DavidCrusoe2

* In the Foreword to Mindstorms, Papert talks about his childhood fascination with gears. Describe an object from your childhood, and explain why it was important to you (then and later).

Surprise -- the object(s) of my childhood fascination were Legos! While this revelation might come as no great shock, it's important nonetheless. The Lego brick was, to me, the wonderful building block for constructing entirely new worlds, worlds far beyond what Zach the Lego Maniac ever created in his commercials.

In addition to the hands-on modeling of dream and vision, Legos allowed (and still allow) for something quite rare: the transference of those ideas to others. I could easily manipulate those little bricks to create a rich environment (so what if it was below a ping-pong table?) to not only engage my own dreams, but those of my friends and co-constructors as well.

My environments had a history, too: entire villages could grow and change, in effect they themselves growing and maturing as my understanding of efficient building and wild invention itself matured.

And, my play with Legos was as much about building worlds as it was testing design skills. A particular contest was to build a vehicle that, when raced down a staircase, would land on its wheels. More often than not, those (poor!) Lego vehicles exploded in a shower of two-bit-connectors and black-and-red-orange wheels. But those designs that withstood this race became the model-object for future vehicles, and in this way, the process was transformed my thinking about structure and design.

In looking back, my play with Legos was instrumental in creating within me the type of open-ended creativity that I so relish now. That I engaged in this manner of play during my formative years has affected who I am, and more importantly, my beliefs about, and approaches to, learning. So, anyone up for playing with Mindstorms?

---

* What idea (or passage) in Mindstorms was most provocative, intriguing, or surprising for you?

"Many children are held back... because they have of model of learning in which you have either "got it" or "got it wrong." ... The question to ask about the program is not whether it is right or wrong, but if it is fixable. If this way of looking at intellectual products were generalized to how the larger culture things about knowledge and its acquisition, we might all be less intimidated..." (23)

In considering what of Papert's visions provide provocative visions for future learning, one idea that intrigues me is the idea of what role "debugging" (a form of metacognition) could play in learning.

Certainly for the greater part of my formal education, this metacognitive process had not been taught or reinforced. At no point did we question discrepancies in our history books, or did we search for reasons that history may have been interpreted differently by different people. I was right, or more often, I was wrong. Papert is definitely correct in his implicit assertation that a focus on "got it" or "got it wrong" is detrimental to the learner.

In a broader context, I wonder how this metamodel can be applied to classroom teaching and learning. Let's deconstruct. To properly understand the information we seek in construction new knowledge, we must parse volumes of correct, partially correct or incorrect information to support our hypothesis. So, on one hand, it could be useful to teach students (from an early age) to verify and validate the sources they incorporate into their learning. In the programming environment, this might be applied to following up on code that one found on the internet; is it even correct, to begin with?

However, "debugging" is also parsing what one creates during its creation; a formative evaluation of one's own work. For that, instead of becoming frustrated and "throwing away the program" (as Papert notices in school children) it's more important to step back to examine the merits and weaknesses of one's work. If a program doesn't run, one can ask what part of it does work correctly, and how can the error be contained. If a paper's thesis lacks evidence, or evidence doesn't strengthen a conclusion, why? What's wrong, but more importantly, what's correct, and at what point did the student's errors occur? And most importantly, how can we shift the burden of these questions to the learner so that debugging happens during the learning process, rather than letting the teacher ask - and answer - after the fact?

---

* Mindstorms was written nearly 25 years ago. Which ideas in the book stand the test of time? Which ones don't?

I'm intrigued by Paper's assertation that computers would become vehicles that children could program, rather than vehicles that would program the child. Sadly, it seems that all too often the child (well, the user) has lost out in this equation. Papert may have overlooked the power of commerce to forge user-friendly computing environments.

Certainly to some degree our innovations (programs and operating systems, really) have ensured that an incredibly broad user base can manipulate data (text, video, etc) on their computers. There isn't an insurmountable coding language to learn prior to, say, editing a picture in Photoshop. One certainly doesn't need to learn the manipulative algorithms onesself. (On the other hand, this simplification has (perhaps) relagated the user-as-programmer aspect of computer use to those who truly want to program in "coding languages.")

On the other, I wonder how it might be possible to include (perhaps as part of a nifty new program) user-as-programmer aspects into new types of software. For example, might it be possible to create a "great" web browser that lets the user design his or her browsing experience within an object-based browser program? What conveniences might be built atop a disguised, simplified programming environment that "everybody can learn?" I fear that unless this question is asked, answered and acted upon, Papert's vision will become less and less real as commerce, entertainment and computing continue their merger.