Programming is Cool Again

The Beauty and Joy of Computing

Brian Harvey

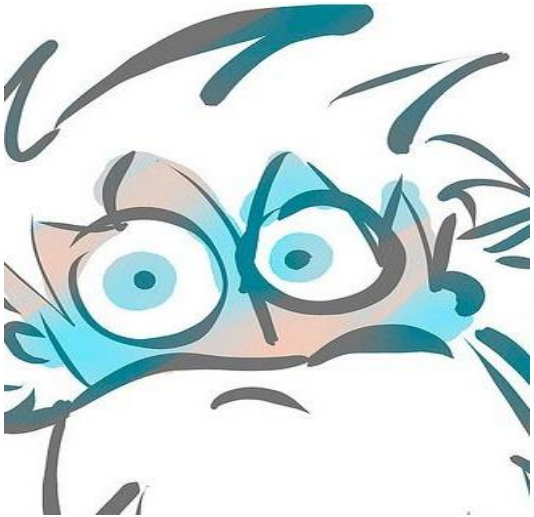
This material is based upon work supported by the National Science Foundation under Grants No. 1138596 and 1441075. Opinions are those of the author only.

EDC Learning transforms lives.

Berkeley UNIVERSITY OF CALIFORNIA

http://tinyurl.com/prog-cool

The Old Conventional Wisdom



- “Girls and minorities are seriously underrepresented in computer science.” **(**True!**)**
- “That’s because programming is thought to be nerdy, isolating, and irrelevant.” **(**Not any more!**)**
- “Therefore, to attract girls and minorities we have to define computer science in a way that deemphasizes programming.” *(NO!)*

CS Principles

Computational Thinking Practices

- Connecting Computing
- Creating Computational Artifacts
  (does this mean programs?**)**
- Abstracting
- Analyzing Problems and Artifacts
- Communicating
- Collaborating

Big Ideas

- Creativity
- Abstraction
- Data and Information
- Algorithm
- Programming
- The Internet
- Global Impact

Things have changed!

girls who CODE

Programming 4 Girls

Encouraging girls to explore computer science

black girls CODE

imagine. build. create.

GIRLS TEACHING GIRLS TO CODE

HOUR OF CODE

Made w/ Code Google

SCRATCH *uses* visual metaphors *to teach computer science ideas*

This is a *loop.* The arrow pointing back to the top is a subtle hint.

This *event* block can be used only at the top of a script.

This hexagonal yes-or-no question block...

... fits into this hexagonal slot in the IF block.

uses visual metaphors to enable learning of more advanced ideas

For example, the idea of *function as data* is considered difficult partly because it looks mysterious and painful in text-based programming languages:

```
>>> [round, (lambda x: x+3), len]
[<built-in function round>, <function <lambda> at 0x1005ed938>, <built-in function len>]
```

```
STk> (list round (lambda (x) (+ x 3)) length)
(#[closure arglist=(x) 278e30] #[closure arglist=(x) 2e41f4] #[subr length])
```

uses visual metaphors to enable learning of more advanced ideas

But in a graphics language, if you already know that 

is a function, and 

is a list, then this is much easier to interpret even without teaching:

The Moral: Programming is...

- ...motivating! **(**Programmers were the first “makers.”**)**
- ...at the *heart* of computer science. (All that stuff about algorithms, data, and abstraction is to help us program better.)
- ...no longer intimidating, especially in visual languages.
- ...valuable in every subject. (When kids write *programs* about fractions, or about parts of speech, or about the Trail of Tears, they have to understand the topic before they can write a program about it.)
- ...empowering, especially for generally-disenfranchised students, because the locus of control shifts to the learner.

The Moral: Programming is...

- ...motivating! (Programmers were the first “makers.”)
- ...at the *heart* of computer science. **(**All that stuff about algorithms, data, and abstraction is to help us program better.**)**
- ...no longer intimidating, especially in visual languages.
- ...valuable in every subject. (When kids write *programs* about fractions, or about parts of speech, or about the Trail of Tears, they have to understand the topic before they can write a program about it.)
- ...empowering, especially for generally-disenfranchised students, because the locus of control shifts to the learner.

The Moral: Programming is...

- ...motivating! (Programmers were the first “makers.”)
- ...at the *heart* of computer science. (All that stuff about algorithms, data, and abstraction is to help us program better.)
- ...no longer intimidating, especially in visual languages.
- ...valuable in every subject. (When kids write *programs* about fractions, or about parts of speech, or about the Trail of Tears, they have to understand the topic before they can write a program about it.)
- ...empowering, especially for generally-disenfranchised students, because the locus of control shifts to the learner.

The Moral: Programming is...

- ...motivating! (Programmers were the first “makers.”)
- ...at the *heart* of computer science. (All that stuff about algorithms, data, and abstraction is to help us program better.)
- ...no longer intimidating, especially in visual languages.
- ...valuable in every subject. **(**When kids write *programs* about fractions, or about parts of speech, or about the Trail of Tears, they have to understand the topic before they can write a program about it.**)**
- ...empowering, especially for generally-disenfranchised students, because the locus of control shifts to the learner.

The Moral: Programming is...

- ...motivating! (Programmers were the first “makers.”)
- ...at the *heart* of computer science. (All that stuff about algorithms, data, and abstraction is to help us program better.)
- ...no longer intimidating, especially in visual languages.
- ...valuable in every subject. (When kids write *programs* about fractions, or about parts of speech, or about the Trail of Tears, they have to understand the topic before they can write a program about it.)
- ...empowering, especially for generally-disenfranchised students, because the locus of control shifts to the learner.

(But there's lots more to learning than just STEM or computers.)

- You know this already, but there's a lot wrong with schools that has nothing to do with technology:
  - boring curriculum
  - jumping through hoops
  - high stakes testing
  - etc.

  These issues can be addressed only by changing how the institution of school views kids: as real people now, not as future adults. **(**This is not an attack on teachers individually.**)**
- Computers can help *because* they don't impose a curriculum; the computer rather than the teacher is the judge of success; the programming culture *expects and cherishes* bugs **(**imperfect first attempts**)**.
- Let's not get in the way with *too much* curriculum.

http://tinyurl.com/prog-cool

(But there's lots more to learning than just STEM or computers.)

- You know this already, but there's a lot wrong with schools that has nothing to do with technology:
  - boring curriculum
  - jumping through hoops
  - high stakes testing
  - etc.

  These issues can be addressed only by changing how the institution of school views kids: as real people now, not as future adults. **(**This is not an attack on teachers individually.**)**
- Computers can help *because* they don't impose a curriculum; the computer rather than the teacher is the judge of success; the programming culture *expects and cherishes* bugs **(**imperfect first attempts**)**.
- Let's not get in the way with *too much* curriculum.

http://tinyurl.com/prog-cool

(But there's lots more to learning than just STEM or computers.)

- You know this already, but there's a lot wrong with schools that has nothing to do with technology:
  - boring curriculum
  - jumping through hoops
  - high stakes testing
  - etc.

  These issues can be addressed only by changing how the institution of school views kids: as real people now, not as future adults. **(**This is not an attack on teachers individually.**)**
- Computers can help *because* they don't impose a curriculum; the computer rather than the teacher is the judge of success; the programming culture *expects and cherishes* bugs **(**imperfect first attempts**)**.
- Let's not get in the way with *too much* curriculum.

http://tinyurl.com/prog-cool