International Journal of Gender, Science and Technology
in association with Women in Games

Advancing Elementary-School Girls' Programming through Game Design

Ahmet Baytak

Harran University, Turkey

Susan M Land

The Pennsylvania State University, U.S.

ABSTRACT

Rapid technological changes and developments have been seen in schools and workplaces throughout the last decade. However, girls' representation in technology studies has not grown in accordance. Some scholars have suggested that computer gaming could be an entry point for girls to engage with technology, mathematics, and science (Van Eck, 2006). This case study describes how we introduced a class of 5th-grade girls to programming via a computer-game design approach. Although the case study is exploratory, implications are drawn for the potential role of game design as a vehicle to involve more girls in computer science.

KEYWORDS

Children; gaming; environmental education; programming; Scratch software; educational technology.

the UKRC advancing gender equality in science, engineering and technology

The Open University

This journal uses Open Journal Systems 2.2.2.0, which is open source journal management and publishing software developed, supported, and freely distributed by the Public Knowledge Project under the GNU General Public License.

Advancing Elementary-School Girls' Programming through Game Design

INTRODUCTION

Although technology is rapidly developing, there has been a significant drop in computer science majors since the 1960s (Gupta, 1996), and because of this drop, researchers have proposed solutions to increase interest in computer science majors. Yet, few practical, applicable educational strategies or approaches have been identified that increase student interest, especially elementary school and middle school students', toward higher education or careers in computer science (Moore, 2008).

Plass et al (2007) noted that girls and women, in particular, exhibit less interest in computer science majors and programming. In her study, Pinkard (2007) found that girls who elected computer science majors had less prior knowledge than boys electing the same majors. More broadly, it has been found that students who enter computer science majors have gaps in their background knowledge and because of these gaps, there is less student engagement and involvement in computer science activities.

Repenning and Iannidou (2008) claimed that current technology education models do not work effectively in U.S. schools. These researchers raised an important aspect of the problem, which is that current elective technology education courses and computer clubs attract and sustain students who are already strongly interested in computer science. In order to increase students' interest toward computer science majors, Repenning and Iannidou suggest a scalable game-design approach, which can be used as a means to broaden participation in computer science and to advance design understanding. However, Repenning and Ioannidou also note that, usually among girls, computer science and game design are seen as simply learning how to code with complicated combinations of letters and symbols, which decreases their appeal. Game design could serve as a mechanism for exploring computer science in a genre that has appeal to girls as well as boys. Kelleher and Pausch (2006) found an increase in girls' interest in learning to program when they were asked to make visual stories with that software. Others have suggested that computer gaming could be an entry point for girls to engage technology, mathematics, and science (Van Eck, 2006).

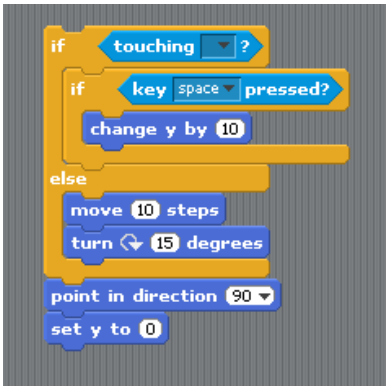
Learning-by-game design is grounded in an established educational pedagogy of constructionism (Papert, 1995) that emphasizes constructing artifacts by programming computers or digital artifacts such as computer games (Kafai, 1996). Kafai claimed that designing artifacts by programming software helps students reformulate their understanding and express their personal ideas and feelings about not only the subject, but also the artifact. Prior studies have demonstrated gains in both programming knowledge and subject-matter knowledge through learning via constructionist pedagogy (Papert, 1995). Papert views programming or game making as a construction tool for personal expression and knowledge construction, and this helps students explore psychological and cultural aspects of learning.

The Game Design Project

This case study describes how we introduced six 5th-grade girls (and 4 boys), age 10-11 years, to programming via a computer-game design approach. The context was a 5th-grade science classroom in an elementary-level charter school[1] in the United States. School systems in the United States are primarily divided into three levels: primary or elementary (kindergarten through 5th grade, ages 5-11), middle (6th-8th grade), and secondary or high school (9th-12th grade). This report focuses primarily on data about the girls, although the entire class of boys and girls participated. The children were asked to design educational games for younger students (2nd graders, age 7-8 years) to learn about environmental science. Although the class was given an overall topic to explore (environmental problems identified from their science textbook), students were given complete ownership over the design of their games. Instead of asking them to design what we wanted, we asked the students design games that they wanted to create about environmental science based on the target audience and their design goals. Thus, the girls were placed into roles of authentic game designers, creating software for a specified audience and purpose.

Students used *Scratch* software to create their games. This program was developed by MIT researchers as a way to make programming accessible to young children and is based on the LOGO concept (Maloney et al. 2008; Peppler & Kafai, 2005). *Scratch* was selected for use with children because it entails a user-friendly interface with visual blocks of programming code, reducing programming errors. As

shown in Figure 1, users drag and snap blocks together to build scripts. It also contains a library of sample designs that could be modified if students needed that level of support to get started. Figure 1 shows sample command blocks and their programming.



Programming with *Scratch*

```
if (!$this->rawFieldContent) {
                $result = "";
        }       elseif ($this->syndicateHtml) {
                $result = "<![CDATA[".$this->rawFieldContent."]]>";
        } else {
                if ($this->truncSize and is_int($this->truncSize)) {
                        $result = FeedCreator::iTrunc(htmlspecialchars($this-
                        >rawFieldContent),$this->truncSize);
                } else {
                        $result = htmlspecialchars($this->rawFieldContent);
                }
        }
        return $result;
```

Programming with PHP

Figure 1. Comparing the complexity of syntax in PHP and Scratch programming languages.

This report presents a summary of a pilot project that explored whether 5th-grade

the girls, except one, was able to set a script for a simple action with the command blocks provided. Although the school offered technology clubs in its after school program, only two of these six girls attended the clubs.

We examined in depth what computational or programming concepts these elementary students used as they worked with *Scratch* to program their games. In general terms, we found that, despite students' minimal prior experience with *Scratch* and almost no design experience, all of the students were able to program a functional game after the 10th design session (Figure 2 shows a girl's game in raw format). To analyze programming concepts[3]used, the students' games were analyzed based on the work of Maloney et al. (2008) and Malan and Leitner (2007) who have classified each *Scratch* command as different programming concepts. During the analysis process, each of the students' games was opened with *Scratch* software, played, and examined for the codes on a daily basis.

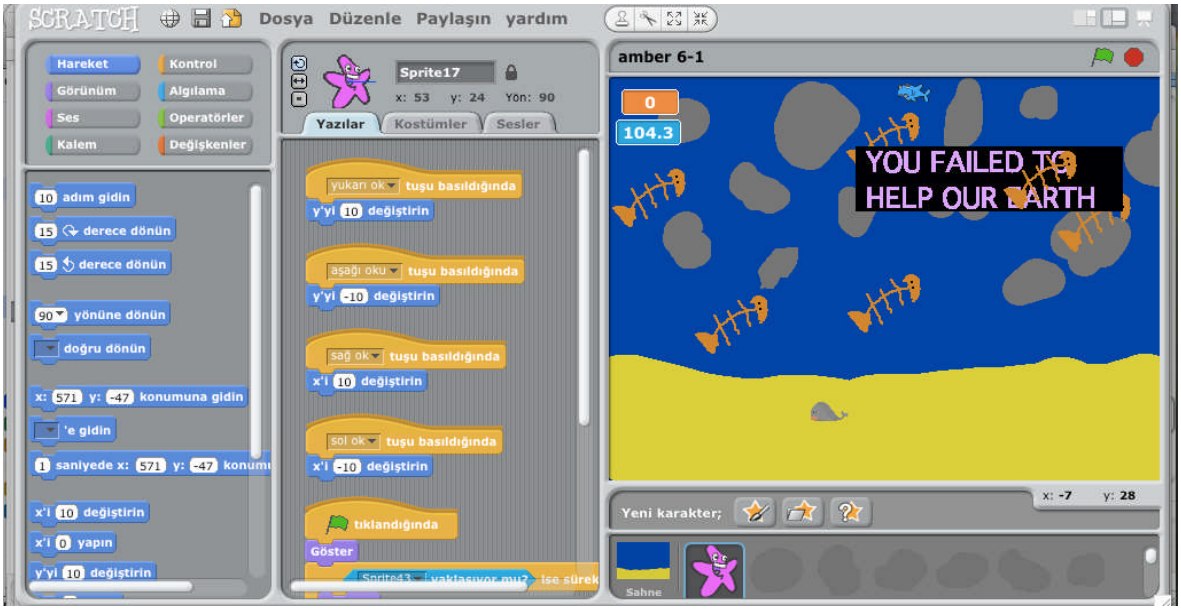


Figure 2. A screenshot from one girl's game; the left panel has programming scripts, right top panel has the actual game, and right bottom has the sprites used in the game.

Frequency counts of programming concepts were derived for each session's game by counting commands representing each programming concept. Based on the analysis of the games, girls used more programming concepts than boys. For

example, girls used 26.36 sprites[4] and 92.4 statements for these sprites in their games where class average was 23.56 sprites and 78.4 statements (Table 1). Since these girls used a higher number of sprites, the number of scripts that they used for each game was high. Compared with other students in the design project, girls used more statements, Boolean expressions, conditions, and loops in their games (Table 1). However, on average, there was a smaller number of variables used in girls' games.

Table 1. Students' average use of different Scratch programming concepts.

	Characters	Statements	Boolean Expression	Conditions	Loops	Variables	Threads	Events
Male Average	20.75	64.4	16.5	14.5	12.9	4.67	36.7	0.04
Female Average	26.36	92.4	24.7	19.1	21.1	3.58	49.1	7.72
Class average	23.56	78.4	20.6	16.8	17	4.13	42.9	3.88

As Maloney et al. (2008) indicated, students could use fewer commands to have the same function in their games. Nevertheless, the increased use of complex commands of *Scratch* such as variables and events show that the girls in this project were able to use advanced programming concepts with *Scratch*. It was interesting that boys rarely used the broadcasting feature, which is an advanced feature of *Scratch*. Yet, most of the girls, after getting help from the teacher, included many broadcasting commands in their games. With this broadcasting function, the students were able to make new game levels.

Thus, comparing the girls programming skills at the beginning of the design project and at the end of the project, an obvious improvement was observed in their skills. During the pre-interview, before starting the game design project, students knew the functions of command blocks in *Scratch* but they were not able to set an algorithmic program with these command blocks to run a simple script. However, as seen in Table 1, the girls used a high number of command blocks and programming concepts in their games. It should be noted that using high numbers of programming concepts is also an indication of a good algorithmic program,

otherwise the games might not work properly. In addition, all the girls' games functioned properly in terms of their specifications to teach second graders about environmental issues.

Gaming Elements

Despite the short time period of the project, all the students were able to successfully design playable games and incorporate standard gaming characteristics and elements. From the size ratio of the characters to color matching of each sprite, the students designed a realistic computer game that others could play. In addition, they were able to consider accessibility of their games. Besides providing detailed instructions at the beginning of the game, they also decided to use commonly-used arrow keys for the control options and variety of feedback types for reinforcing game players. There were different scoring and timers used to show competency in the games and graphic and sound feedback to show players' success and failure. The children drew their own graphics and some composed sound for the games. Overall, the students showed evidence of organization and creativity throughout the game design process. We also conclude, based on the interviews, that the types of commercial computer games students played at home may have influenced the design choices they made in the games they made.

In sum, the girls in this project did not have any game design experience prior to this opportunity, yet they were able to design games that incorporated important gaming elements and characteristics. Overall, the use of different game characteristics varied based on individual games, goals, and level of experience. Using these characteristics made the students' games more authentic and professional.

A Classroom Culture of Game Designers

The game design project set out to explore the potential of this medium to engage girls in the culture of programming. Achieving this goal depends in part on creating a culture where girls can engage with each other in ways that bridge their identities as girls and as practicing programmers. Consequently, we looked for instances of how girls engaged in dialog and social practices that were consistent with the culture of design and programming.

The first way that girls engaged in social practices related to programming was through informal knowledge sharing. During the design process, the girls typically sat close together and were curious to look at each other's games. By looking at games informally, they shared knowledge and strategies that subsequently spread through the group. The script code for scoring, for example, was a function that most of the students learned from each other. These unplanned interactions took place when a girl would show her game to the person next to her, saying "see what I have". One student showed another how she designed clouds for her game background, and one of the comments that she received was "you should change this to white" (referring to the color of an image in her game). Hence, the girls engaged in knowledge sharing and informal design review, a practice important to any community of programmers.

Girls also engaged in social practices by peer-testing each others' games, and asking each other for help, often a result of either being 'stuck' or identifying a function from another's game that they wanted to incorporate in their own. For example, one girl asked another how to use the *Scratch* image editor to create her own game characters, after seeing that her peer had created her own images in her game. It became evident that involvement in the game design increased students' willingness to communicate when it came to doing actual tasks without waiting for teacher's encouragement. The students were often able to resolve each other's design problems together even though they were assigned to work on design tasks individually. Since it was perceived that there was not a single expert in game design in the class, the students negotiated different aspects of game design. Thus, by sharing their games with classmates (and sometimes family members), girls engaged in social conversation about programming and gaming.

CONCLUSION

This case study provides some exploratory support for the potential of using game-design as a vehicle to support elementary-school girls to engage in and learn programming. The success of the young students in this small time frame is largely due to use of software with a visual interface that is matched to the novice programmer's level of expertise (Lin et al, 2005). Similar to Pinkard's (2007) findings, this report showed that most of the girls started the game design project with low prior experience, but the number and type of the programming concepts

used in their games increased, demonstrating an ability to conceptualize these concepts and build their scripts. These results may also have implications to influence the cycle of elective technology education courses and computer clubs that attract and sustain male students who are already strongly interested in computer science (Repenning & Iannidou, 2008). In addition, girls' involvement in gaming and programming prompted social interaction and community about computing, which is rarely seen among female students at that age level. Designing games that reflected ownership of ideas, sharing those with others, and playing others' games served to create a culture that supported conversations and practices around computing and gaming.

ENDNOTES

[1] Charter schools are primary or secondary schools in the U.S. that receive public money from the state government. But, they are not subject to all of the rules, regulations, and statutes of other public schools. Instead, they are accountable to produce results that are set forth in the school's charter. Charter schools are attended by choice and are an alternative to public schools.

[2] Wikipedia defines object-oriented programming as "a programming paradigm using 'objects' – data structures consisting of data fields and methods together with their interactions – to design applications and computer programs."

[3] Programming concepts refers to computer programming ideas such as program structure, variable declaration, conditional and looping constructs. A programming language consists of loops (repeating some function), statements (actions), characters (objects and subjects), Boolean expression (and, or) and conditions (if).

[4] A sprite is a two-dimensional image or animation that is integrated into the game environment.

REFERENCES

Gupta G.K. (1996). *Teaching Computer Science as the Science of Information*, TR11, Dept of Computer Science, James Cook University.

Kafai, Y. B. (1996) 'Gender differences in children's constructions of video games' in P.M. Greenfield & R. R. Cocking (eds.), *Interacting with Video* (pp. 39–66). Norwood, NJ: Ablex.

Kelleher, C. & Pausch, R. (2006) 'Lessons learned from designing a programming system to support middle school girls creating animated stories' *IEEE Symposium on Visual Languages and Human-Centric Computing,* Brighton, UK, September 4th – 8th. (pp.165-172).

Lin, J., Yen, L., Yang, M., & Chen., L. (2005). 'Teaching Computer Programming in Elementary Schools: A Pilot Study' Presented at NECC, Taiwan, Retrieved March 25, 2009, from http://www.stagecast.com/pdf/research/Lin_NECC2005_Paper_RP.pdf

Malan D. J. & Leitner, H. H. (2007) *Scratch* for budding computer scientists, *SIGCSE*, 2007, Covington, Kentucky, March 7th-11th

Maloney, J., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. *SIGCSE 2008* Portland, Oregon, USA, March 12th – 15th. Retrieved March 20, 2009 from http://info.scratch.mit.edu/Research

Moore, D. (2008, April 11). National Science Foundation offers $2.7 million to foster kids' interest in computer science. *News Center (Dallas, TX).* Retrieved April 10, 2009 from http://www.utdallas.edu/news/2008/04/11-001.php

Papert, S. (1995). *The Children's Machine: Rethinking Schools in the Age of the Computer*. New York: Basic Books.

Peppler, K. A. & Kafai, Y. B. (2005). *Creative Coding: The Role of Art and Programming in the K-12 educational context*. Retrieved March 15, 2009 from http://scratch.mit.edu/files/CreativeCoding.pdf

Pinkard, N. (2007). *Girl Power, Encouraging Sixth Grade Girls to Give Video Games a Try.* Retrieved April 2, 2009 from

http://spotlight.macfound.org/main/entry/pinkard_girl_power_encouraging_girls_games/

Plass, J. L., Goldman R., Flanagan, M., Diamond, P., Dong, C., Looui, S., Rosalia, C., Song, H., & Perlin, K. (2007). RAPUNSEL: How a computer game design based on educational theory can improve girls' self-efficacy and self-esteem. *The 87th Annual Meeting of the American Educational Research Association*, Chicago, IL: April 12th. Retrieved March 10, 2009 from http://create.alt.ed.nyu.edu/courses/2176/reading/AERA_07_Rapunsel_Plass_etal.pdf

Repenning, A, & Ioannidou, A. (2008). 'Broadening participation through scalable game design', *SIGCSE Bulletin, (40)*1, 305-309.

Van Eck, R. (2006). 'Using games to promote girls' positive attitudes toward technology', *Innovate: Journal of Online Education*, *2*(3). Retrieved August 11, 2010, from http://www.innovateonline.info/pdf/vol2_issue3/Using_Games_to_Promote_Girls'_Positive_Attitudes_Toward_Technology.pdf