An Exploration of Artistic and Technological Symmetry

Artemis Papert

Abstract

In this interview, biologist, artist, and shiatsu healer Artemis Papert describes how computational thinking can help people organize their thoughts in a more formal way. She discusses TurtleArt, a software that allows both children and adults to create two-dimensional static art images using geometry and coding as a medium. TurtleArt not only bridges the worlds of math and art, but is also easy to learn. She concludes by reading an excerpt from the article, "Teaching Children Thinking"—written in 1971 by her father Seymour Papert—as a still relevant starting point for where technology is heading.

You're known as an artist and a healer, can you tell us how your career unfolded and how these two areas of talent emerged?

I've always liked drawing and painting since ever I can remember. I might have gone to art school had I been born in a different family, but I was born in a family of academics. So, it felt like academia was the thing to do. My mother worked with Jean Piaget and my father was a mathematician who also came to work with Jean Piaget because Piaget wanted to understand how children understand numbers. And he thought it would be a good thing to have mathematicians in his team—so that's how my parents met and how I came along. I always wanted to become a biologist. Back then, I was young and naïve and wanted to save the Antarctica, and the Rain Forest … My career as a biologist didn't quite evolve as I had hoped it would. I did my master's degree and PhD in biology and had a few post-doctoral positions. However, my career was heading more and more towards molecular ecology, rather than ecological research dealing with whole organisms.

While I was working in biology I was thinking, "It would be nice to expand and also do something else." I was living in Scotland at the time and discovered shiatsu, which is "acupuncture without needles" and also based on Traditional Chinese Medicine. The more I was doing shiatsu, the more I was getting enthusiastic about it and more and more drawn into it, to the point where I decided to do shiatsu and stop doing biology. Of course, in shiatsu you deal with human bodies, deal with human beings—deal with the whole organism. So that's back to me wanting to be an ecologist—I'm not dealing with the ecosystem of the rainforest or the Antarctica, but I'm dealing with the ecosystem of the human body.

Shiatsu is a healing art; there is an art in there. Yes, of course you need to know your theory, you need to know what you're doing, but with every person, in every session, it's different. The Dutch artist Vincent van Gogh said, "You need to know your technique so well that you can forget it." And it's the same thing with shiatsu. After my first few hours of shiatsu classes I was thinking: "What do I do?

Where do I press? What's happening here and there?" And after a while you just do it because you don't think about it. You take a paper and pencil and you write something—you don't think, "I'm holding a pencil in my hand and this is a sheet of paper"—you just write. It's the same with art after a while: you don't think about the medium you're using—it's just a medium. You just think about the creation you are creating or the healing you are giving.

There is a growing emphasis and a need to develop computer programming skills among students in elementary and high schools. What are your thoughts on this?

Computers are a very powerful tool if used in the proper way. They can be used as a substitute for television or as a tool to help you think. The big word these days is "computational thinking," which nobody has quite described very precisely. One definition of computational thinking that I quite like is from Stephen Wolfram (2016), who said, "It's a way of expressing yourself in a way that a computer will understand." And it's using "algorithms," which is also a big word and a way of theoretically saying something. It's a way to learn some thinking skills and organize your thinking in a more formal way.

An important thing when you program is debugging. You write a program and ask yourself: "Did it do what I wanted it to do?" If yes, good. If not, "How do I fix it so that it can be doing what I want it to do?" It's also a way of learning skills that you can apply to life. Things in life don't always go as expected, so what do you do? How do you "debug" them?

The other question about computer and programming skills is: Why are we teaching that to children? Yes, it's an important skill to have, but why? It's important to understand what you can do with a computer because it allows you to do some very powerful projects and things that you would not otherwise be able to do as easily—or not all maybe, depending on what it is. It's not necessarily to become a professional programmer; it's not a trade skill. Some people will become professional programmers later in life, but that's not the goal—or that should not be the only goal. A little bit of programming goes a long way.

TurtleArt lets you make images with your computer. The Turtle follows a sequence of commands. You specify the sequence by snapping together puzzle like blocks. The blocks can tell the turtle to draw lines and arcs, draw in different colors, go to a specific place on the screen, etc. There are also blocks that let you repeat or name sequences. Other blocks perform logical operations.

The sequence of blocks as a program that describes an image. This kind of programming is inspired by the LOGO programming language. It was designed to be easy enough for children and yet powerful enough for people of all ages. TurtleArt is focused on making images while allowing you to explore geometry and programming.

Fig. 1: Description of TurtleArt (Source: https://turtleart.org/)

TurtleArt was developed by your partner Brian Silverman and his work partner Paula Bontá. It has encouraged in many settings around the world constructionist beliefs about learning—actually initiated by your father, Seymour Papert. Can you talk about what TurtleArt is and the potential it has for use across subject areas and with all levels of students?

TurtleArt is based on the Logo programming language co-invented by my father and his colleagues Cynthia Solomon and Wally Feurzeig. That was in the 60s, early 70s. The idea was to have a programming language that was easy enough for even very young children to use and powerful enough that even a professional programmer could use. The idea behind TurtleArt was to go back to Turtle Geometry and do one thing and do it well. It's about making two-dimensional static art images using geometry, using coding as a medium. It's a very simple programming language. The vocabulary is only 50 words. You go very quickly from learning it to learning with it.

After that, there is the artistic expression, which takes hours and hours and is an artistic exploration. And finding your voice takes the time it will take. It's about how you express your voice with this specific medium you have. Because it's using code as a medium, the people that are more math oriented will do some artwork, but they think they're doing math and programming. While the people that are art oriented are doing art, but in the process they do some programming. It's a way of getting both coming together. One thing that Seymour said in his book, "Mindstorms" (1980), is that there is this language-oriented/math-oriented dichotomy. And it's an artificial dichotomy. Yes, some things come more easily to some people, some people pick up languages more easily, some people get theoretical mathematical concepts more easily, but it doesn't mean that because you're good at English you're necessarily going to be stupid in programming and math—or vice versa.

One image that my father was using was the image of "Mathland." He said, "Someone who is born in a French-speaking family will speak French." There is no story like: "Oh, I'm not good at French." If you are born in an English-speaking family, you speak English. There is no question: "I'm not good at languages; I'm not good at English." Everybody will speak the language of the family they are born into. Seymour said for math: "Why don't we have a place where you learn math because everybody speaks math. That's the language that's spoken." So, that's how the idea of the Logo programming language came in. And TurtleArt could be a subset of Mathland—it is a Mathland for the people from "Artland." You could get into art by using TurtleArt without thinking you are doing art. Or you can be into doing art using coding without thinking you are doing coding.

One way where TurtleArt will fit with all levels of students is that it's simple enough that younger children can use it. And it's powerful enough also that adults can use it. In the classroom, it's a good way of getting the art teacher and the math teacher to collaborate on some projects.

What have you found are the most appealing aspects of TurtleArt to novice students? Can you give a couple of examples that you've witnessed?

One of the exciting things about TurtleArt is you can get started very easily and get a result in the first 15 minutes. You are probably not going to get the most amazing piece of artwork. There are exceptions, of course, but it will take maybe a few hours before you have something artistically really interesting. A first result happening quickly can be encouraging for children. In art, there's nothing that's right or wrong—it's about your own personal taste: "Do I like it or not?"

Fig. 2: Sea Moon, image made with TurtleArt – Designed by Artemis Papert

What kinds of suggestions do you give to teachers who want to introduce and become involved in TurtleArt in their classrooms?

First thing: get the software at www.turtleart.org (click on "email us" and request a download link) or at the App Store if you have an iPad. Then play with it yourself. One thing that Seymour said is: "Children learn best by doing personally meaningful projects." Of course, it's the case for adults too. If you want your students in your classroom to get excited about something, but you find the thing boring, how much of a chance of success do you have there? My guess would be "not impossible," but "not a lot." If you tell your students, "Look, I discovered this great thing that I'm really excited about! Let's play with it together." Then the chance of getting more kids involved is going to be better. Don't be afraid of not knowing everything—it's okay to discover with your students. Be a few steps ahead of them, but you don't need to know everything. It's important also for students to see that the adult doesn't know everything. It's a way of showing them how you discover things, how you learn things.

In the best of all worlds, where do you think digital technologies are headed and what advice do you have for educators?

To answer that question, I want to go back to something that was written in 1971 by Seymour, entitled "Teaching Children Thinking":

> The phrase "technology and education" usually means inventing new gadgets to teach the same old stuff in a thinly disguised version of the same old way. Moreover, if the gadgets are computers, the same old teaching becomes incredibly more expensive and biased towards its dullest parts, namely the kind of rote learning in which measurable results can be obtained by treating the children like pigeons in a Skinner box.

> The purpose of this essay is to present a grander vision of an educational system in which technology is used not in the form of machines for processing children but as something the child himself will learn to manipulate, to extend, to apply to projects, thereby gaining a greater and more articulate mastery of the world, a sense of the power of applied knowledge and a self-confidently realistic image of himself as an intellectual agent. Stated more simply, I believe with Dewey, Montessori, and Piaget that children learn by doing and by thinking about what they do. And so the fundamental ingredients of educational innovation must be better things to do and better ways to think about oneself doing these things.
>
> I claim that computation is by far the richest known source of these ingredients. We can give children unprecedented power to invent and carry out exciting projects by providing them with access to computers … (Papert, 1971)

In 1971, the idea of a child using a computer was like science fiction. There were no laptops, iPads, or smartphones. It was like: "What do you mean a child can use a computer?" Now, every child has some electronic device of some kind or another. And how is that empowering them? If you use them as a babysitter and have your two-year-old just watching YouTube videos all the time, it's not going to help his or her intellectual development. If you are using the electronic device to help the child do something creative, and think about what they are doing, then it will help intellectually. Programming is a very powerful activity because there is no way of doing coding without thinking about: "What do I want the computer to do?"

Alan Kay said, "The best way to predict the future is to invent it." If we want computers not to be used only to do in a more expensive way the same old stuff, then one thing to invent is ways that allow to use computers in a more intellectually engaging way. You need to have the tools that will allow you to get engaged. There are people who invented tools that allow you to go deeper and think deeper about what you're doing. A lot of the software you see out there for the apps for your phone, for your iPad, whatever device you have, is just doing the same old things just using a new more expensive technology. I hope that the pendulum will swing back at one point to making more applications available, which people of all ages, not only children, will be able to use while getting more intellectually involved.

References

Papert, S. (1971). Teaching children thinking. *MIT AI Lab Memos* No. 247. Retrieved from http://www.citejournal.org/volume-5/issue-3-05/seminal-articles/teaching-children-thinking/

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books, Inc.

Wolfram, S. (2016, September 7). How to teach computational thinking. [Blog post]. Retrieved from http://blog.stephenwolfram.com/2016/09/how-to-teach-computational-thinking/

Artemis Papert is an artist creating art in both traditional and digital media. She has led TurtleArt workshops for a wide variety of groups in many countries. After a first career as a research biologist, she retrained as a Shiatsu therapist. Artemis has been practising shiatsu since 2003 and is training to become a Jungian psychoanalyst. Artemis is a volunteer with the Montreal organizations Spa de la Rue and Maison Monbourquette. Artemis now lives in Montreal. She previously lived in Switzerland, The Netherlands, and Scotland.