

StickAndClick

sticking and composing simple games as a learning activity

Valente, Andrea; Marchetti, Emanuela

Published in:

Learning and Collaboration Technologies. Human and Technology Ecosystems

DOI:

10.1007/978-3-030-50506-6_24

Publication date:

2020

Document version:

Accepted manuscript

Citation for pulished version (APA):

Valente, A., & Marchetti, E. (2020). StickAndClick: sticking and composing simple games as a learning activity. In P. Zaphiris, & A. Ioannou (Eds.), *Learning and Collaboration Technologies. Human and Technology Ecosystems: 7th International Conference, LCT 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Proceedings* (Vol. 2, pp. 333-352). Springer. https://doi.org/10.1007/978-3-030-50506-6_24

Go to publication entry in University of Southern Denmark's Research Portal

Terms of use

This work is brought to you by the University of Southern Denmark.

Unless otherwise specified it has been shared according to the terms for self-archiving.

If no other license is stated, these terms apply:

- You may download this work for personal use only.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying this open access version

If you believe that this document breaches copyright please contact us providing details and we will investigate your claim. Please direct all enquiries to puresupport@bib.sdu.dk

StickAndClick – sticking and composing simple games as a learning activity

Andrea Valente^{1[0000-0002-6295-9511]} and **Emanuela Marchetti**^{2[0000-0002-7949-9953]}

¹ Mærsk Mc-Kinney Møller Institut, Embodied Systems for Robotics and Learning, University of Southern Denmark (SDU), anva@mmmi.sdu.dk

² Media, Department for the Study of Culture, University of Southern Denmark (SDU), emanuela@sdu.dk

Abstract. StickAndClick is a new digital tool meant to support teachers and pupils to define simple interactive digital contents. We propose to use a minimalistic and asset-based redefinition of coding to support algorithmic thinking, focusing on the creative and design-like aspects of Computational Thinking. Here we discuss the design and implementation of 3 prototypes of StickAndClick, and present the results from the first of a series of tests, and address the games developed by the children, game mechanics, usability and engagement with the prototype. We observed the emergence of game design dialogue, use-modify-create thinking, both important in a tool for scaffolding Computational Thinking. Our study is placed at the intersection between Computational Thinking and multimodal literacy, as both combine problem solving with the use of digital tools and the creation of digital content with the goal of fostering critical thinking and creativity.

Keywords: game-based learning, game design, learning ecosystems, Computational Thinking.

1 Introduction

The creation of simple paper-based games and multimodal narratives has become a broadly accepted practice in Danish schools, enabling forms of active learning. At the same time, systems like Minecraft¹ and Powtoon² (a digital platform to create animated comics) have become popular, as they enable the pupils to express digitally their creativity. Moreover, these systems have been introduced in the classroom with the goal of supporting learning of Computational Thinking (CT for short) in the classroom, defined as a set of skills from computer science [19].

A typical scenario that we observed in Danish primary schools' classrooms involves the teachers assigning groups of pupils the task to create a board game, related to a topic they have studied. For example, after having read and discussed a topic from a textbook or novel, the pupils might be given the task to design a game based on the discussed topic, modifying the classic goose game so to require the player to answer questions about the book or its author, in order to play and win. Typically, the

¹ Official website: www.minecraft.net

² Official website: www.powtoon.com

pupils are divided into small groups (from 3 to 5 pupils) and are given the task to design and test a variation a board game based on a topic introduced by the teacher. The games are then played by all, in turns, and feedback is given by the teachers and the other pupils. Sometimes the best games are also rewarded or given acknowledgement, but that is not the usual goal of the activity. In other cases, we observed competitions among classes, where one of the games per class is played by pupils of other classes. Again, winners might be rewarded or simply acknowledged, according to good-hearted forms of gamification.

Interestingly some of these activities have been digitized in recent years, by adopting free online tools, like *Kahoot!*³, which are appealing to schools because of their high usability, and because they afford easy-to-setup, class-wide networked games. However, we find the game aspects in these tools to be severely limited. Kahoot! in particular was originally only capable of supporting multiple choice questionnaires, even if recently it has been extended with more features, precisely to cope with the in-class-games we have observed [18].

Minecraft offers more possibilities to design and implement large worlds-like games, but to our knowledge it is mainly used in time-boxed workshops, often in connection to experts' visits (such as researchers or master students from universities, or professionals from IT companies). In these cases, the teachers host a workshop with their classes for the duration of a week, the experts take over the teaching and focus on demonstrating the CT potential of Minecraft. However, when the experts leave at the end of the workshop period, the teachers have issues with keeping using Minecraft in their normal teaching and revert to their usual lecture modalities.

In the past 10 years we have worked with Danish institutions, developing and testing e-learning tools [6], to expand the creative spectrum of learning practice in the classroom. Our goal has been to empower digital creativity and close the gap between tinkering in the physical world and authoring of game-like digital materials, which normally require a significant degree of programming skills. In this sense, our previous studies suggest that digital games are experienced by teachers and pupils as less creative than paper-based games, which can be easily redefined based on social agreement among teachers and pupils. Digital games on the other hand, come as black boxes, which can be played only in a particular way, decided by the developers, and cannot be customized to follow pedagogical needs nor created from scratch without advanced programming skills [6, 16]. However, during our studies [6, 15, and 16] teachers have systematically expressed the desire to edit or at least customize existing digital games, and to enable their pupils to customize or design digital games, to better meet their pedagogical needs. Moreover, Danish primary schools have engaged in different initiatives to enable their pupils to learn how to program, using systems like MIT's Scratch⁴, Minecraft or Google's Blockly⁵, and are currently in the process of formalizing CT as a subject. But these systems were often found too complex to use

³ Official website: www.kahoot.com

⁴ Official website: scratch.mit.edu

⁵ Official website: developers.google.com/blockly

in the learning of curricular subjects, therefore they remained limited to the practice of teaching pupils programming fundamentals.

Based on these insights, we have been working on re-contextualization and re-conceptualization of interactive digital contents, to make them more accessible to teachers, pupils and young students, in and outside the classroom. In this respect, we are investigating a new tool called *StickAndClick* (as documented in [14]) to support creative authoring of simple digital games, aimed at teachers and pupils, and to meet new curricular requirements, in introducing computational thinking as a subject and in connection to other curricular subjects. So far, we developed 3 different high-fidelity prototypes, our goal: to empower teachers and pupils, enabling them to create interactive content, and to expand the creative affordances offered by digital games and learning platforms.

In section 2 we present related work and studies that provide the basis for the design and implementation of the new tool. The requirements and inspiration are presented in section 3; our first and second prototypes are discussed in section 4, and the current one in section 5. Tests and discussion are in section 6, and section 7 concludes the paper, presenting ongoing and future work.

2 Related work and theoretical foundations

In our study we are working at the intersection of CT and multimodal literacy, as we aim at:

- promoting CT as a creative thinking approach that could be applied to the analysis and representation of any field of knowledge,
- supporting forms of multimodal literacy in schools, enabling teachers and pupils to create multimodal texts, seen as interactive representations of knowledge.

In our study we approach CT as an emergent competence in a symbiotic relation to multimodal literacy, combining problem solving with the use of digital tools and the creation of digital content. We see CT and multimodal literacy as rooted into each other. In this sense, CT can be cultivated and applied in schools, to the representation of knowledge across curricular disciplines.

CT has been defined as a subset of knowledge, mindset and skills from the computer science and engineering domain [19]. CT has been proposed as a fundamental component of literacy, together with writing and counting, including skills like: problem solving, design thinking, understanding human behavior drawing on fundamental concepts from computer science that would be desirable for “everyone” and not only IT professionals [19]. However, this perspective has been criticized for being “arrogant”, suggesting that everybody should aspire to learn to “think like a computer scientist”, no matter what they do [13]. Moreover, so defined computational thinking appears as overlapping, including or included within soft skills or the of the 21st century, such as: digital literacy, innovation and creativity, critical thinking, communication and collaboration, and also (digital) citizenship, self-regulated learning [17].

However, we believe that CT should involve also hardcore computational skills, such as: understanding of “algorithmic thinking, navigating multiple levels of abstraction, decomposing problems into manageable pieces, and representing data through models” [3, p. 1]. Hence, CT can be seen defined as “a collection of computational ideas and habits of mind” that are acquired by learning to make “software, simulations, and computations performed by machinery” [13]. In this sense, we argue that CT should include competent use of software tools and coding, at different levels of proficiency, applied to a set of problems originated within other fields than computer science. An interesting example is represented by digital humanities, in which CT is already applied to solve problems from the domain of the humanities [7].

Considering the multitude of definitions that have emerged in recent years, the meaning of CT is going through a complex elaboration process and it is still ambiguous [13]. We find that the main issue is represented by ambition levels for hardcore computational skills, such as proficiency level in the use of software and if coding should be a requirement at all. In our view, CT is emerging as an interdisciplinary field, including also creative thinking skills from the design field and collaboration from the management field. However, if CT is deprived of specific skills related to the process of using, testing, and constructing software, as it is defined in [19], CT becomes a mix of design and management skills, with superficial relation to Computer Science. Therefore, in our view, CT should strive to include understanding of the distinction between static code and running code (i.e. program versus process), the practice of coding, and the role of testing and debugging practices. We see CT as including passive competences of coding practice, similarly to comprehension skills in language learning, where pupils should gain an understanding of how software is being planned and constructed, and how programmers can imagine the program in action by looking at its code. In this way, pupils should become expert users of technologies, aware of how software works and how programmers think, able to communicate effectively and negotiate with programmers, according to their needs.

Starting from these premises, the relation between CT and multimodal literacy appears blurred and overlapping. Multimodal literacy is in fact defined as a set of abilities in creative and design thinking, communication, and competent use of digital tools for audio-visual production in collaborative settings [5]. In this way, CT and multimodal literacy can include one another. During our previous studies, we observed Danish teachers leveraging multimodal literacy as a complex entanglement, related to the analysis and creation of multisensory texts [5]. This takes place through creative activities, such as: design of games, quizzes on Kahoot, comics or simple animations, videos, paper based boardgames, and artistic artefacts such as posters, drawings, and sculptures. These activities leverage an ecological understanding of multisensory media creation [10], entangled with learning and sense-making processes. These activities are aimed at training critical thinking skills, bridging between children’s the living ecologies and their free-time interests and their learning ecology in schools [10]. Recent studies have showed that use of digital games, also commercial games found on Youtube and on the Internet, but not specifically developed for learning, can foster shared forms of critical thinking and sense-making, eliciting motivation and rich social interaction in the classroom [1]. Misfeldt and Zachø [8] have

explored how the design of interactive scenarios in relation to open-ended projects in mathematics, can foster creative engagement, collaboration, and peer-learning also supporting understanding. A study on movie editing on iPad has given similar results [9]. Therefore, designing and editing multimodal texts are found beneficial for the children to foster self-expression, collaboration, and creativity, which figure among the 21st century skills. At the same time, the production of multimodal texts provide escape from the black box of commercial digital games, empowering teachers and pupils in create new playful experiences and narratives for them and their mates. Moreover, multimodal literacy deals with critical analysis of complex semiotic messages, for instance engagement with graphic novels was found beneficial in training pupils to decode information in multiple sign systems. In this way, pupils learn how a story can be crafted and conveyed for different audiences [5]. The same study also discusses how teachers encourage pupils to watch movies in the classroom to acquire information and create films on their own, to communicate to the class their personal understanding on a given topic.

This multimodal approach is enabling teachers to provide their pupils with a rich expressive repertoire, also enabling them to become able to decode contemporary media. On this perspective, O'Halloran et al [11] have proposed MACT, a framework for multimodal analysis aimed at enabling primary school pupils to understand different mediatic genres, identifying main “features, structures, and ideas” in various texts in print and non-print sources, at making pupils able to “plan, organize, summarize and synthesize” information in an aesthetic whole [11, p. 15]. The ability of translating meaning from a school subject into a multimodal text, requires that the pupils gain a deep understanding on the subject. The production process will foster further questions, leading the pupils to iteratively analyze the subject in order to create a good text. According to Jewitt [4], creation of multimodal texts requires the pupils to familiarize with the affordances available in the different sensorial elements and to integrate them harmoniously. This in turn requires the pupils to acquire a meta-language enabling them to shift from one to another, understanding their cognitive, cultural and ideological relations [4], so to create an effective text.

In our study we aim at bridging between CT and multimodal literacy, as we find a significant overlapping between the two concepts and that they complement each other regarding creative and critical thinking, collaboration, and problem solving. At the same time, we find that there is a need to address the appropriate place for hard-core computer science skills (use of software and coding), but little investigation has been conducted on the interdisciplinary foundation of CT and how it is increasingly converging towards the fields of multimodality and digital humanities. Based on these premises, our study explores the connections between Ct and multimodality, through the design of StickAndClick, which we see as a multimodal editing tool, targeted the creation of multimodal texts in the classroom, specifically in the form of point and click games. StickAndClick has to be understood also as a mediational mean [Latour], bridging between CT and multimodal literacy as well as CT and curricular subjects. Moreover, the rules for editing games are aimed at concretizing a meta-language, in the terms of Jewitt [4], enabling the pupils and their teachers to reflect on how visuals, sound, game mechanics, and envisioned interactions can be harmoniously integrated

to create a multimodal text (the game) on a given subject, yet at the same time incorporating key skills in CT.

3 Inspiration and requirements

The initial idea came from the results of the tests conducted with paper materials and digital games [6] and Fables [7]. In both studies we used web-based prototype tools to let the pupils enact our scenarios, moreover, in previous work we used a set of static HTML pages with hyperlinks and images to reason about non-linear narrative as a designerly way to engage with web programming in higher education [6]. In both cases the pupils were able to author their own materials into digital, non-linear, and visual novels as a resource for learning other subjects.

From the point of view of game design, planning a visual non-linear story is often done via sticky notes and tangible materials, a set of activities that is very close to the typical creative tinkering we observed many times in Danish primary schools. However, the final goal of digital game designers is to create a digital game, while pupils and teachers usually would stop at table-top or pen-and-paper games.

The scenarios of use we define for our authoring tool are: one in which the teacher uses StickAndClick to design, implement and deploy a digital game to be used by her pupils in an augmented class situation or at home, according to flipped classroom teaching. In the second scenario the teacher assigns groups of pupils the task of designing and implementing a digital game about a topic that she covered in class. The pupils will be able to create, deploy and playtest their game with other groups, supporting possible peer-learning. In our attempts to find a metaphor that could help us explain our experiments to primary school practitioners and their pupils, we looked at *action transfers* and *sticker albums*, which despite being toys, can be easily re-appropriated as props and materials to be used in game design.

Based on our scenarios and results from previous studies, we define the requirements for our visual, digital game authoring tool:

- minimalistic computational model (i.e. hopefully easy to understand)
 - based on a variation on hyperlinks
 - no math (or minimal mathematical skills required), which in turn implies no (explicit) coordinates
- no universal programming language, so:
 - instead of imperative instructions, just simple before/after rules
 - limited interaction, which restricts us to simple *point and click* game development, or non-linear visual novels
- asset-first, to ground the meaning of programming in multimedia:
 - the tool should allow pupils and teachers to author only simple digital games, but they can use self-generated, custom images and audio
- focus on the software engineering/design-cycle parts of CT
- visual and possibly for web and/or android; our tool should export towards many (and more sophisticated) game development environments

Based on our previous, we decided to base StickAndClick on a tangible metaphor of transferable stickers and the interaction that these can afford, such as pasting stickers on a background picture. These stickers are user-created and can be used to construct narrative and support playful interaction. The games afforded by the system are turn-based and the computer cannot initiate actions, and the only game mechanic allowed is clicking on a sticker. StickAndClick is based on simple conditional rules, according to which pupils can define what happens when a sticker is clicked and also on the state of the stickers (to be specified in the actual prototypes) in the page (i.e. the current room in which the player finds herself in). A central idea in the design of StickAndClick is to look at existing game development environments (such as Scratch or Construct3⁶) and reduce their computational models to a single mechanism, as simple as possible, that we could use as a formal model for digital and interactive artefacts. In this sense, we are searching of a sub-universal meta-language [4] for point and click games, that could be simpler than in existing tools, yet affording the creation of game-like multimodal texts (a similar point of view can be found in the PuzzleScript⁷ project). So far, we have developed three different prototypes with different characteristics, in order to explore how we could reach a minimal meta-language, in which children could set the rules for their games, harmoniously integrating multimodal resources. In the following sections we describe all 3 prototypes: the first 2 in section 4, and the latest in section 5. For each we provide insights on our design process as a hands-on continuous exploration of the technological affordances offered by different platforms and programming languages adopted in each prototype.

4 First and second prototypes

4.1 First prototype – a javascript library

In our first experiment, conducted in 2018, we implemented StickAndClick as a JavaScript library, so we could start creating simple games. Each game consisted of a collection of rooms, each realized as a separate HTML page, with images representing items in the game. The main interaction mechanism was provided by images with HMLT anchors. We used simple CSS tricks so that all the pages were visualized in full-screen and responsive, allowing us to deploy on mobile devices from the start. Minimal persistence-like support was added, so that the state of every image in the page could be stored and retrieved from LocalStorage, effectively allowing a player to stop and resume playing. Since we were targeting mobile devices, such as tablets which are commonly used in Danish primary schools, we realized that our idea could easily express non-linear digital games in the style of point-and-click games.

In our scenario, the author of the game would define one of such games by doing the following:

- define a layout for each page, i.e. rectangular regions that we call markers,

⁶ Official website: editor.construct.net

⁷ Official website: www.puzzlescript.net

- decide which images (i.e. stickers) from a palette of self-created ones, will initially occupy which markers,
- then define rules for the game, so that when the player clicks on an image, 1 of 2 things could happen:
 - the game changes page (i.e. move the player to another room),
 - or the clicked image changes to another

Since this scenario could be summarized as “the game designer places stickers on pages and then the player clicks on them”, we named this system *StickAndClick*. The first set of experiments we performed with *StickAndClick* consisted in the creation of short **point and click** games, using our JavaScript library. We mainly performed functional tests and explored the expressivity of our computational model. One of the earliest games created is shown in figure 1. We purposefully kept the low-fidelity quality of the images, to emphasize that pupils will develop games from hand-drawn paper prototypes. In this “escape the room”-type game the player finds herself in a room with a closed cabinet and a locked door. Clicking on the cabinet opens a drawer and reveals a key; when the key is clicked, it disappears and appears to have moved to the status bar (top-left corner of the first room in figure 1). Clicking on the door initially has no effect, but when the key is in the status bar, then clicking on the door opens it (using the key, that disappears definitively from the game). The door, now open, can be clicked to go to a second room.

The most difficult part of the design of *StickAndClick* was to find a good yet simple way to express the rules avoiding the typical difficulties of coding; therefore, we attempted at describing this entire game with just 2 types of rules:

- RULE 1: cabinet-closed => cabinet-open-with-key
- RULE 2: cabinet-open-with-key , status-bar=>cabinet-open-empty , status-bar-key
- RULE 3: cabinet-open-empty => cabinet-closed-empty
- RULE 4: cabinet-closed-empty => cabinet-open-empty
- RULE 5: door-locked , status-bar-key => door_unlocked , status-bar
- RULE 6: door_unlocked JUMP_TO page2

Rule 1 is very simple, it specifies that when the image “cabinet-closed” is clicked, it changes to “cabinet-open-with-key” (all these images, or stickers, are visible on the right of figure 1). Rules 3 and 4 are the inverse of each other and allow the player to keep clicking on the cabinet repeatedly, opening and closing its drawer, resulting in a 2-frame animation. Rule 2 is a bit more complex, because it is contextual: it states that when “cabinet-open-with-key” is clicked AND the “status-bar” image appears anywhere in the current page, then the first image changes to “cabinet-open-empty” and the second changes to “status-bar-key”, at the same time. If both conditions are not met, the rule cannot execute. This purely textual way of expressing before/after rules is inspired by Join patterns [2], and it fits our requirement about avoiding explicit coordinates: to specify which images are involved in a rule, we can simply use its the name. Placement of images on the HTML page still depends on coordinates and sizes, but it can now be relegated to the definition of a *page layout*, made of *markers*; fur-

thermore, it is easy to imagine how markers can be defined visually in more advanced prototypes of StickAndClick.

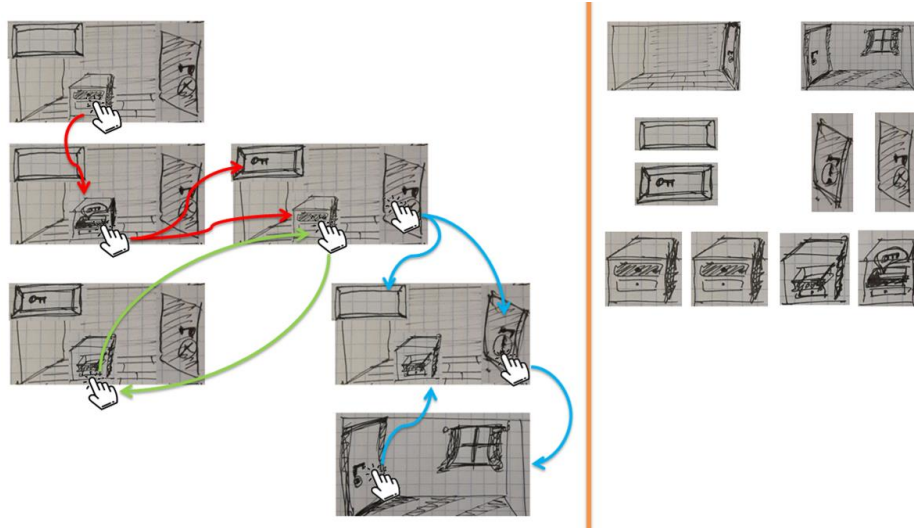


Fig. 1. On the left: the gameplay of the first game. On the right: the manually drawn stickers used in the game; from the top - 2 rooms, an item bar (empty and with a key), the door (opened and closed), and the 4 states of the cabinet (2 closed images, open but empty, and open with key inside).

The last rule, Rule 6, is of a different kind: it does not transform one or more images, but instead it makes the game jump to another room (i.e. page). Later in the design we decided to merge these 2 kinds of rules in a single rewrite-and-jump type of rule, where the jumping part is optional: in this way the whole gameplay can be described with just 1 type of rule.

A special situation might arise when the game author wants to make a sticker disappear: this is possible by (for example) changing an image of a dog with a placeholder. In StickAndClick a placeholder is name (in this example it could be “empty-dog”) that does correspond to any image. When the dog image changes into the “empty-dog” placeholder, it looks to the player like the dog disappeared since all placeholders are invisible. Moreover, we decided that a placeholder cannot be the target of a click, so no rule can directly change it back into another image. However, it is possible to write rules that change multiple images, including placeholders.

4.2 Second prototype – a node.js generator

After having worked for a few months with this first implementation, we wrote a *generator* in Node.js, capable of taking a JSON specification file and a folder of images, and create an entire website of static webpages, implementing the game de-

scribed in the JSON file. This file contains a serialization of the rules of a StickAndClick game, grouped by page; the generator creates one HTML page per game-page, and 1 initial page for the game, with a description of the game. When the game is reset, the player finds herself again at the initial page. Parts of the library developed for the first prototype are used by this generator that adds them to the game folder, together with images and HTML pages.

We developed the generator iteratively, and the final iteration was able to create a single-page application (SPA for short) for each game, with the advantage that the player would always resume from the correct place in the game. With the generator we wanted to address two limitations of StickAndClick: global images, and better persistence; we also wanted to explore the expressivity of StickAndClick’s rules.

4.3 Exploring expressivity via game creation

We decided to expand the scope and expressivity of StickAndClick by creating few games, each pushing the envelope of what the tool could express, and each game exploring more complex game mechanics. During these iterations, we relied also on feedback from a minimalistic, convenience focus group (a boy and a girl, age 13 and 14 respectively) who were periodically available to performing functional testing and evaluation with us. The games we created were: the “simple game”, the “lives” game, the “backpack” game, the “Flip a coin” and the “LCD” game.

The “simple game” has 2 rooms, and its goal is to escape by finding objects and unlocking the door. We used it to test:

- how to work with more than a single room,
- that the state of the game is correctly saved, and the game can be resumed by a single “initial” HTML page, after closing the game at any stage the player desires,
- the navigation between the 2 rooms,
- that rules can express the need for the player to have collected a few items hidden in a room, in any order she wants, in order to unlock the door that leads to the other room,
- that multiple items are place in the first room of the game (e.g. a large carpet and a sofa) in 2.5D, i.e. the order in which places are defined in each room, also defines the z-order of the images. The background images are therefore simply the first images to be added to a room definition.

The “lives” game also has simple rules: the player can click on her lives, represented by the 2 hearts (see figure 2), and decide how many should be enabled. The door from the first room to the second works each time it is clicked, but the door in the second room, to come back to the first, only works when the player has exactly 2 hearts. When the player moves from a room to the other, the amount of hearts stays the same: i.e. the hearts are a global value, persistent across rooms. The purpose of this game was to test how to use the StickAndClick rules to express global values. To allow for global values we used a HUD (or global transparent page) that is always drawn “above” the current page. The HUD has its own places that can be transparent or contain stickers, and its own rules. Moreover, the rules of every page can also involve

stickers in the HUD; in this case the image of the 2 hearts is in the HUD, and not in the 2 pages of the game. The rule that decides whether the door in the second room should open or not looks like this:

- `doorClosed , 2lives => doorOpen , 2lives`

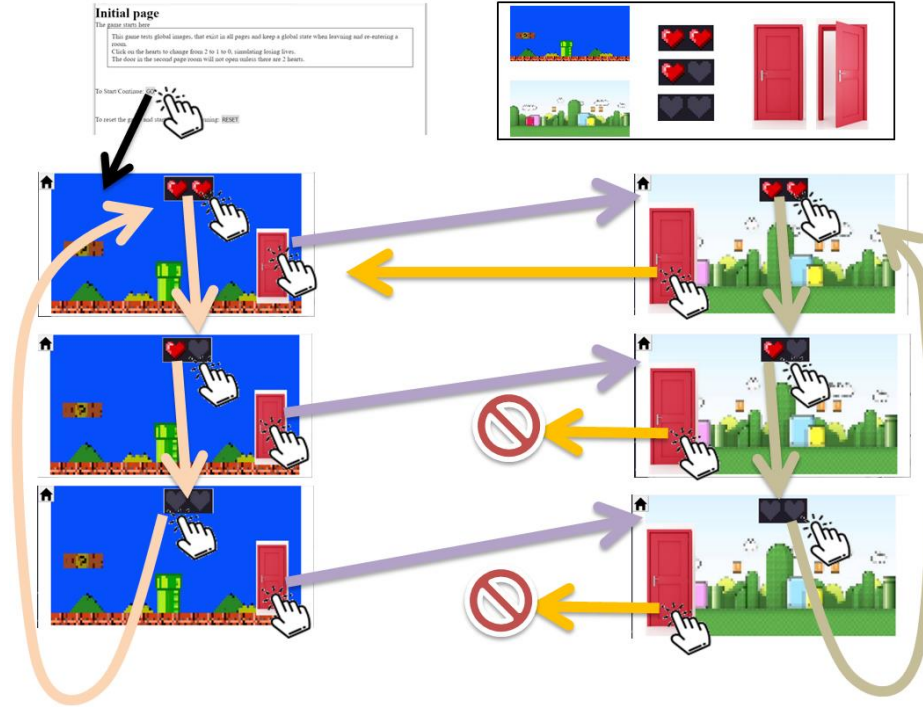


Fig. 2. The gameplay of the “lives” game. The stickers used in this game are visible on the top-right corner.

which reads as “if the **doorClosed** image is clicked, and there is another image anywhere in the page, called **2lives**, then change **doorClosed** into **doorOpen**, and change **2lives** in **2lives**”, i.e. the door opens and the 2 lives stay as they were.

The next was the “backpack” game. In this game the player stands in a room with shelves and can zoom in 2 of the shelves. The player also has a “backpack” area on the screen, inspired by head-up display (HUD) and backpacks in RPG games. The player can explore zooming in and out of the 2 shelves, and when “inside” each of the shelves she can decide to leave or pick-up items from her backpack. The goal with this game was to test navigation inside realistic-looking settings (in fact all images in this game are actual photos of shelves and objects) and how the rules can be used to express the workings of a “global” backpack, that moves with the player across multiple rooms.

So far, all rules in StickAndClick are deterministic, which means that it is impossible to express random behavior in games created with our tools. Therefore, we created

the game “flip a coin”, to clarify how to implement random-choice rules: in this game the player tosses a coin and a random result is displayed. We expect that pupils will be introduced to deterministic rules first, and when they can master them, to random rules.

Finally, we created the “LCD” game, to see if StickAndClick could express much more than just point-and-click game mechanics. The game (shown in figure 3) is based on a paper prototype we created and digitalized, and its gameplay is inspired by LCD games⁸. The game is single-page and has 2 possible endings: for this we extended the rules with a “win” and a “lose” command, so that when the player loses (or wins), the game shows a pop-up message and automatically resets. The apparent movement of the player’s avatar (the warrior character in figure 3) is realized by using placeholders. The game defines markers for the warrior in 5 locations: at the far left of the page, by the on/off lever, on the bridge, under the bridge, and at the far right of the page (i.e. the victory location). The rules are defined in such a way that at any given moment, 4 of these markers are empty and 1 contains the image of the warrior.

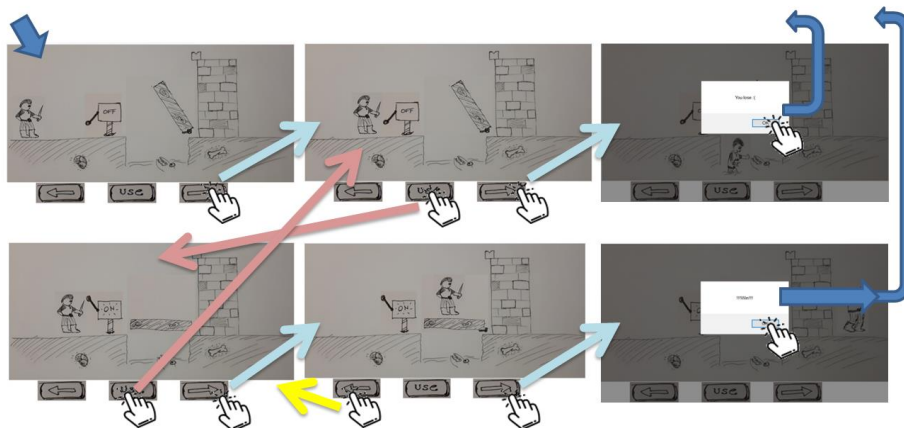


Fig. 3. The LCD-inspired game. The controls appear to the player as functioning buttons, but they are based on the same rules as in all other StickAndClick games. Notice the win and lose pop-ups (on the right side).

To summarize, at this stage of development StickAndClick offers:

- a single-page application, with reset capability and persistence of the game state;
- definition of multiple rooms and custom graphics;
- a special HUD-like page that allows games to have global stickers, useful when implementing backpacks and lives;
- a unified way to express rules for:

⁸ More information at: archive.org/details/handheldhistory

- changing an image (when clicking on it), or many images together depending on the dynamic state of (some of the) images in the same page,
- jumping to another page (when clicking on an image)
- choosing randomly among different outcomes
- “invisible” placeholders to be used in place of stickers, to make them disappear

StickAndClick however, does not have audio capabilities nor can express animations, such as an image shaking or rotating. We have received feedback about those issues during our testing with the 2 participants to the focus group. Audio and animations will be added in the P5 prototype, section 5, together with a visual editor for StickAndClick.

5 Current web-based prototype in P5

We considered implementing StickAndClick in various platforms; we made spikes in Processing, Java (using Android Studio), and Python 3 (attempting to take advantage of the Pygame Zero library). However, an important feature of our new tool is that it should be the ability to run online and on mobile devices, so finally we opted for P5⁹, a JavaScript implementation of Processing.

The new web-based prototype of StickAndClick is composed of 2 web-apps: an editor and a player. Figure 4 shows an early design of the proposed GUI for editor: this design is mobile-first, so most operations are based on drag-and-drop or tap-like single clicks. The actual GUI of the web-editor is visible in figure 5, where the game “Boy and Key” is defined visually.



Fig. 4. Early design of the GUI for the new StickAndClick web-editor.

We designed the workflow of a pupil creating a game with the StickAndClick web-editor as follows: first the pupil uploads her custom images to the web editor, then layout and rules are defined, and the game description can be saved. Only games with 1 page can be defined for now, and therefore we did not implement the global HUD-like backpack; rules are defined in a visual before/after fashion, without the use of block-based code. Saving a game generates 2 files: a JSON file with the description of the game, and an image file containing a vertical list of all stickers used in the game

⁹ Documented at p5js.org

(the atlas file). The web-player can then be used to open the game files and play. The save/load capability allows pupils to exchange their games or play games generated by their teachers. Other features are:

- game author can create a game using her custom stickers images,
- only deterministic rules are implemented,
- a short animation (from a predefined list) can be associated to a rule,
- the author can select a background image and/or background sound in the editor,
- the web-player execution the rules and triggers animations when needed; it also signals the player when she wins or loses the game.

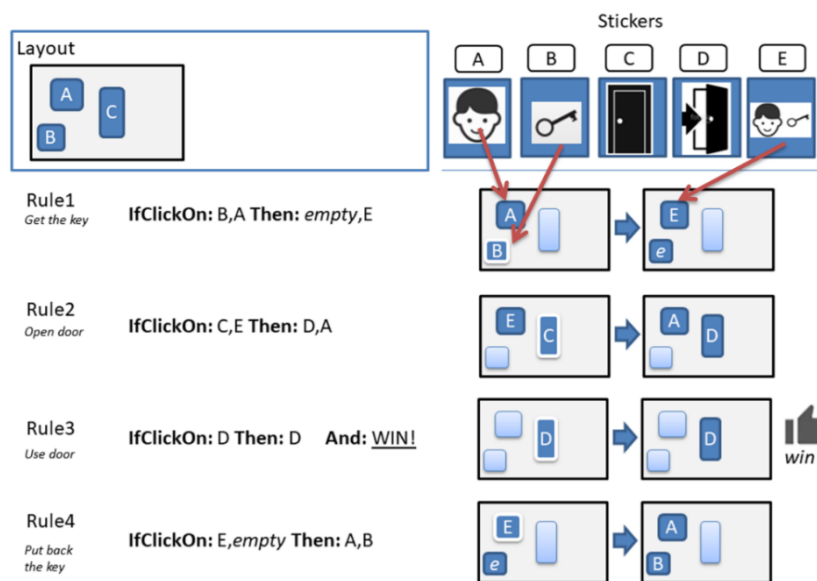


Fig. 5. Visual definition of the "Boy and Key" game.

The main limitation with respect to the previous, node.js versions is that only small images (less than 20kb each) can be uploaded.

6 Tests and discussion

The latest of our 3 prototypes was tested in spring 2020, with a group of children, from a class of 8 to 12 years old: the test was part of Teknologiskolen (translated as "technology school", TS for short), an afternoon activity organized by SDU to promote digital literacy and skills in the area of CT. Our test aimed at evaluating how the children related to our approach to game design and if they could make sense of the tool. We wanted to verify if primary school children could understand the computa-

tional model behind StickAndClick (i.e. a simple extension of the hyperlink). At the same time, we wanted to gather data on the usability of our P5 prototype, in order to improve its features. Finally, we wanted to analyze which kind of games the children would develop, which game mechanics and narratives they preferred and how to better support their engagement with our prototype.

Methodologically the test was based on ethnography [12]: we observed the children while engaging with our system, as we took notes. Since we did not have the permission to film the children, we recorded only their voices, took notes and sketches of their actions. In this way, we could collect data and documentation material for our test without violating their privacy. The test was conducted in a semi-structured fashion, as we gave the children loose tasks in trying out the game examples and in creating their own games.

Preliminary results show that StickAndClick can be understood and used productively by our intended target audience, and that testers attempted to create games beyond the point and click genre. We observed emergence of game design dialogue in the group and use-modify-create thinking, both of which match the intent of our tool; we believe that these elements are important in a tool for scaffolding Computational Thinking. This was only the first of a series of tests we have planned for the TS classes and in local, Danish primary schools; we intend to promptly start a new iteration of development, bringing improvements to the current version of the prototype, based on the gathered data.

6.1 Test procedure and results

Our test lasted for about 1 hour and involved a group of 6 boys around 12 years old. The test was articulated in approximately three stages, such as:

1. introduction to the interface,
2. free play with one or more of the example games,
3. creation of new games,

During the first step, introduction, the boys were introduced to the interface and basic functionalities of StickAndClick (figure 6). We showed StickAndClick projecting from our computer to a big screen on the wall, we ran through the system, describing and demonstrating its functionalities, such as: Player, Game Examples, the Editor and the Assets. The Game Examples included a set of 5 pre-made mini games of different genres and using various visual assets. The first is called “Boy and Key” (a variation of the game defined in figure 5) and it is an “escape the room” game showing a closed room with a door and a boy-looking character, who has to find a key in order to open the door and leave the room. The second game is “CastleGame” (a version of the “LCD-inspired game” described in section 4.3), a fantasy platform game in which a character has to discover how to open the drawbridge to enter the castle. The third is called “Mix”, a digital toy inspired by dress up games, in which the players have to mix and match heads, torsos and legs of three different robots and/or aliens. The fourth game, called “Animals”, is intended as an interactive book

about animals, in which a player can learn about the voices of different animals; the *voices* are not actual audio files, but rather images of text bubbles.

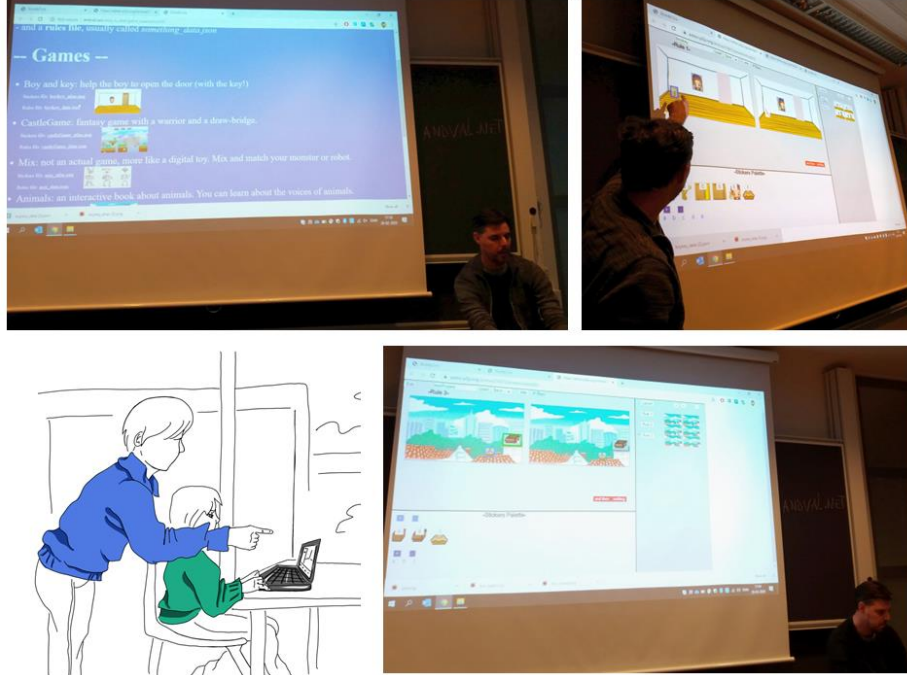


Fig. 6. Introduction of the functionalities of StickAndClick to the children.

The last game is called “Combination” and it requires the players to answer a question to find the combination of a digital lock, and win. All the games are available online¹⁰ and the children could simply access them on their laptops through the URL we gave to them. During the first stage, we also demonstrated how to play the games and how to edit new games, introducing stickers and setting up rules, as well as how to handle the 2 downloadable files containing the rules and the graphic assets. This part was rather short, about 5 minutes, as the boys were eager to start engaging with the system on their own.

Moving towards the second stage, we encouraged the boys to start playing on their own. The boys chose among the games based on their own taste: most boys started with the “Boy and Key” game, which one of the boys recognized as an “escape the room game”. A couple of boys played mostly with “CastleGame”, another boy tried them all but then used his remaining time playing with “Mix”, while the other boys spent their time trying all the games. The length of this stage varied according to the children’s needs, as soon as they were satisfied with playing around with the games, they shifted into editing the games. Generally, our data shows that the boys did not

¹⁰ At StickAndClick’s website: http://andval.net/stick_n_click/game_examples.html

have any issues with the player's interface, they did not ask for help and that stage ran smoothly. As expected, challenges emerged during the third stage, in which the boys were supposed to create their own games.

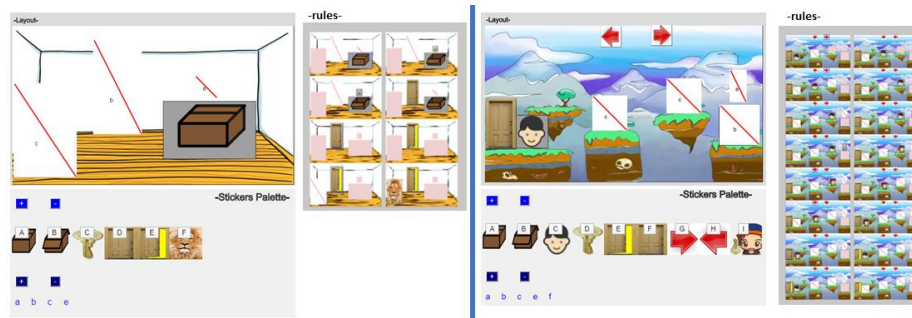


Fig. 7. Two games created by the children during the test. On the left the “Jack in the Box” game. On the right the platformer game, with its many rules.

During our observations we noticed that the visual assets we provided affected significantly the creativity of the boys, suggesting specific game genres and game mechanics. All the boys started playing with the room background that we employed in the “Boy and Key” game, which shows the inside of an empty room (the background in figure 7, on the left). It seemed that they used that background to play around with basic interactions, as a *safe room* to experiment with the basic functionalities of the editor and game mechanics, using only a few of the provided assets. Interestingly, one boy created a platformer game, in which a character had to open a box to find a key; after the key was found it appeared on the top of the open box (figure 7 on the right). So instead of an “escape the room” game he ended up creating a third person version of the Jack in the Box toy, in which a character finds a surprise object popping out of an initially closed box.

Afterwards the boys played with the platformer background we employed in the “CastleGame” and the urban background of the “Animals” digital book. One of the boys, made a different version of the “Animals” game: he went searching for images of other animals online and then he had fun mixing them with the wrong sounds. Another boy used the background from “CastleGame”, the game he played the most during the second stage. He created an articulated version of an “escape the room” game with many rules, in which a character started near a door, on one side of the screen, and had to go all the way across to collect a key, come back to open a door. This game was structured as a typical bonus level that can be found in platform games, in which a player has the opportunity to collect different bonuses, like coins or lives, without fighting enemies, and once the bonuses have been collected the player can go back to the regular levels.

The children were generally concentrated on engaging with the system and seemed to have understood well the basic functionalities and affordances offered by StickAndClick, however, they also encountered a series of difficulties. We need to consider that since the test was conducted in the late afternoon (around 6 to 7 pm),

towards the end of the test the boys started to get tired. Hence, a few of them, especially two boys sitting in the back got distracted, one continued to play with StickAndClick while chatting and looking at his mate screen, who instead shifted from testing out our system to watching videos on the Internet, at time playing loud music to be noticed by the others. However, we were able to carry out the test and we got inspiration to improve the interface.

We expected that the children were going to be confused about downloading and uploading the 2 files to edit their games, but in fact they managed that part quite well, without any issues. On the other hand, one of the boys in the back criticized the system for being “too easy”. He said this smiling with self-confidence, meaning that it was too easy for him since he said: “I have already tried Python!”. He also suggested inserting a dropdown menu for selecting the musical backgrounds, as the one the prototype already has for the background images. Four boys had issues with understanding how to model the sequence of steps in editing the rules for the game, not being sure how much could happen in the same rule. But in general, through trial-and-error experiments they could figure it out; however, we could reflect on how to simplify the process of editing rules or provide an embedded tutorial/guideline.

We noticed that the boys had issues with handling the various kinds of buttons found in the StickAndClick interface. We have a “plus” button on the top left area to add a sticker to the layout panel, and another “plus” button in the bottom area (the palette panel, containing all stickers for the current game), which allows the game author to add new stickers to the stickers’ palette. Each sticker in the stickers’ palette also has a button, to edit that sticker, i.e. to upload a different image for all copies of that sticker in the current game. Finally, double-clicking on any sticker in the layout (or selecting a sticker and clicking the “edit” button of the layout panel) opens a popup and lets the game author change the size of the selected sticker. We could see that this workflow is quite complex and the boys seemed confused about the different roles of the “plus” buttons, since these buttons look the same but are placed in different areas, therefore, the children expected correctly that these buttons had a different function but were unsure how to relate to them. Hence, we observed them clicking back and forth on those buttons to understand their function. We should re-conceptualize this workflow in future versions, where we think that we should develop a drag and drop system, so that after stickers have been uploaded in the bottom palette, they should be simply dragged to the layout panel.

In conclusion, we found that the workflow in the system provided a meaningful meta-language for engaging in editing multimodal game-like texts, as in [4], however, it needs further simplification; however, the system was fairly easy to learn and understand. Moreover, we found interesting how the provided backgrounds, actively inspired possible game genres and game mechanics, fostering creative thinking. In this respect, in a future test we might add more visual assets, such as backgrounds, characters, enemies and objects, photographed and drawn, to analyze how different visuals can be interpreted as affordances, inspiring specific game concepts and interactions, ultimately contribute to children acquisition of the meta-language needed to effectively engage in the creation of multimodal texts.

7 Conclusion and future work

In this paper we have addressed the central challenge of simplifying the practice and logic behind programming simple point-and-click games, to be accessible to teachers and pupils, without previous programming knowledge. Even if StickAndClick is not meant to become a universal programming language, and it not intended to be used in the creation of games outside the point-and-click genre, we have shown how to support multiple rooms, implement global values, and extend the rules to non-determinism.

Our early test data shows that the children did not have problems with the prototype's interface and the save/load mechanism; they also quickly grasped the idea of our simple click-inspired game mechanic and the stickers' model behind it. However, the workflow for adding and editing stickers in the games requires simplification.

Finally, further experiments are planned involving Teknologiskolen and classes of primary school pupils.

References

1. Fenyvesi, K.: English learning motivation of young learners in Danish primary schools. *Language Teaching Research*. Sage Journals (2018).
2. Fournet, C., & Gonthier, G.: The reflexive CHAM and the join-calculus. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pp. 372-385 (1996).
3. Jacob, S.R. and Warschauer, M.: Computational thinking and literacy. *Journal of Computer Science Integration*, volume 1 (2018).
4. Jewitt, C.: Multimodality and Literacy in School Classrooms. *Review of Research in Education*, 32(1), pp. 241-267 (2008).
5. Lenters, K.: Multimodal becoming: Literacy in and beyond the classroom. *The Reading Teacher*, 71(6), pp.643-649 (2018).
6. Marchetti E. and Valente A.: Learning via Game Design: From Digital to Card Games and Back Again. *The Electronic Journal of e-Learning*, volume 13(3), pp. 167-180 (2015).
7. Marchetti, E.: Bildung and the Digital Revolution. In D. Remenyi, K. A. Grant, & S. Singh (Eds.), *The University of the Future Academic Conferences and Publishing International* (2019).
8. Misfeldt, M. and Zacho, L.: Supporting primary-level mathematics teachers' collaboration in designing and using technology-based scenarios. *Journal of Mathematics Teacher Education*, 19(2-3), pp. 227-241 (2016).
9. Møller, T. E.: Collaborative learning through film production on iPad: Touch creates conflicts. *The Digital Literacy and Multimodal Practices of Young Children: Engaging with Emergent Research*, *Proceedings of the first Training School of COST Action IS 1410*, pp. 127-134, University of Minho, Braga, Portugal (2016).
10. Nardi, B. A., & O'Day, V.: *Information ecologies: Using technology with heart*. Mit Press (1999).

11. O'Halloran, K. L., Tan, S. and E, M. K.: Multimodal analysis for critical thinking. *Learning, Media and Technology*, 42(2), pp.147-170 (2017).
12. Pink, S.: *Doing Visual Ethnography*. Monash University (2013).
13. Tedre, M., and Denning, P. J.: The Long Quest for Computational Thinking. *Proceedings of the 16th Koli Calling Conference on Computing Education Research*, November 24-27, 2016, Koli, Finland, pp. 120-129 (2016).
14. Valente, A., & Marchetti, E.: The road towards friendly, classroom-centered interactive digital contents authoring. In *27th International conference on computers in education*, pp. 38-46, Asia-Pacific Society for Computers in Education (2019).
15. Valente, A., & Marchetti, E.: Kill it or Grow it: Computer Game Design for Playful Math-Learning. In *2012 IEEE Fourth International Conference On Digital Game And Intelligent Toy Enhanced Learning*, pp. 17-24, IEEE (2012).
16. Valente, A., & Marchetti, E.: Fables for Teachers and Pupils. In *International Conference on Human-Computer Interaction*, pp. 206-224. Springer, Cham (2019).
17. Van de Oudeweetering, K. and Voogt, J.: Teachers' conceptualization and enactment of twenty-first century competences: exploring dimensions for new curricula, *The Curriculum Journal*, 29(1), pp. 116-133 (2018).
18. Wang, A. I.: Jumble vs. Quiz - Evaluation of Two Different Types of Games in Kahoot! In *proceedings of the In 13th European Conference on Games Based Learning*, Academic Conferences International, Odense, Denmark (2019).
19. Wing, J.M.: Computational thinking. *Communications of the ACM*, 49(3), pp.33-35 (2006).