Creating an Adventure Game with an Embedded Novice Programming Environment for Learning LOGO

Ioannis Sarlis[1], Dimitrios Kotsifakos[2], Dr. Panagiotis Zaharias[3]

[1]MSc, Med, IT Teacher, ioannis@sarlis.me
[2] MSc, PhD (candidate), Electronic Engineering, VET, University of Piraeus, dimkots@sch.gr
[3]Adjunct Faculty Member, Open University of Cyprus, pz@aueb.gr

Abstract

This article presents the construction of a game that includes a Novice Programming Environment for high school students and is to be used for teaching LOGO programming language. This programming environment is a game that is designed, implemented and tested specifically for junior high-school students for them to acquire basic programming skills. This game is called "Super Turtle Adventures" and is based on Digital Game-Based Learning (DGBL). It helps students to learn LOGO Programming Language, by creating various challenges and puzzles, which they must solve using LOGO code. Students pass from one level to the next by solving these puzzles. The benefit for students playing this game is, on the one hand, the improvement of their programming skills on the other that they learn to respect the environment through recycling. The article treats programming as a new digital skill that needs to be mastered by high school students. The article concludes that programming should be taught using DGBL in schools and suggests "Super Turtle Adventures" game as a Digital Game-Based Learning application for teaching LOGO in schools.

Keywords: LOGO Programming Language, Teaching Programming, Novice Programming Environment (NPE), Digital Game-Based Learning (DGBL)

1. Digital Game-Based Learning (DGBL)

Digital Game-Based Learning (DGBL) is a teaching approach that involves the use of digital games into teaching, in order to learn, explore and/or practice some form of training material. Learning through a digital game environment integrates learning principles while enabling students to act within the game environment. Drawing from the constructivist theory of education, digital game-based learning (DGBL) connects educational content with a computer or video games and can be used in almost all subjects and skill levels (Kotsifakos, 2018). Proponents of digital game-based learning contend that it provides learning opportunities that engage students in interactive instruction and helps prepare them to participate in the globalized, technological society of the 21st century (Coffey, 2009).

ISBN: 978-960-578-058-6

Educational games were created primarily for teaching purposes and so researchers are now taking advantage of the educational benefits these games procure. As noted by Dede (2018), Sarlis (2018), and many other researchers, Digital Game-Based Learning DGBL is essentially a CAI (Computer Aided Instruction) and for that reason, DGBL can provide an environment for students to practice and acquire the necessary skills required by the school curriculum. Recent studies have shown that there is significant interest in Digital Game-Based Learning (Prensky, 2003). These games can be used as tools together with their school lessons so that the students are motivated in learning.

Gee (2007) points out that students understand better the things they are going to do when an idea is visualized. For that reason, an educational digital game forms a rich multimedia environment in which children can think, understand, and easily perform things. Therefore, digital games can be used as an alternative way of teaching. Some of the learning principles are directly related to the digital game, and these are:

- Interaction
- Creativity
- Experiment

According to Gee (2009), digital games provide a continuous learning environment through the attractiveness and the entertainment they provide to users. The purpose of an educational game, in comparison with all digital games, is not only to achieve the goals of the game but also a way to acquire knowledge and skills based on the game and on situations that focus on the learning objects of a lesson. An educational game can provide students with a framework to achieve the educational goals that have been originally set by the teacher. Gaming applications offer on the one hand a pleasant and entertaining environment for students and on the other hand an indication and an incentive for the student to achieve learning goals. Also, these applications uses a familiar process that students often experience when they play digital games generally; e.g. placing a “gold star” on a level is a confirmation that the player has completed a level, or that he/she is on the right path, giving the player positive feedback, thus increasing the self-esteem of the player.

Children, like all people, love to learn at any time when they are not forced to do so (Wabishabi, 2019). Games are a way of attracting students to learning in a pleasant way; modern computers and video games provide young people with such learning opportunities in a very short time, often through a touch that only takes seconds (Prensky, 2007). Gee argues that “the real significance of good computer games and video games is that they allow people recreation in new worlds and thus achieve entertainment and deep learning at the same time” (Gee, 2007). Some teachers, in addition, consider that Game-Based Learning is a strong educational approach (von Wangenheim & Shull, 2009). Educational games move students to the center of the

ISBN: 978-960-578-058-6

learning process, which makes this kind of learning easier, interesting and more effective. With the aim of Information and Communication Technologies (ICT) in Education, computers turn into learning multi-tools; teaching and learning processes become pleasant, interesting, entertaining and comprehensible. Digital Game-Based Learning (DGBL), is a learning approach that involves the use of computers and digital games for exploring and practicing educational materials. The creation of an educational game is not only for entertainment purposes; the main purpose of this creation is the acquisition of knowledge and skills for situations that are not primarily related to the game itself, but which focus on the learning object of the course.

2. LOGO Programming Language

LOGO Programming language is an educational programming language, designed in 1967 by Wally Feurzeig, Seymour Papert and Cynthia Solomon (InfoSys Foundation, 2017). LOGO is not an acronym but comes from the Greek word “Logos” which means “word” or “logical thinking”. The name “LOGO” was given by Artificial Intelligence researcher W. Feurzeig (Goldenberg, 1982), in order to separate it from other programming languages that were mainly numerical-oriented and had nothing to do with graphics or logical thinking. According to Papert, (1993), “We learn better by doing ... but we learn even better if we combine action with speech and reflection on what we do”. LOGO is a programming language that is specifically designed to be used by young children (students of primary and secondary education) to learn code. Using LOGO, students can communicate with the computer and then program it very easily and quickly, even if they are new to programming. LOGO uses functional programming techniques. For example, the user's commands are executed by calling special procedures, the so-called functions.

MicroWorlds Pro is a multimedia environment for programming; in fact, it is a software implemented to create programs and multimedia applications that are based on LOGO. It belongs in the educational software category which is suitable for composite projects development. MicroWorlds Pro uses the Logo programming language to enable players to program turtles. Turtle is a virtual character executing LOGO commands, depending on how it is programmed. Each turtle has a name, position, direction, pen thickness, pen color, shape and can be instructed to execute certain commands when it is clicked. Turtles can be used to design, “decorate” a page, or create animations.

3. The Features of LOGO Language

All programming languages are tools for modeling. LOGO was designed to create models in a very easy way and is therefore particularly suitable for children. With LOGO, reality can be represented as a model. The term “Modeling” describes the process that begins with the whole and continues with fragmenting the whole into

ISBN: 978-960-578-058-6

smaller entities which are called sub-projects. For example, a LOGO programmer can "teach" a Virtual Turtle how to complete these specific tasks using programming code. (Swan, 1991) through coding. Often, teaching new words to the "turtle" is like writing procedures, ie. sets of instructions for executing small tasks. Learners can interact with LOGO by collecting, processing, analyzing, comparing, representing (symbolically, graphically, virtually) generalizing and interpreting data, while simultaneously adopting a variety of problem-solving strategies

This interaction starts with simple tests and debugging commands and goes on to more complex coding entities through constant self-monitoring and feedback. By becoming familiar with the processes of hypothesis – experimentation - trial and error, learners can grasp the principles of debugging and correcting errors in their code. They develop a high degree of confidence, responsibility, and appreciation for different methodologies and they acquire skills of expression, cooperation, and communication with the computer. Finally, debugging commands and programs help students to reconstruct their original approach to the problem (Glezou & Grigoriadou, 2004). LOGO is a programming language that is often perceived as a philosophy of education, known as discovery learning or constructivism. Although its educational effectiveness is often disputed, LOGO language is considered an ideal tool for "learning by doing" (OpenWorld Learning, 2014). It is undoubtedly an important tool at the teacher's disposal for the development of exploration skills, creativity, problem solving, logic and algorithmic thinking development. LOGO was created to educate children in programming and will always be a point of reference when we talk about coding lessons for the young ages. It is, therefore, perceived that LOGO is a language particularly suited to small ages of students as it entails a role-playing learning process. It is very simple in syntax and in direct contact with the students. As students see the effect of the command on the screen and interact with the turtle, they become the authors of their own operating rules (programming commands), through a pleasant programming environment that does not involve strict rules of code syntax.

4. Novice Programming Environments (NPE)

Novice Programming Environments have been developed to replace the traditional code syntax, using visual commands rather than typing. This approach reduces the cognitive load associated with mandate typing, allowing users to focus on the conceptual solution of a problem. Also, programming environments of this type are easy for users of all ages, cognitive backgrounds, and interests, allowing them to experiment with their various components simply by joining pieces of code together just like LEGO blocks (Resnick et al., 2009). Environments that assume the above features are called NPEs (Novice Programming Environments) (Krul, 2012). NPEs, such as Scratch, Alice, and Lego Mindstorms NXT, have been widely accepted and publicized in recent years, as it has shown that NPEs play an important role in attracting and retaining new developers in school and non-school environments

ISBN: 978-960-578-058-6

(Federici, 2011). NPEs uses visual elements instead of programming commands, concealing the complexity of typing code in a programming language, and makes it easier for novice developers to understand basic algorithmic structures (Roy, Rousse, & DeMeritt, 2012).

NPEs have pleasant interfaces which facilitate software development in a user-friendly programming environment. They are not "threatening" or "hostile" for a novice developer. Students experience no stress or low self-esteem when they meet such a user-friendly programming environment. Indeed, due to the ease of use of NPE, students appear more receptive to further deepening in programming (Olabe, Olabe, Basogain, & Castaño, 2011).

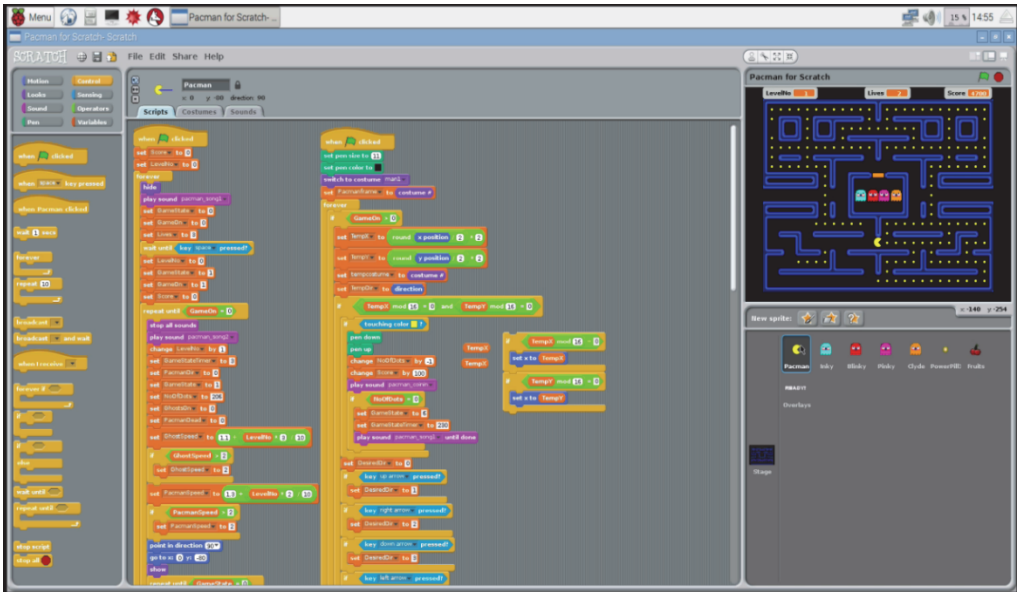

Figure 1. *SCRATCH Novice Programming Environment (source: https://www.raspberrypi.org/magpi/wp-content/uploads/2015/10/Scratch_Pac_Man.png)*

However, it is reminded that the use of an NPE does not solve the problem of coding and syntax of commands, since students will have to deal with it later, usually when they must learn a second “traditional” programming language (Fig. 1).

In most cases, this is postponed until students understand the basic programming principles (Wilson & Moffat, 2010). It must be highlighted that students are actively involved in programming lessons when these are integrated -within a framework of teaching- in topics that they are directly interested in (Gray, Abelson, Wolber, and Friend, 2012). Margulieux et al., (2013), point out that the problem deterring students from participating in programming lessons can be addressed by turning introductory

ISBN: 978-960-578-058-6

programming into an easy and fun experience. There are, in fact, several ways to make this possible; one of these ways is to reduce the cognitive load required of beginners learning to program, with a corresponding reduction in the amount of information used to solve a problem (Robins, Rountree, & Rountree, 2003).

***Figure 2**. First Page of Super Turtle Adventures*
(source: [http://www.TurtleAdventures.info/game)](http://www.TurtleAdventures.info/game)

Super-Turtle Adventures (https://superturtleadventures.info/) is a Novice Programming Environment (NPE) that bridges the gap of visual programming and procedural programming. This game has an embedded programming environment (Editor, Interpreter, Compiler, and Linker) where students can write a program in LOGO using "blocks". Using blocks of commands, students generate code by inserting, changing or removing block commands (Fig. 2, 3, 4). Adding or removing blocks into the programming language editor is like writing or removing code lines of LOGO Programming Language. LOGO Programming language introduces learners to computer programming in a sense that they can type commands, give meaning to the writing of commands, make variables where necessary, create their own functions, etc.

ISBN: 978-960-578-058-6

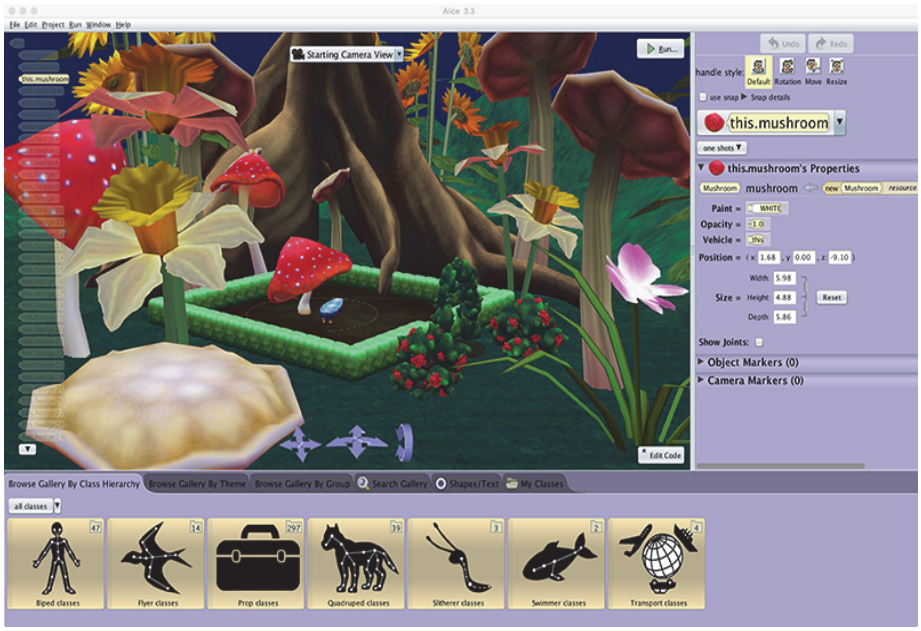

***Figure 3**. ALICE Novice Programming Environment*
(source: https://www.alice.org/wp-content/uploads/2017/05/BuildingAScene_Image.jpg)

5. Conclusions

Computer Programming helps students to develop their analytic and synthetic thinking, enhances their skills in designing and solving problems, and has a positive impact on their creativity and imagination. Novice Programming Environments and coding games have become user-friendly to such high degree, that young children, even Kindergarten pupils, can create their own code. It is pointed out in this article that LOGO is a general-purpose language, but it became more famous for its turtle graphics creations; these creations are made by motion and design commands that are used to produce graphics or paintings on the screen of a computer, or by instructing a small robot called a turtle, to do this. LOGO is a language, which is one of the oldest programming languages, and it was created for educational purposes. LOGO programming language, in relation to Scratch, shows a better correlation of programming with Geometry elements already known to children.

Also, here is a reference to MicroWorlds Pro, which is a multimedia programming environment, a software made to create programs and multimedia applications that are based on LOGO. It is true, that the issue of writing code (coding), cannot be simply solved by replacing the classical programming with “blocks” because pupils or students will, sooner or later, be confronted with the “hostile environment” of an editor, a compiler and a linker to program in a “classical” or “traditional”

ISBN: 978-960-578-058-6

programming environment. Regarding NPE, they have a pleasant interface to enable software development in a novice programming environment. It is not a "threatening" or "hostile" beginner programmer and consequently, by using these user-friendly environments, students do not feel anxious or experience low self-esteem when they meet them.

Figure 4. *Super-Turtle Adventures Novice Programming Environment*

Hence, the creation of an NPE environment for programming is deemed necessary. This article states that, in order to attract students, the research team found it necessary to proceed with the creation of an application which looked like a digital game, but it will be a tool to teach students programming. "Super Turtle Adventures" is an Adventure Game which helps students to learn programming through the integration of an NPE environment. By providing students with all necessary commands, as well as optional loop commands, it helps learners reach the apparent goal of this game, which is simply to recycle various items while the main purpose of this application is for students to learn how to write code in LOGO.

References

Coffey, H. (2009). Digital game-based learning. Chapel Hill, NC.: University of North Carolina at Chapel Hill School of Education.

ISBN: 978-960-578-058-6

Dede, C. (2018, 10). The Potential of Digital Game-Based Learning for Improving Education in the Global South. Retrieved from Digital Learning for Development: http://dl4d.org/wp-content/uploads/2018/10/01-Digital-Game-Based-Learning-Main-Paper.pdf

Federici, S. (2011). A minimal, extensible, drag-and-drop implementation of the C programming language. In Proceedings of the 2011 conference on Information technology education (SIGITE '11) (pp. 191-196). New York: ACM.

Gee, J. P. (2007). What video games have to teach us about learning and literacy (2nd ed.). New York: Palgrave Macmillan.

Gee, J. P. (2009). Deep learning properties of good digital games: How far can they go? In U. Ritterfeld, M. Cody, & P. Vorderer, Serious games: Mechanisms and effects (pp. 67–82). New York, NY: Routledge.

Glezou, K., & Grigoriadou, M. (2004). Playing, probing, and learning, while programming the turtle. In P. Politi (Ed.), Proceedings of the Proceedings of 2nd Conference with international participation "Teaching of Informatics", (pp. 182-192). Volos.

Goldenberg, E. P. (1982, August). Logo - A Cultural Glossary. Byte Magazine, 7(8), p. 210.

Gray, J., Abelson, H., Wolber, D., & Friend, M. (2012). Teaching CS principles with app inventor. Proceedings of the 50th Annual Southeast Regional Conference (ACM-SE '12) (pp. 405-406). NY, USA: ACM.

InfoSys Foundation. (2017, Jun 12). Q & A with Dr. Cynthia Solomon. Retrieved Jan 17, 2018, from Infosys Foundation USA: http://www.infosys.org/infosys-foundation-usa/media/blog/Pages/cynthia-solomon-qna.aspx

Kotsifakos, D., Petrakis, G., Stavrou, M., & Douligeris, C. (2018, September). An Online Game for the Digital Electronics Course for Vocational Education and Training (VET) Students. In International Conference on Interactive Collaborative Learning (pp. 638-649). Springer, Cham.

Krul, K. (2012). Teaching Control Structures Using App Inventor. Master Thesis. Retrieved January 26, 2018, from Utrecht University Repository:

ISBN: 978-960-578-058-6

https://dspace.library.uu.nl/bitstream/handle/1874/254527/ResearchPaper-KevinKrul-25-6.pdf?sequence=1&isAllowed=y

Margulieux, L., Catrambone, R., & Guzdial, M. (2013). Subgoal Labeled Worked Examples Improve K-12 Teacher Performance in Computer Programming Training. Proceedings of the 35th Annual Conference of the Cognitive Science Society (pp. 978-983). Austin, TX: Cognitive Science Society.

Olabe, J. C., Olabe, M. A., Basogain, X., & Castaño, C. (2011). Programming and robotics with Scratch in primary education. In A. Mendez-Vilas (Ed.), Education in a Technological World: Communicating current and Emerging Research and Technological Efforts (pp. 356–363). Badajoz - Spain: Formatex.

OpenWorld Learning. (2014). Open World Learning. Retrieved Jan 17, 2018, from MicroWorlds in Action: http://mia.openworldlearning.org/logo.htm

Papert, S. (1980). Mindstorms: Children, Computers, and Powerful Ideas. New York: Basic Books.

Papert, S. (1993). THE CHILDREN'S MACHINE: Rethinking School In The Age Of The Computer. New York: Basic Books.

Prensky, M. (2003). Digital Game Based Learning: Exploring the Digital Generation. Educational Technology, U.S. Department of Education.

Prensky, M. (2007). Digital Game-Based Learning. MN: St. Paul, Paragon House.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., . . . Kafai, Y. (2009). Scratch: Programming for All. Communications of the ACM, 52(11), pp. Pages 60-67.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. Computer Science Education, pp. 13. 137-. 10.1076/csed.13.2.137.14200.

Roy, K., Rousse, W., & DeMeritt, D. (2012). Comparing the mobile novice programming environments: App Inventor for Android vs. GameSalad. In Proceedings of the 2012 IEEE Frontiers in Education Conference (FIE) (FIE '12) (pp. 1-6). Washington, DC: IEEE Computer Society.

ISBN: 978-960-578-058-6

Sarlis, I. K. (2018, 6). Development of an Adventure Game for learning programming and coding in LOGO. Retrieved 6 20, 2019, from Open University of Cyprus - Kypseli Digital Repository: https://kypseli.ouc.ac.cy/handle/11128/3629?show=full

Swan, K. (1991). Programming objects to think with: Logo and the teaching and learning of problem solving. Journal of Educational Computing Research, 7(1), pp. 89-112.

von Wangenheim, C. G., & Shull, F. (2009, March). To Game or Not to Game? IEEE Software, 26(2), pp. 92-94.

Wabishabi, L. (2019). Why Some Students Dislike School but Love Education. Retrieved from Wabisabi Learning: https://www.wabisabilearning.com/blog/why-some-students-dislike-school-love-education

Wilson, A., & Moffat, D. (2010). Evaluating scratch to introduce younger schoolchildren to programming. Glasgow, Scotland, UK: Glasgow Caledonian University.

Περίληψη

Στο άρθρο αυτό παρουσιάζεται η κατασκευή ενός παιχνιδιού με ένα ενσωματωμένο περιβάλλον προγραμματισμού αρχαρίων για μαθητές Γυμνασίου, το οποίο και πρόκειται να χρησιμοποιηθεί για τη διδασκαλία της γλώσσας προγραμματισμού LOGO. Αυτό το περιβάλλον προγραμματισμού είναι ένα παιχνίδι το οποίο έχει σχεδιαστεί, εφαρμοστεί και δοκιμαστεί ειδικά για μαθητές Γυμνασίου προκειμένου να αποκτήσουν βασικές δεξιότητες προγραμματισμού. Το παιχνίδι ονομάζεται "Super Turtle Adventures" και βασίζεται στη μάθηση μέσω ψηφιακού παιχνιδιού. Βοηθά τους μαθητές να μάθουν τη γλώσσα προγραμματισμού LOGO, δημιουργώντας διάφορους γρίφους και παζλ, τα οποία πρέπει να επιλύσουν οι μαθητές χρησιμοποιώντας κώδικα LOGO. Οι μαθητές περνούν από το ένα επίπεδο στο επόμενο μέσω της επίλυσης των γρίφων. Το όφελος για τους μαθητές είναι αφενός ότι βελτιώνουν τις δεξιότητες προγραμματισμού τους και αφετέρου ότι μαθαίνουν να σέβονται το περιβάλλον μέσω της ανακύκλωσης. Ο προγραμματισμός αντιμετωπίζεται ως μια νέα ψηφιακή δεξιότητα ο οποίος πρέπει να κατακτηθεί από μαθητές του Γυμνασίου. Στον επίλογο του άρθρου τονίζεται ότι ο προγραμματισμός θα πρέπει να διδάσκεται χρησιμοποιώντας μάθηση μέσω ψηφιακού παιχνιδιού στα σχολεία και προτείνει το παιχνίδι "Super Turtle Adventures" ως μια εκπαιδευτική εφαρμογή ψηφιακού παιχνιδιού για τη διδασκαλία της LOGO στα σχολεία.

Λέξεις κλειδιά: Γλώσσα προγραμματισμού LOGO, Διδασκαλία Προγραμματισμού, Περιβάλλον προγραμματισμού αρχαρίων, Μάθηση μέσω ψηφιακών παιχνιδιών.

ISBN: 978-960-578-058-6