Code Micro-Learning: Learn to Code in Six Second Chunks!

Philip J. Guo
University of Rochester
Rochester, NY, USA
philip@pgbovine.net

Productivity Decomposed: Getting Big Things Done with Little Microtasks.
Workshop at CHI 2016, May 7–12, 2016, San Jose, California, USA.

Abstract

I propose to help people learn to code in six second chunks by having them watch short looping video clips and interact with flashcards that are auto-generated from a log of each learner's own past coding activities.

Computer Programming is Cognitively-Demanding and Time-Consuming to Learn

Computer programming is a cognitively demanding skill that takes thousands of hours of deliberate practice to learn well. Novices suffer from hundreds of common misconceptions about how their code works, and newly gained knowledge is often brittle and easily forgotten [3]. The conventional wisdom here is that learning programming requires large uninterrupted blocks of time. However, can ideas from micro-tasking and self-sourcing help people learn to code more efficiently in bite-sized chunks during waiting and idle times?

Learn to Code in Six Second Chunks!

Inspired by related work in micro-learning of human languages [1,2], I propose to design systems that help people improve their programming skills by watching six second videos in a repeating loop. Why six seconds? Because that is the length of clips on the popular social video service *Vine*. The creators of Vine tuned that length to maximize engagement [4]. Viewers find Vine

video clips incredibly addicting, especially on their smartphones. Can a system automatically create coding-related Vine videos for learners to review? Can we make micro-learning as addictive as browsing Vine?

The Key Idea: Record and Replay the Learner's Own Past Coding Activities

Manually-created Vine videos are an adequate baseline for comparison, but my hunch is that recording each learner's *own* coding activities when they are, say, solving assignments or debugging code in their IDE, can be much more pedagogically effective. This approach provides *highly contextualized* videos that are personally significant for the learner. Watching these videos will give them opportunities for *self-reflection* on what worked and did not work in their past trials.

When should such recording occur? One idea is to record everything and then extract six-second clips of *critical incidents* such as fixing an error, thrashing back-and-forth in frustration, and triumphs like new test cases passing after a prolonged period of struggle.

What should learners do while watching these Vine-like video clips? Watching passively may stimulate self-reflection and metacognition. But to be more engaging, a system can automatically generate contextually-relevant reflection questions based on analyzing the code featured in each clip, or show related clips based on similar code that the learner has edited in the past. Doing so can trigger their associative memory.

Automatically Create Dynamic Flashcards

Flashcard learning techniques such as *spaced repetition* are known to be effective for retaining newly acquired knowledge. However, creating high-quality flashcards is a tedious manual procedure that often requires an expert. Can a system automatically create coding-related flashcards based on the learner's own recorded activities? These flashcards can complement or be embedded within the aforementioned Vine-like videos.

One way to automatically create these flashcards is to remove blocks of code from a video and then ask the learner to fill in the blanks. Another way is to rearrange fragments of code (called *Parsons Problems*) so that the learner needs to drag the fragments back into the proper locations. Both kinds of interactions can engage learners more than watching videos, and can also be done in a few seconds on a phone during waiting times.

Relevant Author Background

Philip Guo is an assistant professor at the University of Rochester. He builds systems to enable millions of people to learn programming online in a scalable way. He has also worked on micro-learning for human languages [1]. This paper unifies those two threads: learning to code and micro-learning. Some ideas were co-developed with Emy Lin, an undergraduate student.

References

1. Carrie J. Cai, Philip J. Guo, James Glass, Robert C. Miller. Wait-Learning: Leveraging Wait Time for Second Language Education. CHI 2015.
2. Darren Edge, Elly Searle, Kevin Chiu, Jing Zhao, James A. Landay. MicroMandarin: Mobile Language Learning in Context. CHI 2011.
3. Juha Sorva. Visual Program Simulation in Introductory Programming Education. Ph.D. Dissertation. Aalto University, May 2012.
4. Laura Sydell. How Vine Settled On 6 Seconds. *NPR All Tech Considered*. August 20, 2013.