Block-based Programming and Preparation for Future Computer Science Learning

David Weintrop* & Uri Wilensky#
dweintrop@uchicago.edu, uri@northwestern.edu
Center for Connected Learning and Computer-based Modeling
* UChicago STEM Education, University of Chicago
# Learning Sciences & Computer Science, Northwestern University

Objectives

Block-based programming is increasingly becoming the way younger learners are being introduced to programming and computer science more broadly (Astrachan & Briggs, 2012; Goode et al., 2012; Weintrop & Wilensky, 2016). One of the major motivations for using block-based programming in introductory computing classrooms is the argument that such tools can lay an effective foundation and prepare students for future computer science learning. This poster will present data showing the soundness of this rationale based on a two-year study of block-based, text-based, and hybrid blocks/text programming environments in high school classrooms.

Theoretical Framework

This work brings an empirically-driven, design-based research approach to the study of classroom computer science education. Building on Structuration Theory (Wilensky & Papert, 2010) and the constructs of Webbing and Situated Abstractions (Noss & Hoyles, 1996), we investigate the representational affordances of different introductory programming modalities. The tools and curricula used in this study are grounded in the constructionist tradition (Papert, 1980).

Methods and Data Sources

To better understand the impact of programming modality on future learning, we designed a quasi-experimental study in three high school computer science classrooms with each class using a different programming modality (block-based, text-based, or hybrid blocks/text). Starting on the first day of school, students spent five weeks working in the introductory programming environment before transitioning to Java in the sixth week of the class. All three classes followed the same curriculum and were taught by the same teacher.

Results

Results from this study show students in the block-based condition scoring significantly better on the content assessment after five weeks compared students in the text condition after controlling for prior programming experience ($F(2, 75) = 4.53$, $p = .01$ with a Tukey HSD post hoc test showing the difference between Text and Blocks to be significant at $p = .01$). However, after ten weeks of working in Java, the students in the text condition's scores continued to improve, while students in the blocks condition saw no gain in their performance on the assessment, resulting in no difference between the classes after 10 weeks of Java programming ($F(2, 74) = .85$, $p = .43$). This result shows that block-based environments support students in their learning while using the tool, but do not better prepare students for future text-based programming endeavors. This finding is

consequential as it shows that block-based programming environments do not inherently better prepare learners for future text-based programming instruction, thus challenging the belief that the foundation laid by block-based tools will seamlessly transfer into text-based contexts.

Scholarly Significance

The contribution of this work is to open a critical dialog around the use of block-based programming environments in formal computer science classrooms and suggest future work that needs to be done to best take advantage of this increasingly popular programming modality in formal contexts. This work is timely and relevant for this venue given the increasing number of curricula that are being developed around block-based tools and the assumptions that designers, educators, and administrators hold around the outcomes of the use of block-based programming tools. Collectively, this work shows there is more work that needs to be done in considering how best to introduce learners to programming and computer science more broadly – both with respect to the design of programming environments and the curricula and pedagogies that accompany them.

References

Astrachan, O., & Briggs, A. (2012). The CS principles project. ACM Inroads, 3(2), 38–42.

Goode, J., Chapman, G., & Margolis, J. (2012). Beyond curriculum: the exploring computer science program. ACM Inroads, 3(2), 47–53.

Noss, R., & Hoyles, C. (1996). Windows on mathematical meanings: Learning cultures and computers. Dordrecht: Kluwer.

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

Weintrop, D., & Wilensky, U. (2016). Bringing Block-based Programming into High School Computer Science Classrooms. In Paper presented at the Annual Meeting of the American Educational Research Association (AERA). Washington DC, USA.

Wilensky, U., & Papert, S. (2010). Restructurations: Reformulating knowledge disciplines through new representational forms. In J. Clayson & I. Kallas (Eds.), Proceedings of the Constructionism 2010 conference. Paris, France.