Contents

EMPOWER OUR CHILDREN 3

TEACH YOUNG CHILDREN TO CODE 4

  Give them confidence to take control 5
  Start young to master the skills 6
  Digital literacy is the future 7
  Teach children how to think 8
  Think innovatively and independently 9

CODE IN THE CLASSROOM 10

  Widen the walls 11
  Turn off the computers 12
  Mistakes are good 13
  Keep it simple 14
  References 15
I don’t believe that technology will be our savior. Unfortunately we cannot code our way out of economic, environmental or humanitarian crises. But technology in the hands of able humans will contribute to positive change, and computational thinking and coding will help us to understand how and when technology is appropriate.

At KUBO we believe that empowerment of our children to prepare them for the future requires a peer-to-peer centric and playful form for learning that is relevant and inclusive. We want to contribute to a learning environment where children are able to “shine” because they experience how it is to work collaboratively; to give space to each other’s ideas and to build each other’s confidence as they learn together.

I taught myself to code when I was eight years old and as I became aware of all the amazing things I could create with coding, a whole new world opened up to me. I want all children to experience the same kind of wonder and empowerment.

Today’s computer science curriculum delivers little more than a snapshot of what technology is already doing based on what we already know. For technology learning to be of real value we must continually push the boundaries of what is possible within the classroom. As EdTech designers it is our role to provide the tools and stimulate the mindsets that enable students to explore unforeseen possibilities.

I want children to become creative agents in pursuit of meaningful impact. Technology shouldn’t be about a virtual reality, but instead a better reality, and for that reason we pledge to deliver educational technology that doesn’t remove children from the world around them, but opens children’s minds up to the world and people around them.

In this paper we want to share our passion with fellow educators, policy-makers and industry partners; to contribute to a growing understanding of the purpose and vision for educational technology. We share our views on why coding should be part of primary education and how we believe it can be integrated into the classroom. It is a plea to take coding seriously as a future skill that allows children to participate actively and creatively in society, in an increasingly digitized world.

Daniel Lindegaard
Founder and Chief Technology Officer
TEACH YOUNG CHILDREN TO CODE
When the future is shaped to a large degree by technology those who are able to understand, use and influence it will be granted much greater decision-making power. By providing all children with technological knowledge and skills through formal education, control is shared and teaching to code becomes a democratizing act. There are widespread predictions of an increasingly significant digital gap among citizens. This gap refers to the division between those who have the skills to participate in a high-technological society and those who don’t [1]. In the past the gap has been a reference to access to technology, but with cheaper and more accessible tech developments the gap in the fourth industrial revolution will refer to those who have the ability to manipulate data – those who are digitally literate – and those who are not.

We believe that digital literacy should be taught with a healthy dose of ethics, and that students should be encouraged to think about the impact technological creation has on their community and society. They should be taught to think about ethical and sustainable purpose. Even with complex technologies increasingly present in our everyday lives many people do not understand it, nor do they believe they have much power to influence it. It is not necessarily fear, but helplessness that people experience when confronted with technology. Digital empowerment is about changing that power dynamic so that instead of experiencing technology as having power over one’s life, we are empowered to take back control. As Rushkoff [2] famously said: Program or be programmed.
Children appear to be masters of technology at an increasingly younger age and they intuitively explore and use a wide variety of technology in their everyday lives. But, while being raised with computing makes most children masters of consumption, it does not mean that they understand it, or are able to reflect critically on its use.

Most coding initiatives have targeted older children, but there are good reasons to introduce coding to children as young as four, to allow them to develop the brain models for greater abstract thinking. An early introduction also allows more children to develop excitement about coding and means that gender-based stereotypes of careers within STEM can be prevented [3]. Studies suggest that introducing coding at a very young age gives children the chance to become truly fluent [4], similar to teaching other core skills such as literacy and mathematics.
Digital literacy is the future

Digital literacy is not simply the ability to read and write code [5]. Literacy and the process of understanding an entirely new language includes so much more than reading and writing - it includes both social and cultural contexts to be able to navigate confidently, and to be truly fluent [6], [7].

Put simply, a computer only understands binary ones and zeros. It is in fact not that smart.

Coding is essentially the language that translates human language to computer language. Various coding languages exist, all intended to program a computer to do a series of operations. Although each is slightly different, coding languages, or dialects, are all based on the same vocabulary and underlying concepts.

Literacy, seen holistically, has operational, cultural and critical aspects, and these three aspects are also relevant when speaking of digital literacy [6], [8].

The operational aspect of literacy is concerned with the skills of reading, writing and sense-making, and being able to access information. It is essentially communication. The cultural aspect focuses on meaning-making and the critical aspect on the ability to engage critically with digital texts and artefacts, as well as issues of power and representation [6].

Words don’t exist in a vacuum, and neither does code, and all three dimensions of literacy are important for a comprehensive understanding of coding. Therefore, coding in education should not just be perceived as the skill of communication, but also acknowledged for its cultural and critical aspects.
Teach children how to think

Let’s teach children how to think, not what to think. Let's pave the 21st century way of thinking as the computational way.

Jeanette Wing’s 21st century definition defines computational thinking as “solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” [9]. She also asserted that computational thinking involves the thought processes needed to come up with solutions to problems, formulated so that they can be effectively carried out by computers.

In K-12 educational contexts, the concept of computation was introduced already back in the 80's by Seymour Papert and his pioneering LOGO program [10]. It was quickly and mistakenly abandoned, but the importance of coding has since re-entered the education arena. Various definitions are used, but to agree on the core concepts including abstraction, algorithmic thinking, automation, decomposition and generalization, related to the practices of debugging, collaboration, creativity, and the ability to deal with open-ended problems [11].

Coding and computational thinking are closely related. Learning coding concepts can be seen as a number of practical skills, and computational thinking processes as cognitive skills [12]. While coding would be easier to learn if you already had a good grasp of computational thinking, learning to code is simultaneously a very good way of practicing computational thinking.

So, learning to code is not just about learning to code - it is about abstraction and algorithmic thinking as well, breaking a problem down into a step-by-step process to guide someone or something to complete a task or solve a problem [13]. This is why computational thinking is not only relevant to computers, but is and can be transferred to a variety of situations and problems in life, such as baking a cake, following a recipe or assembling furniture.

Based on figure in Tsarava, 2017 [12]
Think innovatively and independently

Nobody can be expected to know the future, but current trends suggest that there is a growing desire to go beyond STEAM to nurture, practice and develop skills that robots, as machines, are not capable of [14], [15]. Among other things, this means being creative; able to think innovatively and independently.

To use the words of Stephan Turnipseed, executive vice president and chief strategy officer, Pitsco Education, “we must prepare children to be uniquely human in an increasing unhuman (and occasionally inhuman) world.” [16]

STEAM competencies are undoubtedly important for success in tomorrow’s world, but these are increasingly likely to be accompanied by more character-based life skills. ‘The future of jobs report’ by the World Economic Forum, includes an overview of skills that will be most sought after in the future [14]. That overview shows creativity has jumped from the tenth most sought-after skill in 2015, to the third most needed by 2020. Likewise, Emotional Intelligence, not even on listed in 2015, is now the sixth most sought-after.

To code is not only to learn that particular skill, but also to give children a wider set of competencies - this includes social belonging and values, moral intelligence, being creative and coming up with new ideas, nursing human emphatic abilities, and making ethical decisions. According to a recent study, the educational benefits of coding in the early years classroom are not just related to STEM skills such as math and problem solving, but also to verbal, causal and social reasoning [17].

---

**Top 10 skills**

*Source: Future of Jobs Report, World Economic Forum*

<table>
<thead>
<tr>
<th>2020</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Complex Problem Solving</td>
<td>1. Complex Problem Solving</td>
</tr>
<tr>
<td>2. Critical Thinking</td>
<td>2. Coordinating with Others</td>
</tr>
<tr>
<td>3. Creativity</td>
<td>3. People Management</td>
</tr>
<tr>
<td>4. People Management</td>
<td>4. Critical Thinking</td>
</tr>
<tr>
<td>5. Coordinating with Others</td>
<td>5. Negotiation</td>
</tr>
<tr>
<td>6. Emotional Intelligence</td>
<td>6. Quality Control</td>
</tr>
<tr>
<td>7. Judgment and Decision Making</td>
<td>7. Service Orientation</td>
</tr>
</tbody>
</table>
CODE IN THE CLASSROOM
Widen the walls

Considering the responsibility of formal education to prepare children not only for a career but also for a meaningful life, [15] coding and computational thinking competencies have earned their place in primary education. The question becomes how is this best delivered?

We believe in the principles of low-floors and highceilings; that things should be relatively easy to understand, but fairly hard to master, making them both accessible and engaging. Mitchel Resnick and his MIT team also stress the importance of ‘wide walls’ - allowing for a wide set of interests to motivate and apply knowledge [18]. A good way for coding to be delivered is to integrate it into existing subjects and relate it to what teachers already do in their classroom [19].

With the current pace of technological development, it is paramount that we, as Edtech suppliers, support teachers, by making products that are easy to set up and that are genuinely designed for a classroom environment. We must also provide training for teachers, along with curriculum inspiration and structured support that guides students to deeper engagement and higher-level thinking [19], [20].
Turn off the computers

Un-plugged teaching of computational thinking and coding is gaining recognition [4], and although it may seem counterintuitive to take screens out of the equation, there is plenty of evidence to support that this strengthens the learning process. With growing concerns about the amount of screen time children are exposed to and a cluttered and crammed marketplace of resources, it is also hard for teachers to determine which resources are the most effective.

Seymour Papert’s well-known work Mindstorms and his theory of constructionism, has firmly established the benefits of having ‘objects to think with’ [10]. Objects facilitate mental models by providing concrete representation to enhance the understanding of abstract concepts. In the early years of a child’s development, learning is all based on sensing and manipulating the world around them. It is only later they learn to think abstractly [21], [22]. Tangible learning tools are therefore essential in early years to accommodate children’s way of learning.

By removing the screen from the learning-equation in the early years’ classroom, children not only benefit from hands-on learning, but also from enhanced peer-to-peer interaction. This means that instead of having parallel lines of vision pointing at a screen, they have the opportunity to connect, share, talk, and inspire or be inspired in a more direct and intimate way with their peers.
Mistakes are good

Coding is one of the few areas where it is acceptable to make mistakes and this helps to create a learning environment that welcomes mistakes as a natural part of learning. A code rarely works the first time around, and one of the essentials of becoming a good programmer is the ability to identify and fix errors in the code. This process is called debugging and is an important skill in coding, and one of the processes in computational thinking [9], [13], [20].

In a society that is increasingly focused on perfection, coding is an opportunity to learn how to be comfortable with trial and error in an iterative process. When coding, children shouldn’t necessarily strive towards perfection but should be encouraged to experiment with various creative solutions.

Children should practice articulating and identifying bugs in the code, and should address suggestions on how to fix it. In coding, there is often more than one solution to a problem, which also gives children the benefit of comparisons and discussions on how and why one algorithm may be better than another.

Facing difficulties or challenges along the way when learning about coding is almost inevitable, but by making coding engaging and creating a learning culture where mistakes are allowed, children will be more willing and determined to try, try, and try again until they find a solution to the problem.
Keep it simple

It is our belief that coding should not be sugar coated but taught in its purest sense, in the same way as teaching the alphabet is taught in order to learn to read and write.

Coding should not be overly gamified or treated as a secondary element, for example by being introduced in a task that has greater focus on building a robot, or other mechanisms. The basics of coding concepts and coding terminology should be the primary focus of learning in the early stages. This does not mean that the learning cannot be playful, but the activity should not disguise or dilute the principle learning objective.

Coding must be relatable. Real life examples, and those closest to children’s existing interests help to make abstract concepts and terms more understandable. And these do not need to be STEM centric. Too often examples of coding in real world contexts are related to traditional functions such as building bridges, navigating ships, or coding machines to process tasks. There are many more ways to ensure that coding and computing resonates for all children. Examples of the use of code in real life can be anything from light installation to aesthetic decoration or interaction, to design of products and environments that influence surroundings and human wellbeing.

By starting with normal daily activities such as following a recipe, or planning a route on a map, children learn to grasp the basics of algorithms. They experience them as visible, tangible and logical sequences that can be manipulated and repurposed in new ways, opening up new possibilities.

Coding is used in all walks of life, and the real opportunity for our young learners is to fully understand its use, and to imagine how it can be used in the future; for jobs that don’t exist, for a world that they will master, where coding and computational thinking are essential skills.
References


