Comparing Text-based, Blocks-based, and Hybrid Blocks/Text Programming Tools

David Weintrop
Northwestern University
2120 Campus Dr. Suite 332
Evanston, IL, USA 60208
dweintrop@u.northwestern.edu

ABSTRACT
This dissertation investigates the comparative affordances and drawbacks of blocks-based, text-based, and hybrid blocks/text introductory programming tools. Blocks-based programming environments are growing in popularity and are increasingly being used in formal introductory programming contexts. To date, much of the work evaluating such tools has focused on their effectiveness in out-of-school contexts and emphasized engagement and attitudinal measures over content mastery. Given their growing presence in classrooms, it is important to understand the benefits and limitations of the use of the blocks-based programming approach in formal learning contexts relative to text-based or hybrid blocks/text alternatives. This dissertation will carry out a quasi-experimental study in high school computer science classrooms to answer questions related to the impact of blocks-based, text-based, and hybrid blocks/text introductory tools, assess the suitability of such tools for preparing students for future computer science learning opportunities, and explore the design space between blocks-based and text-based programming. The goal of this work is to better understand the tools we are using to introduce today's learners to computer science and lay the foundation for creating the tools of tomorrow.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education.*

Keywords
Blocks-based Programming; Introductory Programming Tools; High School Computer Science Education

1. PROGRAM CONTEXT
I am currently a fifth year PhD candidate in the Learning Sciences program at Northwestern University. Northwestern University's Learning Sciences program was the first of its kind and brings together cognition, design, and socio-cultural factors to study how learning happens in the real world. I have defended my dissertation proposal and, this past fall, I conducted a pilot of my dissertation study. I have done some analysis of the pilot data and am in the process of preparing the next iteration of my study, which will take place at the start of the upcoming school year.

ICER '15, August 09-13, 2015, Omaha, NE, USA
ACM 978-1-4503-3630-7/15/08.
http://dx.doi.org/10.1145/2787622.2787752

2. CONTEXT AND MOTIVATION
The ability to express ideas in a computationally meaningful way is becoming a critical skill for students to master in our increasingly digital world. Bringing programming into K-12 education is a critical step for introducing learners to this fundamental skill. A growing number of K-12 computer science classrooms are using blocks-based environments to introduce students to programming. These tools leverage a primitives-as-puzzle-pieces metaphor and support drag-and-drop composition, allowing learners to assemble functioning programs using only a mouse by snapping together instructions. The use of this programming modality has become a prominent feature of many introductory computer science curricula and programming interventions targeted at K-12 students. Notably, national curricular efforts including Exploring Computer Science, the CS Principles project, and Code.org's curricular materials all utilize blocks-based tools to introduce students to programming.

Despite its growing popularity and widespread use, relatively little work to date has focused on the conceptual and affective benefits of using blocks-based tools in formal educational contexts. Open questions remain about the effectiveness of the approach for helping students learn basic programming concepts and whether or not gains made in introductory environments, be they blocks-based or textual, effectively prepare students for future computer science learning opportunities. Further, it is unclear what the strengths and weaknesses of block-based programming tools are compared to isomorphic text-based alternatives. Given the number of initiatives being undertaken to bring programming, and computer science more broadly, into high school classrooms, it is essential that we understand the affordances and drawbacks of the tools we are using to introduce a generation of learners to the field. The goal of this dissertation is to shed light on these questions in order to improve curricular and design efforts that are shaping contemporary high school computer science education and to better inform teachers on how to make the most of the tools they are using.

3. BACKGROUND & RELATED WORK
"The tools we use have a profound (and devious!) influence on our thinking habits, and, therefore, on our thinking abilities." [3]

A growing body of literature is investigating the effects of the blocks-based programming approach. Notable work has been done looking at Scratch with younger learners (e.g., [5]) and Alice with university students (e.g., [8]), with relatively little work focusing on high school aged students. A number of smaller studies have done comparative work looking at textual vs. blocks-

based tools side-by-side, such as Lewis' [4] study comparing Scratch and Logo, which found students who worked with text-based Logo were more confident, while students in the Scratch condition performed slightly better on some of the content areas covered. Ben-Ari and colleagues have conducted a number of studies looking at the suitability of Scratch as serving as the primary introductory programming language finding both strengths and drawbacks to the approach [6, 7]. In a recently published paper, they found that students who learned Scratch in middle school more quickly grasped concepts in text-based languages when they reached high school (although they did not perform better on content assessments) [1]. Another thread of work has looked at how learning with introductory blocks-based tools transfers to conventional text-based tools with mixed results [2, 9]. Work on these questions is growing, but large gaps in the literature remain, this study will address some of those gaps.

4. STATEMENT OF THESIS/PROBLEM

This dissertation seeks to answer three sets of interrelated research questions. The first set pertains to the effects of programming modality (blocks-based vs. text-based) on students' learning experience. Specifically, what is the relationship between the programming modality used and learners' understandings of programming concepts? What programming practices do learners develop when working in different modalities? And how does the modality affect students' perceptions of programming with respect to utility, authenticity, and enjoyment? The second set of questions look at the effectiveness of introductory programming tools for preparing students for future computer science learning opportunities. Namely, how do understandings and practices developed while working in introductory programming environments support or hinder the transition to the text-based programming languages used in non-introductory computer science courses? Our final research question investigates the design of introductory learning environments. Can we design hybrid introductory programming environments that blend the strengths of blocks-based and text-based programming to effectively introduce novices to programming and computer science more broadly? All three sets of questions are designed to be comparative as we are interested into how aspects of learning differ across text-based, blocks-based, and hybrid blocks/text introductory environments.

5. RESEARCH GOALS & METHODS

The goal of this research is to understand the affordances and drawbacks of different programming modalities in formal high school computer science contexts. To answer the stated research questions, a quasi-experimental, mixed-method study will be used. The study follows three sections of an introductory programming class for the first 15 weeks of the school year at an urban public high school. For the first five weeks, students will use either a blocks-based, text-based, or hybrid blocks/text introductory programming environments before transitioning to Java for the remainder of the study. Students are randomly assigned to the three sections and will follow the same curriculum regardless of the environment they use. Pre/mid/post content and attitudinal assessments will be administered during the study. We will also carry out one-on-one cognitive interviews with students from all three conditions as well as conduct classroom observations and record all student-authored programs. By gathering this set of data and studying students across the three programming environments, as well as following them as they transition to Java, we will be able to comparatively evaluate the environments and answer the stated research questions.

6. DISSERTATION STATUS

I have completed a pilot study where I followed 90 students in 3 sections of a computer science classroom for the first 10 weeks of the school year. I observed students as they spent five weeks working in three distinct, customized versions of the Snap! environment and then followed them as they transitioned to Java. The customized Snap! environments added features like the ability to see text-based versions of blocks-based scripts, and define new block behaviors in JavaScript. I conducted 27 interviews, administered pre/mid/post attitudinal and content assessments, and collected over 75,000 student-authored programs. As part of the pilot study, I created all necessary curricular and assessment materials and designed interview protocols and automated data collection procedures. To date I have completed a few analyses of these data, including comparative evaluations of student perceptions of the introductory tools and how students performed on the content assessments. I am currently preparing for the second iteration of the study to be carried out this upcoming school year, which includes the development of a hybrid blocks/text tool.

7. EXPECTED CONTRIBUTIONS

I expect that findings of this dissertation will be of great interest to educators, curriculum designers, and the larger computer science education research community. The findings from this dissertation will contribute to our understanding of how the latest generation of block-based programming tools fit into more formal, structured educational spaces, as well as provide insight into the cognitive and affective aspects of such tools. Additionally, this work will provide insight into how blocks-based introductory tools perform relative to text-based programming environments designed for novices and evaluate one potential approach to blending the two modalities. We are at a critical juncture in the history of computer science education. The practices, tools, and curricula that are being developed today will become the standards used for years to come. It is essential that we are confident that the approaches we advocate today are effective at teaching the core concepts, engaging learners from diverse backgrounds, and successful in preparing students for the computational futures that await them.

8. REFERENCES

[1] Armoni, M. et al. 2015. From Scratch to "Real" Programming. *ACM TOCE*. 14, 4 (2015), 25.

[2] Dann, W. et al. 2012. Mediated transfer: Alice 3 to Java. *Proc. of the 43rd ACM SIGCSE*, 141–146.

[3] Dijkstra, E.W. 1982. How do we tell truths that might hurt? *Selected Writings on Computing*. Springer. 129–131.

[4] Lewis, C.M. 2010. How programming environment shapes perception, learning and goals: Logo vs. Scratch. *Proc. of the 41st ACM SIGCSE*, 346–350.

[5] Maloney, J.H. et al. 2008. Programming by choice. *ACM SIGCSE Bulletin*. 40, 1, 367–371.

[6] Meerbaum-Salant, O. et al. 2011. Habits of programming in Scratch. *Proc. of the 16th ITiCSE* (Darmstadt, Gr), 168–172.

[7] Meerbaum-Salant, O. et al. 2010. Learning computer science concepts with Scratch. *Proc. of the 6th ICER*, 69–76.

[8] Moskal, B. et al. 2004. Evaluating the effectiveness of a new instructional approach. *Proc. of the 35th ACM SIGCSE* 75-79.

[9] Powers, K. et al. 2007. Through the looking glass: teaching CS0 with Alice. *ACM SIGCSE Bulletin*. 39, 1, 213–217.