Learners' perspectives on block-based programming environments: Code.org vs. Scratch

Johannes Krugel
Technical University of Munich
School of Education
Munich, Germany
krugel@tum.de

Alexander Ruf
Technical University of Munich
School of Education
Munich, Germany
alexander.ruf@tum.de

ABSTRACT

We report from a comparative study regarding two popular block-based programming environments for the introduction of algorithmic concepts, namely Code.org and Scratch. A quasi-experiment was conducted with five classes (grade 7) from two secondary schools ($n = 122$). To investigate the effects of both learning environments, the students were divided into two groups.

Following a mixed-methods research approach, we use quantitative and qualitative methods to gain a comprehensive understanding of learner's perspectives and skills. We measure the learners' perceived self-regulation and intrinsic motivation. The learners of Code.org show a higher intrinsic motivation compared to the Scratch group. Qualitative analyses of answers to open-ended questions reveal positive and negative aspects of the learning environments. With Code.org, the learners in particular like the fun and the challenges, while for Scratch, they like the feeling of actually programming something and the very free environment.

CCS CONCEPTS

• **Applied computing** → **Computer-assisted instruction**; *Interactive learning environments*; • **Social and professional topics** → *Computational thinking*; *K-12 education.*

KEYWORDS

Interactive learning environments, Block-based programming, Algorithmic thinking, Computational Thinking, Motivation

ACM Reference Format:
Johannes Krugel and Alexander Ruf. 2020. Learners' perspectives on block-based programming environments: Code.org vs. Scratch. In *Workshop in Primary and Secondary Computing Education (WiPSCE '20), October 28–30, 2020, Virtual Event, Germany*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3421590.3421615

1 INTRODUCTION

Computational Thinking is considered to be an important ability every student should learn in school [12]. Block-based programming languages are frequently used for teaching nowadays [1, 10]. We focus on web-based tools because they can easily be used in a flipped classroom setting [3] or when learning at scale [4]. This is of special interest in the current COVID-19 ("Coronavirus") pandemic where students have to learn at home with their own devices.

WiPSCE '20, October 28–30, 2020, Virtual Event, Germany

ACM ISBN 978-1-4503-8759-0/20/10.
https://doi.org/10.1145/3421590.3421615

2 TEACHING INTERVENTIONS

We target learners without any previous knowledge of algorithms and compare teaching with Code.org and Scratch. The learning objectives cover basic algorithmic concepts usually included relatively at the beginning of courses on computational thinking [5]. Due to the COVID-19 pandemic, the students learned at home with detailed instructions prepared by the teachers (both authors of this paper). They received a 1-page working sheet each week with reading instructions, links to explanation videos, and exercises. The format of the tasks varied to include different categories [2]. We did explicitly not make the learning sequence of both groups exactly the same because this would contradict the inherent differences of the learning environments. The learning objectives and the difficulty was, however, very similar. The first experimental group learned on Code.org with the *Accelerated course*[1]. The learners program a figure on the screen to, e. g., follow a path in a maze or to draw geometric shapes. The course is based on game-based learning and microlearning, consisting of a series of small games with several levels. The course contains short videos introducing the concepts (in English with German subtitles); we created 13 additional short explanatory videos very similar to the videos of the Scratch group. The second group learned with the web-based Scratch 3.0[2]. Figures can be programmed to move on the screen, perform other actions and react to events. Similar to Code.org, the figures also draw geometric shapes (using the *Pen* extension). The teaching material is based on a continuing education course for teachers[3] and includes 18 explanatory videos of an experienced teacher.

[1] https://studio.code.org/s/20-hour/
[2] https://scratch.mit.edu/projects/editor/
[3] https://lehrerfortbildung-bw.de/u_matnatech/informatik/gym/bp2016/fb1/

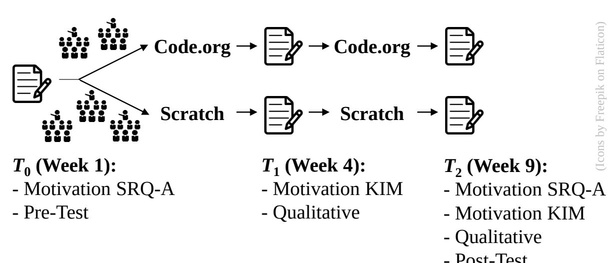

Figure 1: Study Design

3 METHODOLOGY

We conducted a quasi-experiment with 5 classes of two secondary schools in grade 7, where Computer Science is a compulsory subject and has 1 lesson (45 minutes) per week. There were 122 students of age 12–14 with a balanced gender distribution, see Table 1. The intervention spans 9 weeks and we collect data at three time points T_0, T_1, and T_2. At the time of writing this paper we had completed the first 5 weeks and data collections at T_0 and T_1. The research design is visualized in Figure 1. Other similar experiments often focus on quantitative methods [8, 9]. We use a mixed-methods approach to compensate the weaknesses and combine the strengths of quantitative and qualitative methods. We use a pre-test, a post-test, and free text feedback. Additionally we employ the "Academic Self-Regulation Questionnaire" (SRQ-A) which can be used to determine the self-determination index (SDI) [7]. The intrinsic motivation is assessed with the KIM instrument [11] with 4 subscales, see Table 1.

4 RESULTS REGARDING THE MOTIVATION

At the beginning of our study T_0, both experimental groups have the same average SDI of 2.21 (see Table 1). At T_1, we received 79 responses regarding the intrinsic motivation. In this paper, we analyze the data with a relatively simple comparison based on the arithmetic mean of the Likert items (like [8]). The students of Code.org exhibit have a higher intrinsic motivation (with consistently higher average value on the individual subscales *Interest/Enjoyment*, *Perceived Competence* and *Perceived Choice*, and a lower value for *Pressure*). This was surprising for us because we expected a higher intrinsic motivation for Scratch which offers more freedom to the learners. Our mixed-methods approach proved to be helpful here because the qualitative analyses yields further insights.

5 QUALITATIVE RESULTS

The answers to the qualitative survey were analyzed using an inductive qualitative content analysis based on the methodology of Kyngäs [6]. Each answer could be assigned to multiple categories. We performed several iterations over the data to adapt and refine while maintaining a coding manual. This resulted in 18 positive and 15 negative aspects.

For Code.org, the mentioned positive aspects of the learners frequently fall into the categories *Fun* and *Challenges* (e. g. "*I had to think but at the same time it was fun.*"), as well as *Easiness* and *Usability* (e. g. "*It is very easy to understand and you only have to pull blocks together.*"). The most prominent negative aspects is, that some students regard the exercises as too easy.

For Scratch, the by far most frequent category of positive aspects is *Programming*, followed by *Trying out* (e. g. "*I liked to do my own programming and just try out how that works.*"). Also aspects related to the freedom of the learning environment are frequently mentioned (e. g. "*With Scratch you can make the character do almost anything.*"). As negative aspects, the learners mentioned the difficulty, especially at the beginning of the sequence and some technical issues. This perceived difficulty is probably also due to the fact that Scratch offers a rich set of functionality which might be overwhelming for some beginners; we assume that this is also an explanation for the relatively lower intrinsic motivation.

Table 1: Quantitative results per learning environment (T_0 and T_1 denote the respective number of participants).

Group	n	Female / Male	T_0	SDI	T_1	Interest/ Enjoyment	Perceived Competence	Perceived Choice	Pressure/ Tension
Code.org	47	24/23	43	2.21	34	3.21	2.98	3.21	0.73
Scratch	75	38/37	67	2.21	45	3.02	2.80	2.93	0.85
Total	122	62/60	108	2.21	79	3.11	2.89	3.06	0.79

6 CONCLUSION

A limitation is the number of participants and the non-random assignment to the experimental groups. Additionally, only a fraction of the learners participated in data collection T_1. However, at T_2 we were able to collect data from the majority of students. The analysis of this data will provide a more detailed view and allows the comparison of the whole teaching sequences, including the achieved learning outcomes.

REFERENCES

[1] David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, and Franklyn Turbak. 2017. Learnable Programming: Blocks and Beyond. *Commun. ACM* 60, 6 (May 2017), 72–80. https://doi.org/10.1145/3015455

[2] Katharina Geldreich, Mike Talbot, and Peter Hubwieser. 2019. Aufgabe ist nicht gleich Aufgabe – Vielfältige Aufgabentypen bewusst in Scratch einsetzen. In *Informatik für alle*, Arno Pasternak (Ed.). Gesellschaft für Informatik, Bonn, 171–190. https://doi.org/10.18420/infos2019-c3

[3] Jeong Ah Kim and Hee Jin Kim. 2017. Flipped Learning of Scratch Programming with Code.Org. In *Proceedings of the 2017 9th International Conference on Education Technology and Computers* (Barcelona, Spain) *(ICETC 2017)*. ACM, New York, NY, USA, 68–72. https://doi.org/10.1145/3175536.3175542

[4] Johannes Krugel and Peter Hubwieser. 2017. Computational thinking as springboard for learning object-oriented programming in an interactive MOOC. In *2017 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 1709–1712. https://doi.org/10.1109/EDUCON.2017.7943079

[5] Johannes Krugel and Peter Hubwieser. 2018. *Strictly Objects First: A Multipurpose Course on Computational Thinking*. Springer International Publishing, Cham, 73–98. https://doi.org/10.1007/978-3-319-93566-9_5

[6] Helvi Kyngäs. 2020. *Inductive Content Analysis*. Springer International Publishing, Cham, 13–21. https://doi.org/10.1007/978-3-030-30199-6_2

[7] Florian H. Müller, Barbara Hanfstingl, and Irina Andreitz. 2007. *Skalen zur motivationalen Regulation beim Lernen von Schülerinnen und Schülern: Adaptierte und ergänzte Version des Academic Self-Regulation Questionnaire (SRQ-A) nach Ryan & Connell, Wissenschaftliche Beiträge aus dem Institut für Unterrichts- und Schulentwicklung*. Technical Report. Alpen-Adria-Universität Klagenfurt.

[8] Alexander Ruf, Andreas Mühling, and Peter Hubwieser. 2014. Scratch vs. Karel: Impact on Learning Outcomes and Motivation. In *Proceedings of the 9th Workshop in Primary and Secondary Computing Education* (Berlin, Germany) *(WiPSCE '14)*. ACM, New York, NY, USA, 50–59. https://doi.org/10.1145/2670757.2670772

[9] Mazyar Seraj, Eva-Sophie Katterfeldt, Kerstin Bub, Serge Autexier, and Rolf Drechsler. 2019. Scratch and Google Blockly: How Girls' Programming Skills and Attitudes Are Influenced. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research (Koli Calling '19)*. ACM, New York, NY, USA, Article 23, 10 pages. https://doi.org/10.1145/3364510.3364515

[10] David Weintrop. 2019. Block-Based Programming in Computer Science Education. *Commun. ACM* 62, 8 (July 2019), 22–25. https://doi.org/10.1145/3341221

[11] Matthias Wilde, Katrin Bätz, Anastassiya Kovaleva, and Detlef Urhahne. 2009. Überprüfung einer Kurzskala intrinsischer Motivation (KIM). *Zeitschrift für Didaktik der Naturwissenschaften* 15 (2009), 31–45.

[12] Jeannette M. Wing. 2006. Computational Thinking. *Commun. ACM* 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215