A Classification of Programming Styles in Scratch

Leonel Morales Diaz
Universidad Francisco Marroquin
6 Calle final, zona 10, Guatemala
litomd@ufm.edu

Luis Felipe Ayala Lopez
Universidad Francisco Marroquin
6 Calle final, zona 10, Guatemala
layala@ufm.edu

CLIHC '17, November 8–10, 2017, Antigua Guatemala, Guatemala

ACM ISBN 978-1-4503-5429-5/17/11.
https://doi.org/10.1145/3151470.3156639

Abstract
Scratch, the popular block programming platform created by MIT Media Lab, with more than 23 million projects shared by its users, provides interesting opportunities for studying utilization patterns thanks to the well documented datasets made available to researchers. In this paper, we describe our efforts to identify and categorize different programming styles by analyzing the blocks and sprites utilized by each user. Although our project is in early stages of development, we can point to indications that signal the existence of at least four types of users: scene designers, dialogue storytellers, complex animators, and remixers-copiers. We will describe each style, and outline our strategy to use the available data to validate the hypothesis that most users can be classified in one of these four types.

Author Keywords
Programming styles; computers and children; creativity support tools; social computing; data science.

ACM Classification Keywords
K.3.2 Computers and Education: Computer and Information Science Education; H.5.2 Information interfaces and presentation (e.g., HCI): User Interfaces.

Scene Designers and Dialogue Storytellers in Scratch

Scene Designers are those who are more interested in creating a good-looking scene than in animating it somehow. For them Scratch is like a canvas, an artistic tool where the objective is to craft the most aesthetic scene.

Dialogue Storytellers put a play into scene where one or more characters talk telling jokes or any kind of story. In Scratch it is a bit difficult to coordinate whose turn to talk is and in consequence it is easy to find single character scenes where the story is told by only one character.

Introduction

A fundamental form of human-computer interaction is computer programming [1]. Attempting a classification of the different forms in which programmers interact coding in computers is thus a highly relevant task for the HCI Field, among other reasons because it helps to better understand how that specific form of interaction occurs. Thanks to the availability of programming platforms aimed at young audiences, it is possible to attempt that classification for child programmers.

In this study the focus of such endeavor is Scratch, a product of the Lifelong Kindergarten Group at the MIT Media Lab, specifically designed to enable children to learn and engage in programming activities [2].

The success of Scratch as a programming platform for kids is well documented [2][3]. Its website keeps a counter of the number of projects being shared: more than 24 million at the moment of writing this manuscript [4].

Programming and sharing projects through the website generates interactions, communications, visits, explorations, and reutilizations of user code. Scratch is not only a programming platform, it is also a social media site where users can follow each other, indicate they like the work of others, post comments and messages for the authors, and remix the projects as they like. The platform keeps records of all these social exchanges [5].

Creating projects and interacting with others generates an enormous amount of data that is available for study [6]. In this research project, we are trying to find out if that data may reveal what we call programming styles, that is distinguishable and classifiable patterns of building programs that can be studied to learn more about how Scratch users form mental models on coding and express their personality in the process.

Similar attempts have been made in the past. The literature reports at least two studies, in the first Orni Meerbaum-Salant, Michal Armoni and Mordechai Ben-Ari applied observation, analysis of programs created by young students and interviews to identify habits of programming in Scratch [7], in the second Efthimia Aivaloglou and Felienne Hermans utilized data scraping to retrieve 250,000 projects for analysis, investigating complexity, abstraction and programming concepts in them [8][9]. In their papers they report "code smells" – large scripts, dead code, duplicated code blocks, unmatched broadcast messages – that are common.

Our approach differs in what we are trying to pay attention to. Salant, Armoni and Ben-Ari [7] seem to focus on programming habits and Aivaloglou and Hermans [8] in code smells and in verifying the utilization of advanced programming concepts.

Instead, we are trying to find distinguishable patterns that enable the classification of users in at least four categories: scene designers, dialogue storytellers, complex animators and remixers-copiers. Our hypothesis is that most young programmers will fall in one of these four categories. The existence of users that mix the styles in their projects or that resist classification will also be analyzed.

Although it would be interesting to propose additional hypothesis and research questions like: finding if users persist in the same style through a long period of time, the existence of users that clearly turn to different styles for different projects and others, in this stage of our research project we are only interested in seeing if the classification is possible.

Complex Animators and Remixers-Copiers in Scratch

Complex Animators create animations that may or may not have a story behind. Their interest is not the story but the demonstration of a complex combination of features or instructions that produce complicated effects or movements. Those complex animations often require the use of instruction blocks that represent abstract programming concepts.

Remixers-Copiers prefer to work on the basis of a known project, one they can use as a model to copy or modify. They may need to know as exactly as possible what is the final product they want to produce. In the Scratch platform it is easy to copy and remix existing projects. In fact for each project, it is possible to learn if it is original or a modified version of a previous one.

The Four Styles

The selection of the four styles to investigate is based on three factors, the first is the revision of previous work in a different programming platform where these styles were identified and described [10][11]. Although the names and descriptions do not match exactly with the ones in this study, the connection between the two sets is evident.

The second is the experience of the authors teaching programming or programming-related courses. Through observation and analysis, it is easy to conclude that it makes sense to look for these styles in Scratch programmers. There are inherent differences in skills, motivations, interests and worries among students and those become apparent in the style they use.

Finally, the third factor is the examination of the projects publicly available in the Scratch site [4]. At the moment of visiting the page there is an "Explore" section that categorizes projects into animations, art, games, music, stories and tutorials. The similarity between the proposed styles and the categories in the site might indicate that they are identifiable beyond the subjective criterion of the authors.

Preliminary Findings

The datasets used [12] for this study include data of five years of Scratch online usage (from 2007 to 2012). The Scratch Online Community collects detailed data regarding projects, users and block.

At the moment of writing this manuscript with the available data we can observe that roughly 1.82 projects were built by each user, however there are a little over twenty users with more than one thousand projects developed. The standard deviation of the number of projects per user is 26.87. This suggest that there are different approaches to using Scratch and that may limit the scope of analysis at this point in time since they require a finer revision.

To look for patterns of programming styles, specific block usage was analyzed. Based on the researchers' criterion, blocks that represent the hypothesized programming styles were selected. For example, for the Complex Animators group, the NextBackground, NextCostume and SetSizeTo blocks were chosen. The mean of the number of each of these blocks usage was calculated per user to look for patterns.

The dataset reveals some singularities: two users are far from the behavior of the group in terms of block usage. Also, the number of users that did not use the NextBackground block is 253,948 suggesting the dispersion of the data.

There are indications of heavy usage of the block NextBackground by the top ten users. No relation between the top users of NextBackground and the Next Costume or SetSizeTo blocks seems evident, so further exploration will be conducted.

Similar data analysis was conducted for the other three suggested programming styles in Scratch. The results of these calculations present difficulties beyond the scope of this preliminary report.

Conclusion

In this paper, we have discussed the importance of understanding the different forms in which child programmers engage in using the Scratch platform.

Four basic styles were described: scene designers, dialogue storytellers, complex animators, and remixers-copiers. Our hypothesis is that most users can be classified in one of these four styles according to their utilization pattern of the platform and the programming

Five Steps to Verify our Hypothesis

One: Review metadata and dataset documentation.

Two: With a copy of the blocks table, add two fields: one to classify each block in a category linked to one of the four styles, and a second to store the score of the block in that category.

Three: Design an algorithm to classify each individual project into one of the four styles or mark it as unclassifiable. The basis for this algorithm is the relative intensity of utilization of each of the blocks.

Four: Obtain a sample from the users table on the basis of a inclusion criteria. Users that have little or no projects or have been inactive for a long time or any other, can be excluded.

Five: Run the classifier algorithm on all the projects of users in the sample and identify the dominant style for each one.

blocks. To validate the hypothesis, we devised a process composed of 5 steps. At the moment of writing this manuscript the research project has only covered the first two and advanced in the analysis of the data to design a classification algorithm.

Some features of the data available were identified and will need further analysis. The dispersion of the values for the variables calculated initially are such that a careful review needs to be done.

As a side product of this study, the authors are learning a lot about the general use of Scratch among young programmers. We appreciate this learning very much.

References

1. Jacko, Julie A., ed. 2012. Human computer interaction handbook: Fundamentals, evolving technologies, and emerging applications. CRC press.
2. John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The Scratch Programming Language and Environment. *Trans. Comput. Educ.* 10, 4, Article 16 (November 2010), 15 pages.
3. Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. 2009. Scratch: programming for all. *Commun. ACM* 52, 11 (November 2009), 60-67.
4. Scratch web site. Retrieved August 30, 2017 from https://scratch.mit.edu/
5. Ricarose Roque, Yasmin Kafai, and Deborah Fields. 2012. From tools to communities: designs to support online creative collaboration in Scratch. In *Proceedings of the 11th International Conference on Interaction Design and Children* (IDC '12). ACM, New York, NY, USA, 220-223.
6. B. M. Hill and A. Monroy-Hernández, "A longitudinal dataset of five years of public activity in the Scratch online community," *Scientific Data*, vol. 4, pp. 170 002 EP -, 01 2017.
7. Orni Meerbaum-Salant, Michal Armoni, and Mordechai Ben-Ari. 2011. Habits of programming in scratch. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (ITiCSE '11). ACM, New York, NY, USA, 168-172.
8. Efthimia Aivaloglou and Felienne Hermans. 2016. How Kids Code and How We Know: An Exploratory Study on the Scratch Repository. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (ICER '16). ACM, New York, NY, USA, 53-61.
9. Efthimia Aivaloglou, Felienne Hermans, Jesús Moreno-León, and Gregorio Robles. 2017. A dataset of scratch programs: scraped, shaped and scored. In *Proceedings of the 14th International Conference on Mining Software Repositories* (MSR '17). IEEE Press, Piscataway, NJ, USA, 511-514.
10. Leonel Morales Diaz, Laura S. Gaytan-Lugo, and Lissette Fleck. 2015. Interaction Styles in Alice: Notes and Observations from Computer Animation Workshops. In *Proceedings of the Latin American Conference on Human Computer Interaction* (CLIHC '15). ACM, New York, NY, USA.
11. Leonel Morales Diaz, Laura S. Gaytan-Lugo, and Lissette Fleck. 2015. Profiling styles of use in Alice: Identifying patterns of use by observing participants in workshops with Alice. In *Proceedings of the 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (BLOCKS AND BEYOND '15). IEEE Computer Society, Washington, DC, USA, 19-24.
12. Scratch Research Data Sharing Agreement. Retrieved July 16, 2017 from https://llk.media.mit.edu/scratch-data/