

Programming playfully for a real-life problem: conditional statements on the stage of Scratch

Rene Alimisi, *ralimisi@ioe.ac.uk*

Department of Preschool Education, University of Thessaly

Niall Winters, *n.winters@ioe.ac.uk*

London Knowledge Lab, Institute of Education, University of London

Abstract

The literature has shown that students have problems with certain programming concepts, including the concept of variable (Doukakis *et al*, 2007), conditional statements, repeat structures (DuBulay *et al*, 1989; Putman *et al*, 1989; Soloway and Spohrer, 1989). In this study, we present the design of an activity based upon the 'Kindergarten approach to learning' (Resnick, 2007b) and the 'thick view of authenticity' (Shaffer and Resnick, 1999) using Scratch programming environment. The study aims at exploring 12-13 years old students' understanding of conditional statements. A 'thick view of authenticity' is taken by design a learning activity that is based on a real- life scenario: the functionality of a lift. The ways in which the activity design supported students' understanding and the role that Scratch played in this are explored and discussed. The study found that the four novice programmers, who took part in the study, gained a good level of understanding of the way in which the conditional statements function. The paper details the importance of the interrelation of the real- life scenario and the functionality of Scratch programming environment in supporting the playful exploration of programming solutions and the learning process.

Keywords

Constructionism, Kindergarten approach, real life scenario, programming, playful learning, Scratch

Introduction

Programming is ‘one of the most widely practiced instructional activities’ (Lee and Lehrer, 1987), providing potential learning opportunities for students (Papert, 1993). Papert (1993, p.27) advocates that programming encourages students to explore, reflect upon and develop their own ‘style of thinking’. Kahn (2004, n.p), taking into account Papert’s work as well as other important studies in the area of computer programming, parallels the process of programming to ‘a fertile ground for learning general thinking skills’ such as ‘problem decomposition, component composition, explicit representation, abstraction, debugging and thinking about thinking’.

However, engaging students in introductory programming is not straightforward (Guzdial, 2003). Studies have shown that novice programmers face difficulties in understanding basic programming structures (Doukakis *et al*, 2007; Soloway and Spohrer, 1989). As Pea (1986, p.25) posits, ‘students have such pervasive conceptual misunderstandings as novice programmers that correct programs early in the learning process come as pleasant surprises’.

According to Doukakis *et al* (2007), ‘conditional statements’ are among the programming concepts that cause difficulties to students. A conditional statement is a structure comprising of commands. These commands will be executed upon evaluation of a TRUE/FALSE condition. If the specified condition is TRUE, a set of commands will be executed. Otherwise, if the specified condition is FALSE, another set of commands, possibly embodied in the ‘else’ part of the structure will be executed. For instance, if the *condition* evaluates to true, *statement_1* is executed; *statement_2* is executed only in case the *condition* evaluates to false (see figure 1).

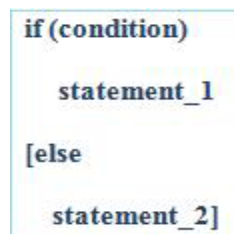


Figure 1. Conditional statement

Taking into account that conditional statements are a fundamental concept in programming languages, it is important to look behind novice programmers’ difficulties. Complex programming environments and traditional programming languages that are usually used for introductory programming courses (Pea, 1986) are often laid behind students’ difficulties and confusion. DuBulay *et al* (1989), identify simplicity and visibility as crucial characteristics for programming languages for novices. The kind of the problems, that students are called to program for, plays also a significant role towards their understanding. Commonly, students are called to work on and to program for mathematical problems disassociated with real life situations (Tzimogiannis, 2005). However, it is of great importance to allow students to work on computational artefacts that are meaningful to them (Guzdial, 2003). Given that students often say that they feel fascinated to work with real-world problems (Mims, 2003), it is worth providing them with the opportunities to program for such problems.

This research focuses on novices’ understanding of the programming concept of ‘conditional statements’. It aims to engage students in a playful learning experience that draws upon the Kindergarten approach to learning and the thick view of authenticity (Resnick, 2007b; Shaffer and Resnick, 1999) using Scratch. A ‘thick view of authenticity’ is taken by design a learning activity that is based on a real- life scenario: the functionality of a lift.

This paper addresses two main issues. First, the learning activity was designed as a real life scenario; the way in which the use of the real life scenario supported students’ understanding of the programming concept of conditional statement, is examined. Second, the way the

programming learning environment of Scratch supported students' understanding of the programming concept of conditional statement is discussed.

Theoretical framework

Resnick (2008) claims that we should strive for a 'Creative Society'. The vision of the Creative Society lays emphasis on the ability to think creatively and such ability constitutes the key to success at both a personal and professional level (Resnick, 2007b). The design and development of new things are seen to play a fundamental role in the concept of creativity and to encourage the revising of the models that are already in one's mind. Two pedagogical approaches closely associated with the idea of creative thinking and capable of meeting the needs of the current society are the 'Kindergarten approach to learning' and the 'thick view of authenticity'.

The Kindergarten Approach to learning is based on the 'creative thinking spiral' (see figure 2) introduced by Resnick (2007a, p.18). The so-called 'creative thinking spiral' (Resnick, 2007a, p.18) is used to describe a process in which children *imagine* what they want to do, *create* a project based on their ideas, *play* with their creations, *share* their ideas and creations with others, *reflect* on their experiences – all of which leads them to *imagine* new ideas and new projects'. The engagement in such a process is seen to encourage the development of creative thinking skills and purportedly to form the fundamental steps towards the Creative Society; or in Resnick's (2007a) words 'to sow the seeds for a more creative society'. Resnick aims at providing students with opportunities to learn through designing, creating, inventing and reflecting, and such an aim seems to have its roots in the Papertian 'powerful ideas'. (Resnick, 2008; Resnick and Silverman, 2005; Papert, 1993, p.4) To Papert these ideas 'can be used as tools to think with over a lifetime' and provide the leverage to help students 'make sense of the world' (Papert, 1980 cited in Resnick and Silverman, 2005, n.p).



Figure 2. The creative thinking spiral (Resnick, 2007b)

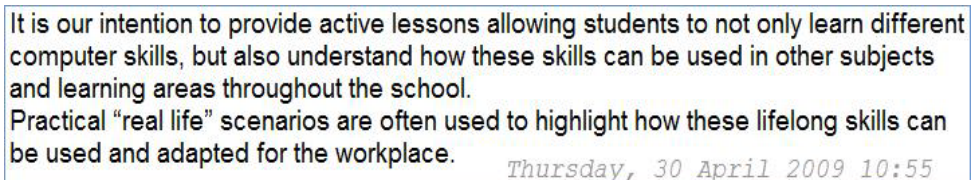
Shaffer and Resnick (1999, p.195) introduce the 'thick view of authenticity', according to which authentic learning is personally meaningful to the learner (*personal authenticity*), is closely associated with the world outside the classroom (*real world authenticity*), offers the opportunity to 'think in the modes of a particular discipline' (*discipline authenticity*) and embodies means of assessment which reflect the learning process (*authentic assessment*). This learning approach encourages students to experiment with knowledge in context and allows them to make connections with the real world (Shaffer and Resnick, 1999). Taking into account the fact that students often say that they feel motivated to work with real-world problems (Mims, 2003), the 'thick view of authenticity' can be seen as a way of engaging students in meaningful activities.

The idea underpinning this study is to engage students with the programming concept of conditional statements through a framework that would allow real-life connections and a creative process as it is described through the 'creative thinking spiral' to occur. Moreover, it is critical to provide students with the opportunity to work on a programming environment which eliminates the complexity of traditional programming environments and supports the engagement with meaningful activities.

Scratch, 'a networked, media rich programming environment' built upon Logo and created by 'Lifelong Kindergarten Group at MIT Media Laboratory in collaboration with Yasmin Kafai's group at UCLA' (Maloney *et al*, 2008, p.367) was chosen as it was seen as capable of engaging students in the different stages of the 'creative thinking spiral' and supporting the thick view of authenticity, which have both been seen to play a key role in the development of students as creative thinkers. Apart from primary reasons, pragmatic ones shaped also this decision. Firstly, programming in Scratch is simplified without degrading the mental process which underpins it. Secondly, the way in which blocks are joined together, eliminates the possibilities for syntactic errors to occur. Thus, students are allowed to focus on their projects without spending a considerable amount of time on syntactic issues.

Methodology

The research was carried out in a secondary school in South London that is specialist in Mathematics and Computing. It was gratifying to know that the ideas underpinning this study could be useful for the school and that future findings could be utilised towards meeting its needs. In particular, the attention paid (see figure 3) in the exploitation of real-life scenarios in the classroom as well as the focus on innovative Information Communication Technologies applications, made the school ideally suited to the purposes of the study.



It is our intention to provide active lessons allowing students to not only learn different computer skills, but also understand how these skills can be used in other subjects and learning areas throughout the school.
Practical "real life" scenarios are often used to highlight how these lifelong skills can be used and adapted for the workplace. Thursday, 30 April 2009 10:55

Figure 3. Screenshot from school's webpage

The project follows a case study approach and occurs in four stages. The first stage employs questionnaires and involves informal discussions which aim to outline students' pre-programming experience and to identify the four novice programmers, participants of the study. The four 12 to 13 years old students that were selected (Kevin, Timothy, Luke and Billy) shared two common characteristics. First, all of them had not been taught any programming language either during school or out of school hours. Second, all of them had previous experience in designing games and mainly animations by making use of particular software applications such as Pivot, Power Point and GameMaker. The question that arose was whether through Pivot, Power Point or GameMaker the students had been involved with programming concepts. Interestingly, it became clear during the informal discussion stage that their experience was not in any way associated with programming concepts and in particular with the programming concept of conditional statements. In fact, students' applications did not demand the use of significant programming concepts, such as conditional constructs. In addition, it is worth mentioning that the creation of the games and the animations occurred through a practice based on 'dragging and dropping' or 'clicking and selecting' pre-programmed behaviours. This conclusion was reached through the students' statements and confirmation from the class ICT teacher.

'I moved the person and I just clicked a button and saved it; and then I clicked another button; and it played it and made it move slowly' (Kevin on 'how he made an animated stickman on Pivot')

The second stage consisted of a number of familiarization activities through which the four participants would have the opportunity to engage with Scratch at a level where they could carry out the main activity. The programming concepts and commands that were necessary for the accomplishment of the main activity constituted the target- content during the familiarization stage. The different programming concepts that would be employed in the main activity were introduced as ways to 'breathe life into the sprites'. Thus, blocks which resulted in making sprites move or display messages or reproduce sounds were presented first. Then, the introduction to the programming mechanisms for synchronizing and controlling the sprites followed. The programming concepts were introduced in the framework of a simple task (the movement of a sprite) which included several steps. In the framework of the familiarization stage, we worked with all the four students on one laptop. However, students were often moved to practice the newly introduced programming concepts individually.

The main activity is carried out in the third stage (for more details see the subsection below). The method of observation was used for gathering data during students' engagement in the main activity. During the stage of observation field notes were recorded. A flip camera and a digital audio recorder were also utilised in order to record parts of students' progress while they were developing their programming solutions. The recorder was also used in the fourth stage where semi- structured interviews with the four participants took place. The interviews aimed to explore students' perceptions of the programming concept as well as to achieve triangulation of the data.

Activity Design

According to the rationale, it was intended that the activity would draw upon a real- life scenario, the implementation of which to be based on the use of conditional statements. The real- life problem of the functionality of the lift was considered suitable in order for a programming solution to be drawn based on the programming concept of conditional statements.

Drawn upon Alexopoulou and Kynigos (2008) study, the idea of the half- baked approach was exploited as it could guarantee the frame needed and the context in which students could construct and explore the concept of conditional statements. The semi- finished nature of the approach could also guarantee that a level of freedom could be given to the students to develop their own thinking and their own programming solutions. In order to restrain students' cognitive load, it was opted to provide them with parts of code that was not relevant to the programming concept of conditional statements; students were free to experiment with this part of code in order to manage to compose a solution putting together the different parts of the 'puzzle' (see figure 12).

The main activity based on the scenario of the lift, consisted of two parts. The purpose of the first part of the activity was to snap together the appropriate blocks so that the lift will function according to the buttons pressed. The buttons '0', '1', '2' (see figure 4) which represented the ground floor, the first floor and the second floor had already been programmed in the framework of the idea of the half- baked approach (see figures 5,6,7). In accordance to the activity: If button '0' is pressed, the lift will move to the ground floor and the message 'ground floor' sounds or appears. If the lift is already on the ground floor, the message 'already on ground floor' will be displayed. Similarly, buttons '1' and '2', if pressed, function following the same concept. *The solution addressed to the problem is represented in figure 8.*

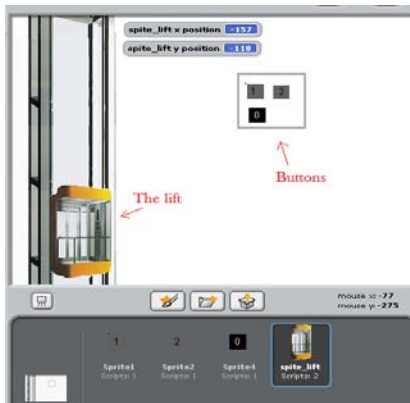


Figure 4. The components of the activity



Figure 5. Script for button '0'



Figure 6. Script for button '1'



Figure 7. Script for button '2'

The second part of the activity was more advanced as it has been designed in a way according to which students should use nested conditional constructs (see figure 11) in order to solve the given problem. This part was based on the first part of the activity which was further enriched with the so called 'risk button' (see figure 9).

The task for the students was to snap together the appropriate blocks in order to put the lift temporarily out of order only when the risk button is pressed and the risk exists. If the 'risk button' is pressed, the lift does not move and is temporarily unavailable (even though button 0, 1, 2 will be pressed). The message 'lift is unavailable' appears, followed by the recorded message 'lift temporarily unavailable due to technical problems'. After a specified amount of time the lift becomes available and functions normally according to the buttons pressed.

The risk button had been programmed for the students and further explanations were given to each of them individually about the idea underpinning this script in order to ensure that students were aware of the way the variable 'risk' is used. Obviously, the variable risk is set to one for a specified amount of time when the button risk is pressed (see figure 10). After this amount of time the variable is set to zero which means that the risk does not exist (see figure 10). *Written instructions, closely associated with the concept of the two parts of the activity, were also delivered to students.*

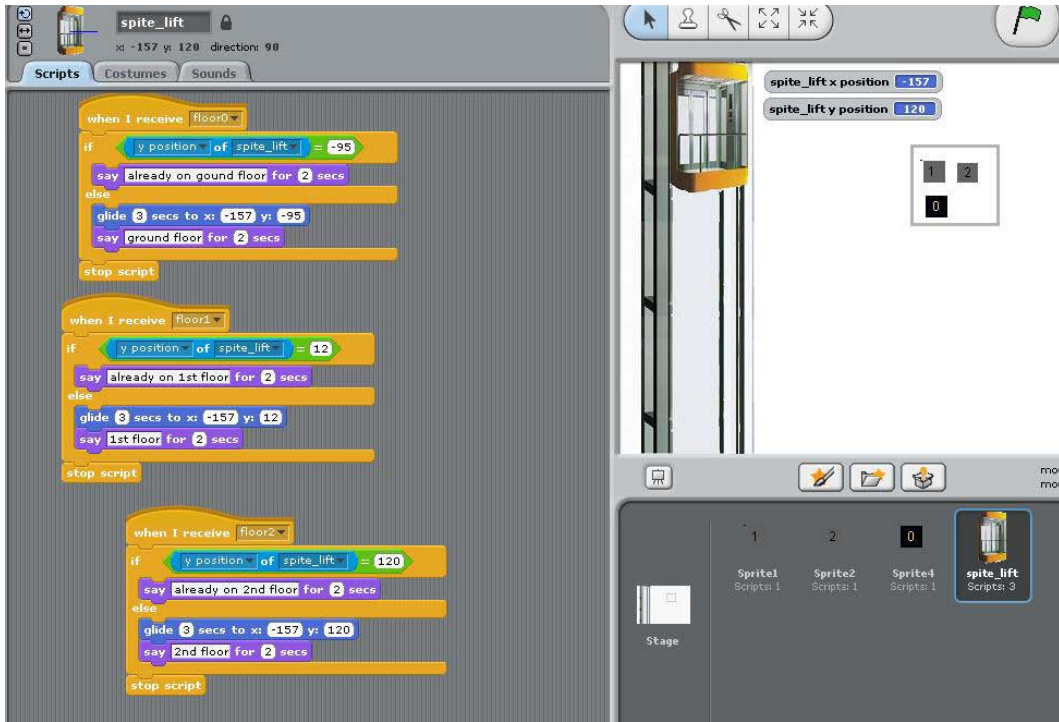


Figure 8. The solution to the first part of the activity



Figure 9. The risk button

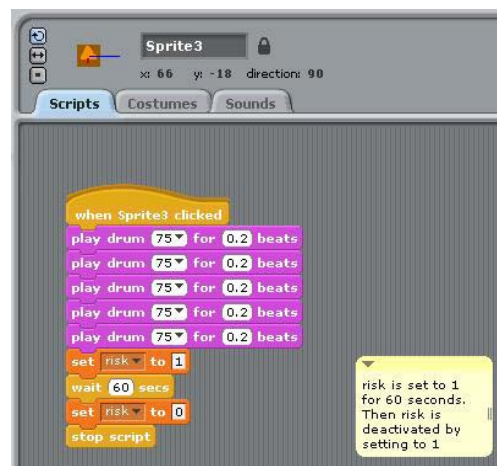


Figure 10. The script for the 'risk button'

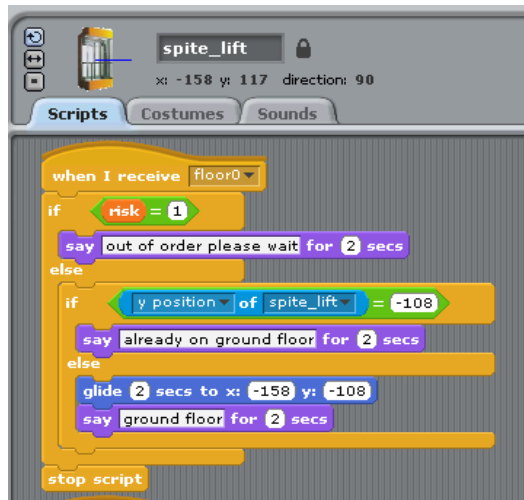


Figure 11. One of the ‘nested if- else constructs’ that students were called to implement

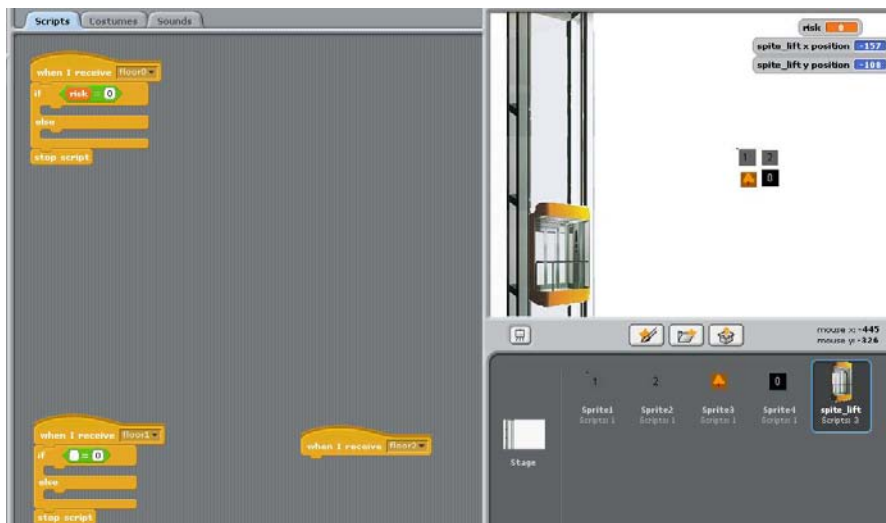


Figure 12. The half-baked approach for the second part of the activity

Findings

This section focuses on what the participants actually did and the way in which the Scratch learning experience supported the development of their ideas. The role that the features of simplicity and visibility, which were embodied in Scratch, played towards participants’ engagement in the programming process is brought into focus. The role of the real-life scenario in the process of engaging in programming concepts is also discussed. The section ends with a discussion of how students’ awareness of the way conditional statements function was emerged and shaped.

Programming in Scratch

The features of visibility and simplicity, (as introduced by DuBulay *et al* (1989)), which are embodied in the programming environment of Scratch, were critical for the students’ engagement in the process of programming. In a way, the absence of concerns about ‘syntactic issues/errors’ and the simplicity underpinning the process of building a script (by snapping together different blocks) encouraged the participants to focus upon the implementation of the programming solution. Although the participants did not verbalise this explicitly, they at no time

expressed that they encountered difficulties in using Scratch. Instead they focused on finding ‘a solution that worked’ and ‘understanding what each block does’. The features of visibility and simplicity can be found lying in Guzdial’s (2003) consideration, in accordance of which, it is of great significance to provide novice programmers with immediate feedback on their work, especially when the work is still a work in progress. Interestingly, Scratch provides this option to users and students unconsciously took advantage of this opportunity. In a way, the whole process of programming in Scratch was based on the feature of the ‘immediate feedback’ which supported students to test their solution and engaged them in a debugging procedure. This allowed them to reflect on the way the conditional statement operated, by checking the outcome when changes were made in the body of the conditional constructs. Discussions that took place with Luke and Kevin are detailed below.

R: I noticed that while you were working you realised that the script did not work and you said to me ‘Oh, this does not work’. How were you sure that your script did not work?

L: Cos basically, I played it. No. First I made it and I played it and after I played it I realised that something was wrong with it. I could look back at everything; and I found the problem and I played it again [...]

Interestingly, the discussion exemplifies Guzdial’s (2003, p.19) point according to which when students are working on their script, they ‘don’t want or need to deal with subtle shades of correctness- they want it to be right or wrong, so that they can correct it and move on’.

In a similar way, the following episode with Kevin exemplifies the fact that the feature of the immediate feedback, as well as the simplicity underpinning the process of changing the script, played a significant role in the programming process and met his needs for solving the problem immediately.

K: Miss?! Look! Let’s go! [he presses the execution button]

R: ...

K: It’s not doing anything! [disappointment and anger]

R: Don’t worry. Take your time...

K: No, no! [he is looking his script again]

R: ...

K: Uhhh... that’s why... I didn’t add this; here is y-position not x. [he makes the needed corrections and moves on] Now, it must work!

The real life scenario

This section focuses on how the use of the real life scenario resulted in supporting student’s understanding of programming concepts. First, it was observed that it was easier for the students to simulate the functionality of the lift due to the fact that they could establish connections with their experience of using it. Second, it was perceived that students had drawn upon the real- life scenario of the activity, developed it and even extended it. Interestingly, the extended scenario required the exploitation of the conditional statements and nested conditional constructs as well as other programming commands at a more advanced level. Both points are discussed further below.

As far as the first point is concerned, it is worth mentioning that almost all the students altered the messages that were supposed to be displayed on screen or to be heard. Initially this was considered to be the result of the playful nature of the learning experience. However, it then became clearer that in fact students’ intention was to change the messages and the sounds, so as these corresponded to their real-life experiences. For instance, it was observed that Billy

changed the message that was displayed on the screen when the risk button was pressed. When he was asked to explain his script, interestingly he mentioned:

'I just did it like the normal lift. If there is risk the message 'lift is temporarily out of order please wait for a member staff' sounds. And the lift is not moving. Like the normal lift'.

The real- life scenario possibly encouraged the establishment of connections with their real- life experiences; this might allow students to engage more easily with the activity (without paying attention to the written instructions) making also clearer the rationale for using the conditional statements.

In relation to the second point, the students were seen to enter Resnick's creative thinking spiral for a second time through imagining a new idea which extended the given scenario of the initially given activity. Interestingly, through the implementation of the extended scenarios students' became involved deeper into the programming concepts. In fact, the extended scenario usually moved them to exploit the programming concepts introduced at a more advanced level or to explore new programming concepts altogether.

Timothy was the first student to explore and implement his own ideas. His idea was to extend the real-life scenario by adding a person on the lift. He focused on achieving synchronisation among the three objects (the lift, the man and the button pressed). Interestingly, the same tendency was perceived in the case of the other three students, who in the time left attempted to either to add more buttons to interface with the lift or to add a man in the scenario (*see figure 13*). The scenario was extended further using this concept: 'The man drives the car, gets out of the car and uses the lift'. To understand and use this concept, students used simple and nested conditional statements, new blocks from the 'palette' ('hide', 'show', 'rotate' commands) and the 'broadcast mechanism' in order to synchronise the different objects.

R: What are you trying to do Luke?

L: I'm just trying to make the lift shake a bit when there is risk.

R: How will you do this?

L: I don't know actually. Maybe I'll use these if-elses. Combined together. [He means nested 'if- else' constructs]. Miss how can I make it move slightly right and left quickly?

R: What about using the 'move' block [...] ?

However, the contribution of the programming environment of Scratch should not be ignored. The students' engagement in the process of extending the real- life scenario is closely associated with the representational pluralism that the programming environment of Scratch supports. The idea of extending the real- life scenario would be impractical if the programming environment of Scratch did not provide users with the necessary tools (i.e. designing area ,range of sprites and blocks, area for recording messages) in order to breathe life into their ideas.

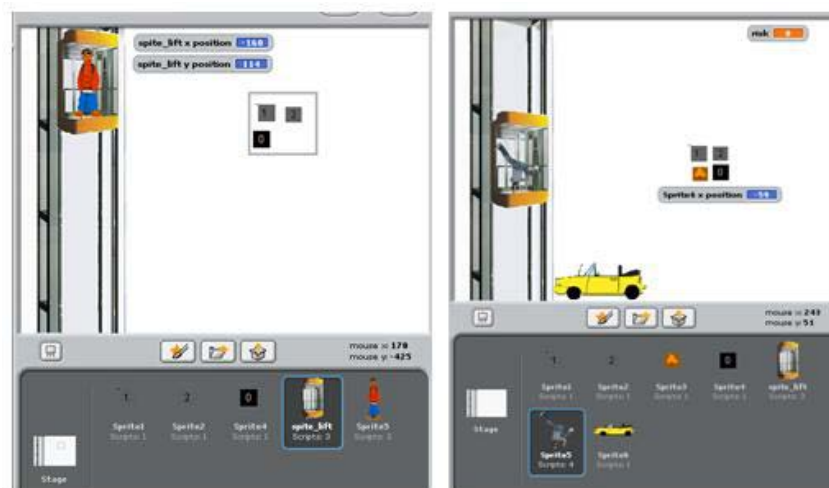


Figure 13. The extended scenarios

Students' perceptions

During the first stage of the study, it was critical to examine whether or not the students had come across a statement like the one given to them (see figure 14) because this would imply that they could be familiar with the concept of conditional statements. Interestingly, none of the four students were able to explain the given conditional statement of the questionnaire. Examining the ability of students to explain the given conditional construct after their engagement in the study was considered to be particularly interesting, as it would indicate their level of understanding.

It was observed that the students' explanations could mainly occur with references to Scratch. Students used the terminology used in Scratch or tried to correlate the given conditional statement (see figure 14) to the ones they had constructed previously in the framework of the activity. Presumably, this fact was expected, taking into consideration that this experience was the students' first time to engage in programming concepts. For instance, Luke was a typical case. He explained the conditional statement (see figure 14) by making use of the terminology embodied in the blocks/commands in Scratch (x-axis, x-position, y-position etc) and parallelizing it to a conditional construct which he had implemented during the activity.

```

If (x=0)
  walk();
else
  run()
```

Figure 14. The given statement

R: [...]Can you now explain this statement [see figure 14]?

L: Yeah! When on x-axis is x-position equals to zero [...] Look here Miss! [he shows to me a conditional statement that he had previously implemented on Scratch]. It is the same thing. Here [he is referred to a condition in his script where x equals to 39] it will not be 39, it will be zero. And it will walk. Otherwise the thing will run.

However, some other explanations were 'less associated' with the programming experience in Scratch. This is not to say that these explanations were completely disassociated with the

activity; but rather to bring into focus the fact that students were seen to develop a more comprehensive understanding of the programming concept of conditional statements and to identify programming scripts behind everyday technologies. Previous experience seems to set a basis whereupon further connections with the wider application of the programming concept of conditional statements can be established. The explanations which were 'less associated' with Scratch were seen as the result of a fruitful interpretation of the new knowledge based on the existing experience. In a way, this process brings into focus the ideas that underpin the Piagetian constructivism, which argues that previous experience and knowledge in general affects the ways in which the construction of new knowledge occurs (Hewson, 1992).

Billys' explanation falls into this category. The fact that he had first come across a conditional statement like the given one on a webpage, which did not load normally, can be seen as a significant factor that activated the process of the de- contextualisation.

R: Initially you stated that you don't know what this statement does.

B: Yes. But I've seen this on a webpage. It didn't load properly.

R: Yes. I remember that you had mentioned this. Can you now explain what the statement does?

B: Basically, it is like here with the lift. If I press '1' the lift goes there [he shows his script]. It is pretty similar to that. If I press something and there is a problem, the page doesn't load properly and probably after refreshing it [noise] there is no problem and it loads properly. It is all programming and on the webpage

R: Interesting. So can you explain to me what this statement does?

B: Basically is a script. And if 'x=0' it will walk. If it is not, the object will run.

R: What will be the value of 'x' in order for the object to run?

B: Basically, it has to be greater than zero or lower than zero but it can't be zero coz the object then won't run.

However, whether closely associated with the experience in Scratch or 'less associated', all the students' explanations illustrated a growth in awareness about the programming concept of conditional statements. The growth in awareness was identified through a change from being unaware of the way the statement functioned to becoming more aware. The achievement of awareness was not a straightforward process for all the students. In whichever way, small or big there was some degree of awareness which was achieved after considering and reconsidering possible approaches, experimenting with multiple solutions and passing the different stages of Resnick's creative spiral.

Conclusion

This study presented how the use of Scratch in a scenario drawn upon real- life, supported the four participants' in developing their understanding of conditional statements. Although, it was a small- scale study and the findings raise new questions for exploration, it seemed that the programming environment of Scratch, as well as the real- life scenario of the activity, was capable of encouraging students' to understand conditional statements. However, there are additional elements that could have contributed to this; the way instructions were provided, the sense of the creation or the feeling of 'developing as a programmer', another designing of familiarization activities and the interrelations between these elements could be the basis whereupon further research could be conducted.

In a way this study could be seen as a small step towards the direction of encouraging students to experiment with knowledge in context, allowing them to make connections with the real world and enabling them to cope with problems creatively and playfully (Resnick, 2007a; Galarneau 2005; Nicaise *et al*, 2000; Siemens 2004).

From a student viewpoint, a learning experience -drawn upon the Kindergarten approach and the thick view of authenticity in Scratch- creates two significant opportunities. Firstly, it allows children to experiment with programming concepts and shape the idea of computational programming. In the framework of this process is likely general thinking skills as well as a general interest for the area of programming and computer science to be developed. Last, it allows students to implement their ideas. Through this process there is the potential for students to see themselves as producers and not merely as consumers of technological 'products'; however, the nature of such production is worth arousing one's interest. It is the type of producer students become that is central to the conception of the Creative Society.

References

- Alexopoulou, E. and Kynigos, C. (2008), 'Half- baked games as a context for understanding conditional constructs' (in greek). Available at: http://www.etpe.gr/files/proceedings/21/1223368001_DIDINFO8_71_80.pdf
- Doukakis D., Tsaganou G., Grigoriadou M.(2007), 'Using animated interactive analogies in teaching basic programming concepts and structures'. *Proceedings of the ACM Conference on the State of: Informatics Education Europe II*, Thessaloniki, Greece, 257-265.
- DuBoulay, B. (1989), 'Some difficulties of learning to program', In: E. Soloway and J. C. Spohrer (eds), *Studying the Novice Programmer*, Hillsdale, NJ, Lawrence Erlbaum Associates , (pp.283-299).
- Galarneau, L. (2005), 'Authentic Learning Experiences Through Play: Games, Simulations and the Construction of Knowledge'. Available at: <http://www.digra.org/dl/db/06276.47486.pdf>
- Guzdial, M. (2003), 'Programming Environments for Novices'. Available at <http://coweb.cc.gatech.edu/mediaComp-plan/uploads/37/novice-envs2.pdf>
- Hewson, P. W. (1992), 'Conceptual change in science teaching and teacher education'. National Center for Educational Research, Documentation, and Assessment, Madrid, Spain.
- Kahn, K. (2004), 'ToonTalk- Steps Towards Ideal Computer- Based Learning Enviroments'. In: Mario Tokoro and Luc Steels (eds), *A learning Zone of One's Own: Sharing Representations and Flow in Collaborative Learning Enviroments*, los Pr Inc.
- Lee, O. and Lehrer, R. (1987), 'Conjectures concerning the origins of misconceptions in LOGO'. *The annual meeting of the American educational research association (AERA 1987)*, Washington, DC.
- Maloney, J., Peppler, K., Kafai, Y., Resnick, M., and Rusk, N. (2008), '*Programming by Choice: Urban Youth Learning Programming with Scratch*'. Available at: <http://web.media.mit.edu/~mres/papers/siqcse-08.pdf>
- Mims, C. (2003), 'Authentic learning: A practical introduction and guide for implementation'. *The Meridian Journal*, 6(1).

- Nicaise, M., Gibney, T. and Crane M. (2000), 'Toward an Understanding of Authentic Learning: Student Perceptions of an Authentic Classroom'. *Journal of Science Education and Technology*, 9(1), 79-94.
- Papert, S.(1993). *Mindstorms: Children,Computers, and Powerful Ideas*. London: Basic Books, 2nd Edition.
- Pea, R. D. (1986). 'Language-independent conceptual bugs in novice programming'. *Journal of Educational Computing Research*, 2(1), 25-36.
- Putnam, R. T., Sleeman, D., Baxter, J., Kupsa, L. (1989), 'A summary of the misconceptions of high school BASIC programmers', In: E. Soloway and J. C. Spohrer (eds), *Studying the Novice Programmer*, Hillsdale, NJ, Lawrence Erlbaum Associates, (pp. 301-314).
- Resnick, M. (2008), 'Falling in love with Seymour's ideas'. Available at: <http://ilk.media.mit.edu/papers/AERA-seymour-final.pdf>
- Resnick, M. (2007a), 'Sowing the Seeds for a More Creative Society'. *International Society for Technology in Education*, 18-22.
- Resnick, M. (2007b), 'All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten'. *Proceedings of the 2007 Conference on Creativity and Cognition*, Washington DC, USA (pp. 1-6).
- Resnick, M., and Silverman, B. (2005), 'Some Reflections on Designing Construction Kits for Kids'. *Proceedings of Interaction Design and Children conference*. Available at: <http://ilk.media.mit.edu/papers/IDC-2005.pdf>
- Soloway, E. and Spohrer, J. (eds) (1989), *Studying the Novice Programmer*. Hillsdale, NJ, Lawrence Erlbaum Associates.
- Shaffer, D. W. and Resnick, M. (1999), "'Thick" authenticity: New media and authentic learning'. *Journal of Interactive Learning Research*, 10(2), 195-215.
- Siemens, G. (2004), 'Connectivism: A learning theory for digital age', available at <http://www.elearnspace.org/Articles/connectivism.htm>
- Tzimogiannis, A. (2005), A pedagogical context for teaching programming in Secondary Schools (in greek). *Proceedings of the 3rd Panhellenic Conference in Didactics of Informatics*, Korinthos.