# Kids Block Coding Game: A game to introduce programming to kids

**Luís E. T. Forquesato[1], Juliana F. Borin[2]**

[1] SIDI – Samsung Instituto de Desenvolvimento para a Informática
Rua Aguaçu, 171, Lot. Alphaville Campinas – 13098-321 Campinas, SP – Brasil

[2] IC – Instituto de Computação – UNICAMP
Av. Albert Einstein, 1251, Barão Geraldo – CEP 13083-852 – Campinas, SP – Brasil

luisforque@gmail.com, juliana@ic.unicamp.br

***Abstract.*** *In the near future, many areas will need at least a basic knowledge of computer science. Nonetheless, studies have shown that professors are currently having trouble teaching those same concepts in universities. In consequence, the idea of introducing programming theory to children is trending. This article presents a game that provides problems for the user to solve using blocks. The game targets the six to eleven age range, although it supports other ages as well. User trials affected the product through the development phase, changing user interaction and game design. A few children have already tested the final product and the results were positive: the tool was regarded as being fun, educational, and engaging.*

## 1. Introduction

Computational thinking education is a topic that has acquired a great deal of attention after a publication by Jeannete Wing [Wing 2006] in which she argues the importance for people in the job market to know basic concepts of computer science, such as abstraction, automation, and complex problem solving.

Teaching programming is a common choice to introduce computational thinking concepts. When learning programming, students face several challenges, problem solving and debugging among them. These activities are central to the concept of computational thinking [Flórez, Casallas, Hernández, Reyes, Restrepo and Danies 2017]. Previous research shows that teaching logic and programming to undergraduates is a hard task [Bromwich, Masoodian and Rogers 2012] [Liu, Cheng and Huang 2011]. This finding lead us to believe that the earlier a person is accustomed with those abstract concepts, the easier it will be for her to learn and use that knowledge when necessary. According to Piaget, kids of age approximately seven to eleven are on the concrete operational stage [Piaget and Cook 1952], in which they already can construct abstract ideas and logical structures on their mind.

Children are spending more time than ever[1] using mobile devices. At the same time, it is becoming common practice to teach children a very basic form of

[1] The Common Sense Census: Media use by kids age zero to eigth 2017
https://www.commonsensemedia.org/research/the-common-sense-census-media-use-by-kids-age-zero-to-eight-2017

programming or robotics, either in school or in extra-curricular classes. This scenario has driven research in educational games and applications. However, most of the research done on this topic focuses on older children [Bromwich, Masoodian and Rogers 2012] [Denner, Werner and Ortiz 2012] [Werner and Campe 2012] [Liu, Cheng and Huang 2011] [Werner, Denner and Campe 2012] [Webb 2010] [Brennan and Resnick 2012]; there are not so many examples of studies analyzing the usage of those educational software for children younger than 10 years old.

Kids Block Coding Game was developed to be a game that introduces programming concepts by using blocks to resolve a pathing problem. The focus is on the 6 to 10 age range. Besides the educational goal of the project, another motivation is to investigate the possibility of using stealth assessment [Shute 2011] to determine whether the kids are indeed learning computational thinking and programming concepts while playing the game. This analysis is not present in this work but will be discussed in the section containing future work. This paper describes in further detail the game's development, user trials, and final product.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 explains how the application was developed: listing important requirements and noteworthy implementation details. Section 4 presents the preliminary user tests and the acquired results. Finally, Section 5 contains the conclusion and the future work for the study.

## 2. Related work

During the research for computational thinking tools and applications, the authors noticed that there are many alternatives for the older audience; in particular, Scratch[2] is very well known. However, there are fewer good options for younger children. Kids in age 6 to 10 can already understand basic logic and abstract concepts, meaning they are valid targets for elementary logic applications.

Duncan et al. produced a theoretical study about teaching computational thinking in different ages [Duncan, Bell and Tanimoto 2014]. The authors ask a valid question in "Should your 8-year-old Learn Coding?", discussing historical elements, developmental psychology, school curriculum, among others. The authors then conclude that it is important for kids to learn this skill, as long as some important restrictions are met. Last year, during the Workshop of Education in Computation (WEI), the Brazilian Computing Society (SBC) proposed a scenario[3] in which a basic sort of computational thinking is taught at schools to children as early as four years old.

There are many tools used to teach computational thinking. Commonly, computational thinking tools can be categorized into games, robotics, or programming [Duncan, Bell and Tanimoto 2014]. Duncan et al. lists 47 tools graded across difficulty

[2] https://scratch.mit.edu/

[3] http://computacaonaescola.com.br/proposta-de-inclusao-de-computacao-na-educacao-basica/ (in portuguese)

and age ranges. Within those, the authors found 20 applicable to kids between 3 and 10 years old.

Similar to the application proposed in this paper, there is the Box Island[4]. It is well known, has received awards and it is considered for the Hour of Code[5] certificate. Its age target is six plus years old. Box Island includes complex concepts like functions and has 100 levels.

Another similar application is Lightbot[6]. Lightbot uses different types of blocks and visual, but the concepts involved are similar. Lightbot is said to be all-ages friendly and has 40 puzzles. Gouws et al. did a case study in which they analyzed the game for which computational thinking concepts are being passed to the user in each puzzle [Gouws, Bradshaw and Wentworth 2013]. The authors concluded that "models and abstraction" are exercised in 92% of the levels, while "inference and logic" is practiced in 50%. By extrapolation, they concluded that the game teaches computational thinking well, presenting 74% of the content of computational thinking in some way.

Game Logic aims to help teaching programming logic [Netto, Medeiros, de Pontes and de Morais 2017]. The game is organized in levels, and the user needs to use programming blocks to solve each problem. The central difference between Game Logic and Kids Block Coding Game is that the latter has a younger age target.

## 3. Development of the game

Samsung has an ecosystem[7] with a handful of mobile applications dedicated to children, and the proposed game was developed to be a part of that context. The characters can be, for that reason, already known to the user. The game's commercial name is Crocro's Adventure.

Kids Block Coding Game works in all sorts of devices and tablets, of multiple screen sizes and hardware. Despite originally targeting children of ages 6 to 11, the game can be played by younger kids because it does not have text, and can be challenging to adults.

### 3.1. Requirements

At the first moment, the idea was to create an application that was fun and challenging for children, but could also teach basic programming. When targeting young children, we knew it could not be too hard or complicated. To that end, the choice to use block programming was natural, seeing that they are intuitive and not punishing by grammar

[4] https://boxisland.io/

[5] *"The Hour of Code started as a one-hour introduction to computer science, designed to demystify "code", to show that anybody can learn the basics, and to broaden participation in the field of computer science."* https://hourofcode.com/

[6] http://lightbot.com/

[7] http://www.samsung.com/global/galaxy/apps/kids-mode/

mistakes. Trying to appeal to children′s spatial knowledge, the game was thought up as a puzzle, in which the main character is lost in a simple maze and the user needs to use blocks to show him the way out. New blocks are unlocked with new levels. Using blocks was a natural decision because they visually represent a command that is going to execute and are easy to use and to understand by kids, without the onus of having a complex syntax.

The application has some elements that closely relate to programming, in order to complete its initial goal of being educative. The order that the user places blocks in the command line is the same as the one in which commands will be executed, for example. This is analogous to how programming works, where code is entered and executed in a specific order. Similarly, the blocks are executed only after a ′play′ button is selected, which is comparable to running a program after coding. During execution, the player can see exactly what action each command has the character do. When the execution fails, the software places a red sign on the wrong block and the user receives the option to debug his solution.

To make the game fun and engaging, the game design has a storytelling element. Upon entering the game, the user sees a video showing why he should help the character by solving his puzzles. Users can also earn prizes and unlock more content after finishing a task. The story shows that the character lives in a boat and has a collection of beautiful candies kept in his fridge, but after a huge storm the boat crashes and he loses all his cherished candy, spread all around the world. Every level is a chance to recollect one of his candies or unlock ways to help him in future tasks, by moving the character from one island to the next.

The game′s structure consists of 41 stages with increasing difficulty, and the user can only proceed to the next level by finishing the current one. The first few challenges presents the basics of the game, with a hands-on tutorial on how the gameplay works. Characters are unlocked throughout the game, up to four by the end. Each character has a special ability based on its personality that helps him cross a specific type of obstacle. Following the common, directional blocks, players unlock the ′jump′ block, that can allow them to jump over a small piece of water; and then multipliers, that can be attached to another block and have that command execute repeatedly, equivalently to a loop. In order to support creativity and increase replay value, many puzzles can be solved with more than one solution.

Using directional blocks provide the opportunity for users to practice their abstraction, because they need to visualize how that block will affect the character without actually seeing it happen. Likewise, the ‘jump’ block represents a very basic logic condition, considering that kids will need to think, “If there is an empty space, then I need to ‘jump’”. The multipliers embody a primitive control flow, and every character’s special powers allow for the design of harder puzzles, which in turn can lead to better problem solving skills.

The application has some secondary use cases. One important requirement was that the application should provide a possibility for multiplayer local interaction,

between child and parent, or siblings, for example. This feature was implemented by creating a screen where one user can create puzzles and another can solve it. Finally, the application also provides a lounge where kids can have fun by interacting with unlocked content, without needing to solve puzzles.

Since this game is part of a research project which aims to investigate the use of stealth assessment to assess computational thinking skills it was important to have every relevant event made by the user logged and saved. Collecting analytics data is crucial to analyze the difficulty of the levels, how long it usually takes a kid to solve it, when he/she grows weary, among others. Logged information includes, for example, adding or removing a block, selecting the 'play' button, winning or losing, entering and exiting the application or each screen.

Finally, while brainstorming the application, the idea to produce a game with as little text as possible appeared. There are no menus and explanations on how to play the game. This not only enables the game to be played by younger kids who cannot read well, but also facilitates the internationalization part of the commercialization.

### 3.2. Architecture and implementation

The application was implemented using Unity3D. Unity3D is a game engine that facilitates game development immensely. The programming language used was C#. Most of the game was implemented in 3D mode, with an isometric camera that moves while the user swipes the screen.

The game does not need an internet connection to be played; all of its content can be accessed locally and is stored locally. The game is available for downloading in Android[8] and iOS[9] devices.

The architecture used was based upon Clean Architecture[10]. Clean Architecture is a collection of design patterns that appeared specifically for Android native development, but can be used in other types of applications, such as Unity3D.

### 3.3. User interface

The user interface is composed of realistic backgrounds with cartoon-like elements to increase engagement by the kids. There are four types of screens: the screen where the user can choose the next level; the screen where he can see the content he has unlocked; the screen where he can build a custom level; and the screen where he solves the level.

The screen where the player chooses the level is a horizontal-scrolling screen with only a couple of levels visible. Only by finishing those, the next few challenges appear. The authors took this decision to increase the gamification of the game; the player can have an achievement effect upon unlocking a new play section. Treasure

[8] https://play.google.com/store/apps/details?id=com.sec.kidsplat.kidsbcg
[9] https://itunes.apple.com/us/app/crocro-adventure/id1364966911?l=pt&ls=1&mt=8
[10] https://8thlight.com/blog/uncle-bob/2012/08/13/the-clean-architecture.html

chests and bosses challenges were scattered to make the game funnier and less repetitive. Treasure chests contain either new blocks or new characters. The screen in Figure 1 is the base; it redirects the user to the others.

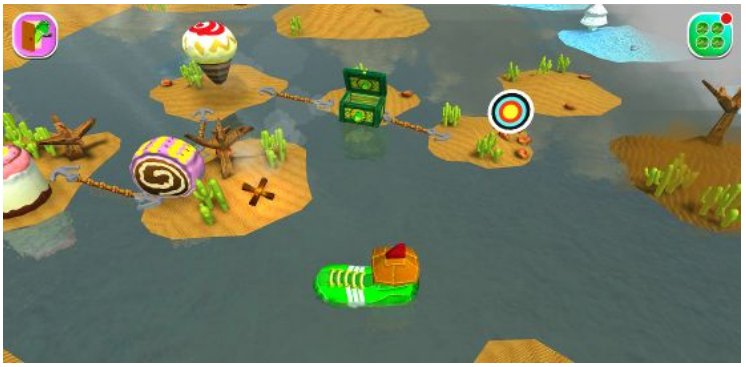

**Figure 1. The first screen, where users choose the level to play**

The puzzle screen (Figures 2 and 3) is where the game actually happens. The user sees one island on the left, where the character awaits, and another island on the right, where the level objective is. On the bottom part of the screen, the player can see the available blocks, the command line with empty spaces where blocks are added, and the 'play' button used to start the execution after the command line is filled accordingly. Once the player taps the 'play' button, the character starts moving in the direction represented by the commands, trying to cross to the other island. If the character reaches the goal and the solution is correct, congratulations appears and the user is taken to the previous screen. Otherwise, the incorrect block renders red, the character goes back to the start and the child can try again. By showing the incorrect block and allowing the player to try again, the authors expect to stimulate trial and error and debugging, both useful programming techniques. The game does not happen live, meaning that first, the kid does something, and only after clicking 'play', the solution runs. This promotes planning, organization and decomposition of the solution in steps.

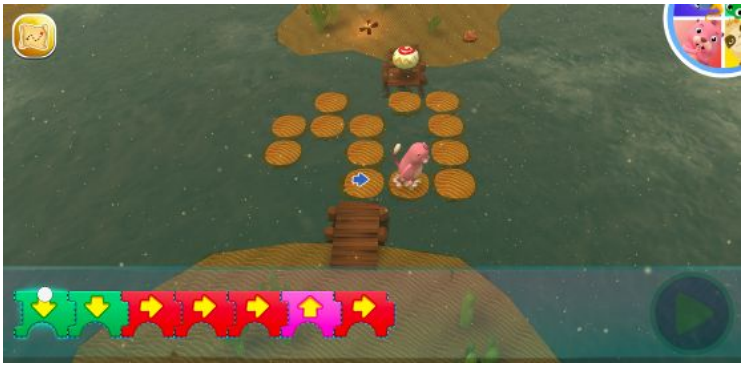

**Figure 2. One of the first levels, while the character is animating the solution**

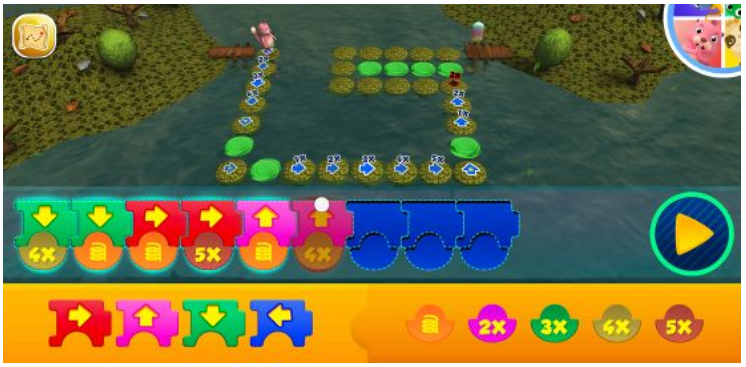


**Figure 3. A harder level showing a wrong solution and the visual feedbacks**

The third screen is where the child can see what he has unlocked. The characters are animating, the candies are listed, and the videos are watchable. This screen encourages the kid to boast his achievements by interacting with others, either his family or his friends, for instance.

In addition, to improve social interaction and collaboration, the application has a screen where the user creates and plays new puzzles. This feature allows, for example, a parent to create a custom level specific to his child difficulties.

There are also a few videos scattered between puzzles, in specific moments when the player is going to unlock important elements. Videos are used to capture and increase a child's attention.

## 4. Tests and results

During the development phase, we had user trials with 15 kids in order to find design problems, validate concepts and discover how they felt about the game. The kids were placed in a closed meeting room with their parents and given the game to play. There was no introduction on how to play the game, and interaction was kept to a minimum.

Early in development, the application had a 'reset' button that would clear the command line, removing all blocks already added. This seemed natural to the authors, being adults, but in the user trial, not a single child used it; they would just simply remove blocks one of a time by dragging and dropping. Similarly, in the beginning of the project, correctly solving problems would give the player stars; he would get three stars for an optimal solution and less whenever he used more blocks than necessary. At the user trial, the authors noticed that most kids were not interested in the stars, did not care how many stars they got, they just wanted to go further in the game faster. In both cases, the user trial affected the game design for the user interaction.

In the first release version, the game play was similar to the present, but the application did not have unlockable characters, videos and sections. In the trial, kids would often run tedious and stop before completing the game. This caused a major rethink of features and how to keep them engaged, which led to the current product.

Furthermore, some kids enjoyed seeing the videos so much that the authors implemented a way that they can watch again all videos that they had already unlocked.

Regarding the final product, the user trial reviews were majorly positive; the kids enjoyed the game new looks and features a lot better. They had more interest in solving the puzzles and continued to practice their logic and problem solving for a larger amount of time. With that said, we did not have a single child that managed to solve all levels and complete the game, possibly because the game is too long to be completed in one play session. Consequently, we had not been able to validate the difficulty of the more advanced levels.

Tests were mainly validating the user experience, and did not focus on learning, which is a harder appraisal. To do that, the authors will use usage statistics and a strict play session with a group of children from different age ranges, on another study.

The authors believe that the final game product turned out fun and engaging as well as challenging and informative. The product received international attention in the event Samsung Developers Conference 2017, in San Francisco[11].

## 5. Conclusion and future work

This paper presented the application Kids Block Coding Game, an educative game proposed to teach logic and basic computational thinking to children using block programming.

The impact of user trials on the development of the current version of the game was discussed. It is interesting to note that many of the choices made by experienced programmers when developing a tool to teach programming had to be rethought during this process. The authors believe that this discussion may greatly contribute to other researchers/developers interested in developing tools for computational thinking and programming education.

In the future, the authors ought to validate the difficulty of the whole game against kids of age six to eleven. Additionally, the application is part of another study that intends to validate the use of stealth assessment [Shute 2011] to determine if the player is learning logic and computational thinking concepts. For this study, a strict user trial will be conducted, and log usage from participants will be analyzed quantitatively to conclude if the application is indeed educational, and thereafter if this assessment technique is viable in this scenario.

---

11 *http://www.samsungmobilepress.com/stories/sdc-2017:-our-commitment-to-helping-kids-gain-digital-literacy*

## Acknowledgements

The authors would like to thank the support from SIDI and Samsung in providing the possibility and time for developing the application and writing this article. The authors would also like to thank the whole team that helped create Crocro's Adventure.

Finally, a special thanks to the kids that participated in the voluntary user trials, providing valuable feedback that resulted in this fun and educational game.

## References

Bromwich, K., Masoodian, M. and Rogers, B. (2012) "Crossing the Game Threshold: A System for Teaching Basic Programming Constructs", Proceedings of the 13th International Conference of the NZ Chapter of the ACM's Special Interest Group of Human-Computer Interaction, 56-63.

Brennan, K., Resnick, M. (2012) "New frameworks for studying and assessing the development of computational thinking", American Educational Research Association 2012, 1-25.

Denner, J., Werner, L., Ortiz, E. (2012) "Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts?", Computers & Education Volume 58 Issue 1, 240-249.

Duncan, C., Bell, T. and Tanimoto, S. (204) "Should Your 8-year-old Learn Coding?", Proceedings of the 9th Workshop in Primary and Secondary Computing Education, 60-69.

Flórez, F. B., Casallas R., Hernández M., Reyes A., Restrepo S., Danies G. (2017) "Changing a Generation's Way of Thinking: Teaching Computational Thinking Through Programming", Review of Educational Research Vol 87, 834-860.

Gouws, L., Bradshaw, K. and Wentworth, P. (2013) "Computational Thinking in Educational Activities", Proceedings of the 18th ACM conference on Innovation and Technology in Computer Science Education, 10-15.

Liu, C., Cheng, Y. and Huang, C. (2011) "The effect of simulation games on the learning of computational problem solving", Computers & Education 57, 1907-1918.

Netto, D., Medeiros, L. M., de Pontes, D. and de Morais, E. (2017) "Game Logic: Um jogo para auxiliar na aprendizagem de lógica de programação", 25º Workshop sobre Educação em Computação.

Piaget, J. and Cook, M. (1952) "The origins of intelligence in children", New York: International University Press.

Shute, V. J (2011) "Stealth assessment in computer-based Games to support learning", Computer Games and Instruction, 503-523.

Webb, D. (2010) "Troubleshooting assessment: an authentic problem solving activity for it education", Procedia - Social and Behavioral Sciences Volume 9, 903-907.

Werner, L., Campe, S. (2012) "Children learning computer science concepts via Alice game-programming", Proceedings of the 43rd ACM technical symposium on Computer Science Education, 427-432.

Werner, L., Denner, J., Campe, S. (2012) “The fairy performance assessment: measuring computational thinking in middle school”, Proceedings of the 43rd ACM technical symposium on Computer Science Education, 215-220.

Wing, J. M. (2006) “Computational Thinking”, Communications of the ACM 49.3, 33-35.