reading Brain*. Harper Collins.

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education*, 141, 103607. https://doi.org/10.1016/j.compedu.2019.103607.

Cómo citar:

Bers, M. U. (2021). Coding, robotics and socio-emotional learning: developing a palette of virtues [Codificación, robótica y aprendizaje socioemocional: cómo desarrollar una combinación de habilidades]. *Pixel-Bit. Revista de Medios y Educación*, 62, 309-322 https://doi.org/10.12795/pixelbit.90537