The Need for Improved Support for Interacting with Block Examples

Michelle Ichinco and Caitlin Kelleher
School of Computer Science and Engineering
Washington University in St. Louis
St. Louis, MO, USA
michelle.ichinco, ckelleher@wustl.edu

***Abstract*—Programmers often attempt to use example code in order to fix bugs, learn, and remind themselves of code concepts. Many existing programming environments make text code examples available for the common programmer. Blocks code examples are typically less available within programming environments and harder to re-appropriate, especially for novices learning independently. This position statement suggests ways of making example code more available and useful to novice programmers in blocks programming environments.**

I. Introduction

Resources for learning programming, such as documentation, tutorials, and online forums, often provide snippets of example code. These resources require that the programmer leaves their programming environment or context. However, many blocks programmers working toward specific project goals, like apps and animations, may not want to leave their code context unless they have a problem. If a novice programmer learning independently outside of a classroom does not explore naturally, they may plateau in the blocks and skills they know how to use. Yet, little support exists for independent blocks programmers to access snippets of example code that relate to their code context. Ideally, blocks programming environments would support novices in learning new programming concepts throughout their coding process.

Imagine a blocks programming environment that adaptively suggests example code both to support the understanding of code the programmer already uses and to help the programmer expand their skill set. While the novice creates their project, they can choose to view example snippets from other projects that use similar programming blocks. This might provide inspiration or confidence that their code is correct. The system could also check the programmer's code for opportunities to suggest ways the programmer could improve their project using new code blocks. The examples the system presents would effectively emphasize the code related to the novice's project to help the novice programmer understand the value of the example. If a novice programmer decides to use a new block from an example, the system would also support them in finding that code block within the environment.

Blocks programming environments should support examples in two critical ways: by 1) making examples more accessible by recommending them within programming environments, and 2) making examples easier to learn from and re-appropriate. Combined, these improvements would make block code examples available throughout the programming process and would support programmers in exploring and learning a wide array of code blocks and concepts.

II. Recommending Examples

Unawareness of API (Application Programming Interface) methods and code constructs is a significant problem for novice programmers in blocks programming environments. One way to work towards supporting awareness of these highly useful blocks is to recommend the blocks and show examples to demonstrate their use. Systems can provide code example recommendations in order to: 1) help novices discover ways to use code blocks that they select on their own, and 2) introduce novices to unknown API methods or programming constructs that would improve their programs.

Many professional programming environments provide code recommendation, like auto-complete. Blocks programmers must typically drag and drop blocks from static palettes, with little assistance or adaptation. Yet, many code blocks allow or necessitate completion with the use of other blocks, requiring extra navigation. Recommending commonly coupled code blocks along with examples could help novice programmers by preventing them from having to search for the necessary code blocks. The code examples would enable users to compare their usage to other programs and find inspiration for new and better ways to complete their code.

Code recommendation could also introduce API methods or programming concepts that a programmer has not yet used. These recommendations would take into account a larger segment of a program and suggest API methods or concepts based on similar, but more complex versions of a novice's program. A recent study showed that novices accessed and used suggested information about API methods more than statically available documentation [1]. We believe this type of recommendation system could also introduce novices to more complex programming concepts as well. If suggested examples motivate novices to explore and learn new programming concepts, systems can begin to shape independent learners' programming knowledge while novices create personally motivating projects.

Suggesting examples to novices seems promising, but studies have shown that example code can often be challenging for novice programmers to understand and re-appropriate [2]. A code example recommendation system for novices should support novices in understanding the examples presented to them.

III. Interactive Examples

Many current resources, like tutorials and documentation, provide static examples. In order to help novices better understand and use examples, we propose creating interactive blocks examples that provide support for: 1) emphasizing, 2) explaining, and 3) appropriating blocks from the examples.

The visual nature of code makes it harder to emphasize relevant elements of an example. In a text example, highlighting can clearly emphasize black text. However, code blocks typically come in a variety of shapes and colors, making typical emphasis methods ineffective. Rather than using color for emphasis, examples should instead use methods like motion or making critical elements pop out, to better focus novices' attention on the relevant code blocks.

The emphasized blocks may also require explanation, as a user may not have seen or used a suggested code block in the past. Providing an explanation that a user can access by interacting with the example, like by hovering, might prevent explanations from overwhelming a user with too much information initially. Providing example code execution might also help a novice to better understand code meaning without drawing attention away from the emphasized code.

Once a programmer understands an example, they may want to use a suggested code block. Unlike text examples, in which a user can copy and paste a method or construct name to reuse it, blocks programming languages necessitate more support for reusing blocks. Examples could provide support similar to Scratch's backpack [3] or Looking Glass's reuse support [4]. Examples could also allow users to drag blocks into their programs directly from the example code, like dragging from a menu. However, these support methods may not help users learn where to find the blocks in the future. Novices would likely benefit from support for finding example code blocks within the programming environment.

IV. Conclusion

Code examples are an important and valuable resource for programmers, including novice programmers in blocks programming environments. Unfortunately, most current blocks programming environments provide little, if any support for code examples. Yet, novices require different and more extensive support than more experienced programmers using text languages, especially when learning independently. Suggested and supported examples could encourage and help independent novice programmers to use and understand new programming blocks and concepts.

References

[1] M. Ichinco, W. Y. Hnin, and C. L. Kelleher, "Suggesting API Usage to Novice Programmers with the Example Guru," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ser. CHI '17. New York, NY, USA: ACM, 2017, pp. 1105–1117.

[2] M. B. Rosson, J. Ballin, and H. Nash, "Everyday Programming: Challenges and Opportunities for Informal Web Development," in *2004 IEEE Symposium on Visual Languages and Human Centric Computing*, Sep. 2004, pp. 123–130.

[3] "Scratch - Imagine, Program, Share." [Online]. Available: https://scratch.mit.edu/help/videos/

[4] P. A. Gross, M. S. Herstand, J. W. Hodges, and C. L. Kelleher, "A code reuse interface for non-programmer middle school students," in *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 2010, pp. 219–228.