THE JOURNAL Computer Science Education

Quelle: https://thejournal.com/Articles/2018/08/07/Beyond-Point-and-Click.aspx

Beyond Point and Click: Real Coding for Students Across the Curriculum

If we need students to be learning coding now, yet the schools are not ready for it on many levels, where does that leave us?

- By **Ruth Reynard**
- 08/07/18

Anyone who has studied languages will know that no language that is currently being used ever stands still. Languages are always changing based on use; specifically, the meanings of words can change over time, based on changing contextual uses. The only languages that do not change are those no longer used, as they fossilize and remain static from the point at which they ceased being used. Additionally, specific "jargon" or descriptive words can be introduced into a language with uses of new technology, new discoveries, or even new human behaviors.

Such a word in English is "coding," as it relates to computer languages that have been used over time to create computer functions, operations and, more currently, interface designs. While most users do not really care to see the coding "source" page for each web page or application, it nevertheless exists, as, without it, the screen would not have an interface nor provide navigational options or functions. Therefore, the combination of the "front end" and the "back end" computer code is what provides our user options and the functions of each page. We do not even really think about it, however, as we are the users, not the creators or the designers, for the most part. It is important to realize, too, that like other languages, the languages of computer coding (programming) not only have a language system but also a logic. That is, coders must use the code systematically and logically within any given context of use. **Steve Jobs** referred to this when he stated:

"Everybody in this country should learn to program a computer because it teaches you how to think".

Background

Some of us are old enough to remember the first Internet pages on the Disc Operating System (DOS), without the graphical user interface (GUI) options that came later. With DOS, we had to focus on the coding to even have any kind of function or movement of objects. Then came the GUIs and HTML languages. Those of us who were early adopters in education began learning HTML in order to design our own web pages and upload content and provide any kind of interaction or online exchange with our students. So, for those of us who remember, we still regard "coding" as the computer language that directs the computer functions. Currently, however, the use of the word "coding" does not automatically refer to the computer languages (programming). It usually, in fact, refers to the interface design, organization and functionality.

We are now being told that younger students should be learning coding for their future, as with increased computerization, Artificial Intelligence (AI), simulation technology, holographic technology, and robotization, many jobs will require employees in various industries and work contexts to be able to code.

However, we should be clear that it is actually computer languages (programming), not only interface design and functionality, that is being referred to as user experience design.

Most educators are aware of the current and future focus on "coding" and are keenly interested in finding ways to encourage their students to learn how to code. Interestingly, however, there is a lack of clarity in understanding what is being referred to by "coding," even for educators. Many teachers also believe it refers to software manipulation, interface navigation and action-oriented functionality. While these may be useful creative skills to develop, they in no way address the actual process of the kinds of skills in logic and creativity which are needed for future employment readiness. The problem with the semantic shift is that it has evolved as a result of user interpretation rather than what employers are referring to as the kind of skills needed for future success. While computer science students and teachers realize coding to mean programming, most other users do not. This is problematic when so many students are using free simulation tools that do not develop skills in creative problem solving. **Burning Glass** recently stated:

Coding skills are in demand across a broad range of careers, not just for programmers. The ability not only to use but also to program software is often required of business people who work with data, of designers and marketers who create websites, of engineers who build products and technologies, and of scientists who conduct research.

Therefore, not only a select few should be able to program, but it will be necessary for individuals across various industries and professional contexts to be able to problem-solve and design and create coded solutions that are specifically designed and customized for direct use. More generally, students should not only learn to program because that is what the jobs of the future will require, but because programming helps students to learn and, as Steve Jobs has said, how to think.

Additionally, Marianne Stenger on **InformEd** states:

Although computer programming was once seen as a skill reserved for geeks and computer nerds, it's now regarded as an essential ability for 21st century learners and is becoming a key component of many curriculums, even in primary schools.

Stenger's article in InformEd states that in addition to the conventional ideas of programming being required for engineers, and computer experts of the future, there is an overall strengthening of logical and applied thinking that is developing as a result. Additionally, young children are benefiting in their literacy development (back to the language characteristics). As we have already discussed, computer languages are, in fact, language systems and do require literacy skills to learn and use the codes. In fact, linguists have for a long time referred to human languages as "codes." Increasing numbers of sources state the need for programming and use the words "programming" and "coding" interchangeably in their commentary. However, given the reality of what is being taught as coding in schools, the terms are not mutually understood.

Current Coding Uses

CodaKid explains:

Computer programming, otherwise known as coding, is currently offered in a small fraction of US K–12 schools. There has been a push to change this recently, as evidenced by several White House initiatives, the heavily publicized Hour of Code program and recent large-scale adoptions of hands on STEM programs such as Project Lead the Way. Serious challenges remain. Many schools find themselves ill-equipped to set up coding for schools' programs, citing reasons such as insufficient

human capital, out-of-date equipment, and high-speed internet issues. Setting up effective coding programs at schools can be challenging, and there as many issues to consider including curriculum selection, staffing, professional development and funding.

Therefore, due to lack of budgets and equipment and trained instructors, actual programming is not being taught (except in traditional computer subjects). Rather, as we have discussed, a generic "coding," which is not programming, is what is generally offered. Therefore, really, nothing is being created, only constructed based on preset options. Where is the real creative thinking and problem solving with these simple "point and click" simulation tools? Steve Jobs's comment about thinking in relation to learning coding is critical, as true programming provides an immersive learning experience that is unparalleled for integrating various skills and knowledge simultaneously.

A More Effective Approach

The kind of learning environments within which coding (programming) would be integrated and taught, however, themselves create problems beyond small budgets and insufficient teachers. That is, a tension exists between the challenges for schools to meet preset standards and outcomes and the reality of learning itself. The more we learn about learning and the individuality of motivation, approach and overall success, the more we are inclined to remove preset everything. Anyone who teaches knows about this tension and would truly enjoy the freedom to explore unconfined teaching and learning. The idea is not completely new, as we've been hearing from theorists already, over the years, just how hands-on and experiential learning are the way to go. **James Neill** (2005) writes:

In the 1920s/1930s, John Dewey became famous for pointing out that the authoritarian, strict, pre-ordained knowledge approach of modern traditional education was too concerned with delivering knowledge, and not enough with understanding students' actual experiences.

Neill also points out:

Thus, Dewey proposed that education be designed on the basis of a **theory of experience**. We must understand the nature of how humans have the experiences they do, in order to design effective education. In this respect, Dewey's theory of experience rested on two central tenets — continuity and interaction.

A Historical Perspective writes about Maria Montessori:

Dr. Maria Montessori is extremely important to the historical development of experiential education because she was the first person to bring it to a model. In fact, she can be considered one of the founding mothers of experiential education.

Therefore, the concept of experiential learning as an immersive experience for each student, based on their own learning abilities and preferences, has long been an ideal, yet not the reality in education, for the most part. While some schools, like Montessori schools and some private institutions have attempted to purse these ideals, yet the general education system and accreditation standards remain preset and non-negotiable. We have streamlined learning through standardized tests and systematized grading and rely on those, for the most part, to assess learning and organize learning. Social Constructivists and Critical Pedagogists have challenged these layers of preset organization for years.

Consistently, however, the challenge has been that the entire educational system would have to be reorganized to accommodate truly applied learning or, as we are referring to it now, "competency-based" education, rather than standardized education.

Or would it?

Perhaps there is a way to unify all efforts with the proper alignment in order to maximize student engagement with an across-the-discipline approach. A recent pilot running in a ND school district is challenging the preset grade levels and implementing an experiment in true competency-based education. **The Hechinger Report** (2018) explains:

In a lone building flanked by farmland, the Northern Cass School District is heading into year two of a three-year journey to abolish grade levels. By the fall of 2020, all Northern Cass students will plot their own academic courses to high school graduation, while sticking with same-age peers for things like gym class and field trips.

The goal is to stop tethering teaching to "seat time" — where students are grouped by age and taught at a uniform, semester pace — and instead adopt competency-based education, in which students progress through skills and concepts by demonstrating proficiency.

While other states have experimented with competency-based programs, what makes this different is the removal of grade levels. In other words, students can learn without being told "they're falling behind" or that "they're too advanced." Each student can truly learn at his/her own pace.

It seems that we're beginning to move more toward a form of education that is truly about learning and motivating each student to maximize their own potential without minimizing potential through standardization. My sense is, however, that we are not there yet; and while we need creative thinking, innovative problem solving and critical design in coding, there are larger issues as to why "point and click" remains.

Beyond Point and Click Approach

So how does this, then, relate to coding/programming for future job readiness? If we need students to be learning coding now, yet the schools are not ready for it on many levels, where does that leave us in becoming more about potential, possibilities and innovation?

With robotization, most "rote" jobs will be done by machines and not humans. Therefore, humans are needed in areas of innovation, creativity, problem solving and design.

Marc Cuban recently stated:

We're going to have a lot of displaced workers — the nature of work is changing. So, a new skill will become more in-demand than it ever has been: **creative thinking**.

Recently, I was approached by a group (**Anomaly Learning**) who have taken an interestingly different approach to coding instruction for K-12. Their connection with me was to ask for my input and content development for their teacher PD. I have written numerous articles over the years about how new and newer technology continues to change how we think, how we process information and, ultimately, how we should approach and design learning. **Anomaly Learning** has created a fully "out of the box" approach to actual programming education so that schools do not have to purchase equipment nor hire additional teachers. Their content is fully online. However, my task is to create training for teachers and administrators in how to look at learning in an integrated fashion and across all disciplines. That is, all students, regardless of their subject area preferences should be able to learn programming within a context of critical thinking, creative design and innovative application. In fact, I have termed the approach as "Integrated Project-Based Learning" as without removing anything standardized, an innovative and integrative approach to problem-based learning can situate or "immerse" students with a real context of learning and help them realize fully developed and creatively designed solutions to real-world problems. Teachers must facilitate this process, providing guidance and support as students learn to code.

The Anomaly Learning motto is "Beyond point and click," and they state:

The Anomaly Learning platform teaches students how to be critical thinkers and problem solvers while developing skills in technology and coding. Our unique methodology and framework weaves together threads of thinking between the STREAM disciplines (Science, Technology, Reading, English, Arts, and Math). We do this in a way that provides teachers with a solution they can directly plug into their classroom for an immersive guided learning experience for students that provides the full spectrum of teaching through to assessment and reporting.

The founder of this group, Josh Lomelino, says that the motto evolved when listening to school children describe their "coding" experiences. They described basic and generic "point and click" environments where functionality and navigation were preset, and choices could be made only in terms of preset parameters of the software program. Anomaly's approach is to maintain the creative and innovative motivation that younger generations now have in using gaming software, for example, but develop it in a cross-discipline, fully STREAMed environment. Thus, not only programming skills are developed but also the kinds of logical, creative and applied thinking that will be critical for future success both in learning and later in professional employment.

While there remains a confusion in the use of the word "coding" in schools today, owing to a variety of obstacles and challenges, there are some advances being made by educational groups and innovative learning companies to attempt to work around some of these obstacles and to provide the real coding education that is required for the future success of our students. Anomaly Learning is one of those innovators. There are numerous challenges currently facing all educators and society is rapidly changing due to the impact of new and newer technology. In a **2009 article** I wrote, I explained it this way:

It would seem, then, that as well as preparing students with knowledge about life, it is important for schools to begin the process of developing students with skills for life and work.

Marc Cuban would agree that creative thinking, problem solving, and innovative design should be skills developed in every student and not only in those students who are in conventional computer courses, digital design courses or mathematics. True coding education must be fully integrated, cross-discipline and resulting in creatively designed and problem-solving innovations for real world success. This is how we can help transition students from being consumers to being producers and "makers."

References

Anomaly Learning (2018). A Division of Anomaly Studios.
Bamboo Innovator (2013). Literacy in Coding is an advantage in this technology-driven economy.
Burning Glass Technologies (2018). Beyond Point and Click: The Expanding Demand for Coding Skills.
Dodge, D. (2016). **Coding for Schools: The Ultimate Guide for Teachers and Administrators**. CodaKid.com. Coding and Game Design for Kids.
Dewey, J. (1938/1997). *Experience and education*. Macmillan.
Neill, J. (2005). **John Dewey, the Modern Father of Experiential Education.** Outdoor Education and Research Center. Wilderdom.
Reynard, (2009). **Web 2.0 Tools and K-12 Challenges**. T.H.E. Journal.
Stenger, M. (2017). **Coding In Education: Why It's Important & How It's Being Implemented.** InformEd.

About the Author

Ruth Reynard, Ph.D., is a higher education consultant specializing in faculty development and instructional design. She can be reached at ***ruthreynard@gmail.com***.