A Mobile Device Based Serious Gaming Approach for Teaching and Learning Java Programming

http://dx.doi.org/10.3991/ijim.v9i1.4380

T. Jordine[1], Y. Liang[2] and E. Ihler[1]
[1] Stuttgart Media University, Stuttgart, Germany
[2] University of the West of Scotland, Paisley, United Kingdom

***Abstract*—Most first year computer science students find that learning object-oriented programming is hard. Serious games have ever been used as one approach to handle this problem. But most of them cannot be played with mobile devices. This obviously does not suit the era of mobile computing that intends to allow students to learn programming skills in anytime anywhere. To enhance mobile teaching and learning, a research project started over a year ago and aims to create a mobile device based serious gaming approach along with a serious game for enhancing mobile teaching and learning for Java programming. So far the project has completed a literature review for understanding existing work and identifying problems in this area, conducted a survey for eliciting students' requirements for mobile gaming approach, and then established a mobile-device based serious gaming approach with a developed prototype of the game. This paper introduces the project in details, and in particularly presents and discusses its current results. It is expected that the presented project will be helpful and useful to bring more efficient approaches with new mobile games into teaching object-oriented programming and to enhance students' learning experiences.**

***Index Terms*—Java programming; mobile learning; mobile teaching; serious gaming.**

I. INTRODUCTION

In the last few years usage of mobile phones has rapidly increased and become more and more important to our daily life: check e-mails, surf in the Internet, chat with friends, play games, etc. Especially young students are using their mobile phones as modern communication devices and for playing mobile games [1]. Since mobile phones are mobile devices, they can be used by students in anytime anywhere, e.g. during idle times outside their university. In education, this means they also provide new opportunities for teaching and learning programming [2]. The current development of mobile phones (for example, providing more powerful hardware, better and larger touchscreen displays, better surfing ability) has made it even more realistic to use mobile-device based games as a new approach for teaching and learning subjects that are often difficult to teach and learn with traditional approaches in education.

This finding triggered the authors to start a research project for creating a new mobile-device based serious gaming approach for teaching and learning Java programming, as traditional approaches are often insufficient in such teaching and learning and our literature review has found that existing programming learning approaches (see Section II) often use desktop computers and currently there is no approach provided for teaching and learning Java programming using mobile phones, even though Java is one of the most popular object oriented programming language [3], [4] and often taught by universities for programming introduction.

In addition to a literature review, the project also conducted an initial survey (see Section III) recently. The result of the survey has explored the students' willing of having mobile games for learning Java programming with convenience. Both of the literature review and the survey encouraged us to continue with the project and create a new mobile-device based serious gaming approach that supports first year computer science students to learn Java during idle times with a mobile game for teaching and learning Java programming. This includes identification of a suitable game scenario that would keep students motivated when learning object oriented concepts and the Java syntax with the game. The project also faces new challenges (e.g. small screen size, CPU power, etc.) for new mobile game development.

The first prototype (see Section 4) of the mobile game has been completed already so far to support the proposed mobile-device based serious gaming approach.

This paper is structured as follows: Section II will show the existing work that has been done so far in this research area. Section III shows the results of our initial survey that has been conducted at the Stuttgart Media University. The game prototype will be introduced in Section IV. Section V will give an outlook of the planned authoring tool that will allow lecturers to create their own teaching materials. The paper will conclude with a discussion in Section VI and an overview of the future work in Section VII.

II. EXISTING WORK

During the last few decades many programming learning games have been created. This section will look at some of them within the four categories (programming learning games for desktop computers, programming learning games for mobile devices, Java teaching and learning games, authoring tools). The section will also discuss the problems with them for teaching and learning Java programming.

A. Programming learning games for Desktop computers

One of the first interactive programming learning games was Karel the robot [5] in the early 1980's. It provides a simple procedural programming language to control a robot on the screen. Jeroo [6] is similar to Karel the robot, but its object-oriented syntax is more close to C++ or Java, which allows an easier access to those languages. The NetLogo [7] framework was developed to teach undergraduate students object-oriented concepts with a simple syntax. As a feature, it provides a simple editor for creating its own levels. Alice [8] is a programming novice friendly language and does have a puzzle-piece based syntax, which is able to modify 3D objects in a 3D scene. A study that has been conducted at the Monmouth University has shown that the students' success rate was increased from less than 50% to 70% [9]. Scratch [10] has similar features (code puzzle pieces), but it focuses on 2D objects. PlayLogo3D [11] game was developed for early programming education and is based on the LOGO programming language.

B. *Programming learning games for mobile devices*

Later, the appearance of web techniques like HTML5 and CSS made it possible to play games using mobile device browsers. For example, E-Training DS [12] allows developers to design educational computer science games for mobile gaming devices. Also instructors can choose a predefined set of mini-games and adapt them for their own teaching purposes. As a special feature, a learning management system (e.g. Moodle) can be connected to each generated game to enable instructors or lecturers to design assessments and check students learning progress.

The authors did not find any empirical evaluation of the above mentioned mobile games and frameworks. However, a study [13] that was conducted in a primary school has confirmed that mobile learning games had positive impact on pupils' problem solving skills. By looking at a small study that was conducted in a primary school, it was found out that mobile learning games are having positive impact on pupils' problem solving skills [13].

C. *Java teaching and learning games*

Some Java learning games have been found for desktop computers. One is RoboCode [14] that plays in the same way like Karel the robot [5], and has a functionality supporting students to compete against each other. Another one is Code Spells [15] that is a role-play game for teaching Java by allowing players to move in a 3D world and requiring them to solve missions with spells. These spells are expressed with Java. Third one, Dream Coders [16] is also a role-play game with which players move in a 2D world, that is incomplete in its game play. Through programming assignments, players can improve playability and fun of the game. Dream Coders provides a framework for adopting an existing assignment and integrate it into courses. Forth one, Greenfoot [17], uses a different approach to provide a framework and an environment for creation of simulation-like, interactive applications on a two-dimensional grid. In terms of Greenfoot, objects displayed on the plane can be browsed through a UML (Unified Modeling Language) class browser. Classes can be instantiated and displayed as game objects on the grid. This Java introduction game frame-work has been used successfully at several universities and schools in order to teach students and pupils the basics of object-oriented programming (e.g. [18]–[20]).

Unfortunately the literature review conducted by the authors has explored that the current serious games for learning Java were only developed for non-mobile devices. This means a lack of the similar games for learning Java with mobile devices.

D. *Authoring tools*

Authoring tools in general allow course instructors to create their own teaching contents within a serious game for their students. Authoring tools for teaching programming are not found in publications by the authors though some authoring tools found available for other areas stated in the following.

E-Adventure [21] is an authoring tool for educational adventure games that supports desktop games and mobile games. It does not require game creators to have knowledge in game development. It does not particularly focus on learning programming. Any learning game can be produced using it. Nice-Game [22] is comparable to E-Adventure [21] and is a tool for educational content creation.

Second one is Emergo [23] which is a methodology and generic toolkit for developing and delivering scenario-based serious games that aims to teach students in higher education. It makes it easier to create serious games without having specific expertise. Third one is KAR [24] that provides a scenario generator for serious games. It provides an automatic method for generating content about every day activities through combining computer science techniques with the crowd. These scenarios can be used to train every-day activities in medical, military and commerce gaming applications. Forth one is U-Create [25] that attempts to make efficient content creation for interactive setups, mixed reality experiences and location-based services. It uses a graphical tool to create elaborated contents in a fast and easy way. Final one is the Create project [26]. It provides an instructional design-authoring tool that sup-ports entire process from identifying training requirements through the delivery of the trainings. Its developed prototype supports the upfront analysis and design process while enabling any game engine to be used as the development and delivery platform.

III. INITIAL SURVEY

In order to find out how the Java programming language and object-oriented concepts are taught at our university an initial survey was set up and completed in November 2013. The survey was split up into two parts:

1. How is Java currently taught at the university and how students perceive the teaching?
2. Is there a need of improvement and would students use a mobile Java learning game as an addition support for their study?

The survey was conducted in the form of online questionnaire at the Stuttgart Media University, Germany. 130 students were contacted and 98 of them actually completed the questionnaire for the survey. The result of the survey is shown and explained in the following.

73% of the respondents were male and the rest were female. More than 75% of them were in age between 21 and 26. 58% of them already had programming experiences before they started study of Java programming.

40% of the respondents do not use their phone for gaming purposes. In the rest 60%, 53% of them use their mobile for gaming up to five hours a week and 7% of them play more than five hours a week.

88% of the respondents used lecture notes and exercises from lab sessions in their study. 48% of them also looked at online tutorials. 4% of them used a learning game. 16% of them used other learning materials like books. When they are asked which of the learning materials or things was most useful to learn for their study, 59% of them regard exercises from lab sessions as most valuable things to do for study; 51% of them regard getting help from their fellow students as most useful thing to do; 42% of them regard accessing online tutorials as additional sources to use. When the respondents are asked if they had problems with learning Java, 43% of them answered "yes".

Furthermore, in order to find out where programming novices would cause the most of problems, the respondents were asked to rank the things that they thought difficult to learn in Java programming. Their answers are as follows (multiple answers were allowed):

TABLE I.
THINGS MOSTLY DIFFICULT TO LEARN

Things mostly difficult to learn	% of respondents
Object concepts, in general	55%
Understanding of the problem (domain) when they had to solve a programming problem	25%
Understanding logical flow in a program	24%
Handling data structures	23%

49% of the respondents had been taught Java with an objects-first approach as a first method [27]; 10% of them had been taught Java in a data structure and algorithm as a first method [28]. 41% of them had been taught with a mixture of both methods.

The survey also investigated if the mark of the mathematics exam is correlated with the mark of Java programming exam. A weak correlation (correlation coefficient: 0.35) was found for this. This means in reverse that students still could get a good mark for their Java programming exam even they did not get a good mark for their mathematics course. The survey also found a similar result for the impact of previous programming experiences on the mark of Java programming exam (correlation coefficient: -0.3).

The survey found that 40% of respondents would like to use a game as a tool for learning Java with their mobile devices, and 41% were undecided. 49% of respondents answered "yes" and 18% answered "no" when they were asked if a mobile Java learning game would help them. When they were asked where they would use a mobile teaching game for Java, most of them answered that they would use it during idle times, e.g. 61% at the bus stop and 51% at home (multiple choice was allowed).

Regarding the features of a mobile Java learning game, the survey found that 79% of the respondents thought of "practical exercises"; 66% considered "a good usability"; and 42% considered "checking their current learning progress" (multiple choice was allowed).

This survey has shown that nearly a half of them have had problems with learning Java in classrooms and students are willing to use their mobile phones for learning Java programming at their own times.

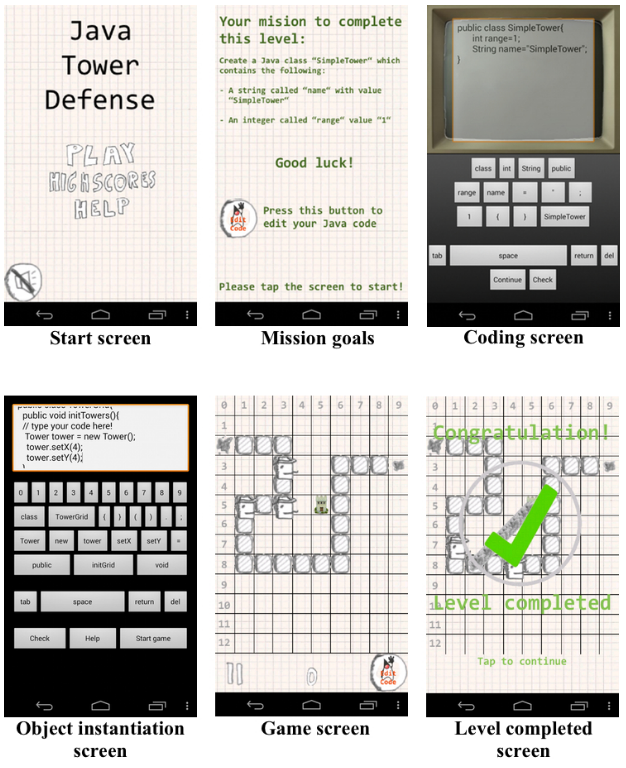
IV. A MOBILE DEVICE BASED SERIOUS GAMING APPROACH AND A MOBILE GAME PROTOTYPE

The following problems were identified by the conducted literature review and the initial survey:

- *No support for learning Java on mobile devices.* The games mentioned above do not support teaching Java on a mobile phone device. This problem means that a new approach of teaching Java and object-oriented concepts is needed in connection with mobile phones in order to attract young mobile phone users to learn Java.
- *Lack of a suitable help functionality for a mobile programming game.* Some of the existing tools do have an integrated help function (e.g. [6], [8], [14], [17]), but the function is very basic. This problem would decrease play of mobile games because the context sensitive help functionality would give more hints for each specific mission that students need to solve and help students to learn and practice each topic in theory. Especially in programming it is necessary to use the help function to understand the concepts and the specific API (Application Programming Interface).
- *Lack of using game design elements [29] known from traditional video games.* Many of existing approaches and games do not use game design elements (such as high scores, leader boards and bonus quests) as they should do. They often provide a customized programming environment with a code editor where behavior of game objects is coded and a game screen where the coded effects can be observed. This problem would lowed student motivation of using games as, according to existing game patterns [30] it is necessary to add incentives to a learning game in order to keep students motivated. For example, a course-wide high score could keep students motivated to solve a level correctly.
- *Many games for teaching and learning programming do not allow content creation for lecturers.* This problem would leave lecturers away of using games as teaching tools. In order to adapt each level to a specific lecture it is necessary to let games allow lecturers to create their own teaching contents. This can be a key problem for both of students and lecturers to use games if the games do not allow lectures to repeat a specific topic of a lecture not only in a lab session or tutorial but also during idle times in a mobile app.

By addressing the above problems, the second stage of our project tackled the lack of mobile device based games for teaching and learning Java and tried to find a suitable game scenario that would be fun to play and also impart programming knowledge to first year computer science students. This would create a mobile device based serious gaming approach for teaching and learning Java programming and develop a new mobile serious game for implementing the scenario.

This mobile java learning approach was not created to replace existing teaching methods such as lectures and lab sessions; the authors see the game as an extension to them. Based on the findings in existing works and the initial survey a first prototype was implemented. It should give students another opportunity to improve their program-



Your mision to complete this level:

Create a Java class "SimpleTower" which contains the following:

- A string called "name" with value "SimpleTower"

- An integer called "range" value "1"

Good luck!

Press this button to edit your Java code

Please tap the screen to start!

```
public class SimpleTower{
    int range=1;
    String name="SimpleTower";
}
```

```
  public void initTowers(){
  // type your code here!
   Tower tower = new Tower();
    tower.setX(4);
    tower.setY(4);
```

Figure 1. Game screens

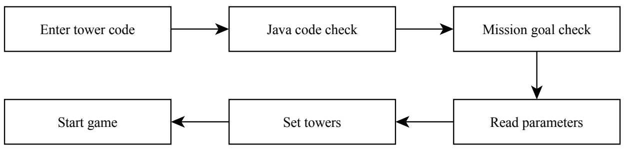


Figure 2. In-game code check workflow

ming skills, especially during idle times. This means for level design that each level should be playable within five minutes and a suitable game genre that fulfills all requirements that had been found.

After talking to students and experts, the authors have selected a tower defense scenario for the approach and the mobile game as it is flexible enough to add suitable and short missions and also students have known it already. Students will implement the behavior of towers via Java code. Then they will place the towers on the game grid by using programming code.

Figure 1 introduces a prototype of the game and its screens, which are explained in the following sub-sections. Each of the screens in the figure is explained in the following sub-sections.

A. *Start screen*

The start screen is used to start the game. It provides basic help functionalities and settings. It is planned to add a high score functionality to its next version so that students of a course can compare their games and scores with each other. This can motivate to solve all levels correctly.

B. *Mission goals screen*

The current goals for implementing the tower are described here. Students can also get hints to solve the level. For example, students have to implement a class where a simple tower is defined by Java code. In order to pass the level, they have to add and set default values to an integer and a string attribute.

C. *Coding screen*

This screen enables students to write their tower code on mobile phones. They will have the opportunity to check their code during writing, which is indicated by a red or respectively green button. Stu-dents can continue to see the next screen only when the code has passed the check. A mobile optimized Java keyboard is provided on this screen that enables students to write their code word-by-word rather than character-by-character. The words displayed on the keyboard are the words only needed to solve the level.

The code check is divided into two phases as shown in Figure 2. First, an integrated Java parser checks whether the syntax is correct. Then the game checks if the entered code matches the current task, which is done by the game by using regular expressions. It will include a strong help system in the next version of the software that will provide more detailed help and possibly link to external learning resources on internet if an error is found via the game.

D. *Object instantiation screen*

This screen allows students to instantiate the towers that have been written in the previous screen. A short introduction and help are shown when the screen is opened at the first time. The code would be checked, similarly to the previous screen, and analyzed with parameters. These parameters can be used e.g. to place towers on the game grid. An optimized mobile Java keyboard is also provided here.

E. *Game screen*

The game screen contains the enemy path, a grid and the instantiated towers. Enemies follow a track to an end point. When enemies cross the end point, the game is over. In order to prevent this, players have to place and code the towers cleverly. New towers can be added during playing the game. It will provide additional help functionality in this screen.

F. *Level completed screen*

This screen will be shown when a level has been completed correctly. Only the achieved score is shown at the moment. This screen will be improved with a more advanced scoring system. For example, code quality and other factors will be rated separately.

In addition, the game provides an integrated high score functionality for students that enables them to compete against each other; and an authoring tool that enables Java teachers to create their own contents and teaching materials for their own course.

V. AUTHORING TOOL

The authoring tool enables lecturers to create their own teaching materials for their own courses. In particular, lecturers can use the tool to design their own levels and link the levels to relating learning resources such as hints to solve the level, API documentation or web references for further information. Then they can use the created game in their lectures and lab sessions to help students understand and practice each topic of lectures using their mobile phones inside and outside classes.

This authoring tool enables lecturers to create new levels as long as students require them in the following steps:

1. Java lecturers use the tool to create and design the levels on a desktop computer first.
2. They then to deploy the game with the levels to a cloud server.
3. Then the game and levels on the cloud server are accessed by the registered students' mobile phones.
4. Students now download the levels by running the game on their mobile phones.

The authoring tool will also include a tracking feature that allows lecturers to collect the information that shows the current learning progress of their students. This information can be collected and displayed in terms of a management website that can be accessed via various mobile and desktop browsers.

VI. DISCUSSION

The approach presented in the above is complementary to existing programming teaching methods and will be used as an extension to the methods. Since the project is still in progress more things still need to be considered and done in the project. They are discussed in the following.

Firstly, the developed mobile game prototype will be tested by a wide range of students in universities of Germany and U.K. in order to find out to what extent the approach can support students in learning and understanding Java. A user study has already been set up in two universities of the two countries for this test.

Secondly, the game will be evaluated to find out how a level for the mobile game could be designed for leading a successful learning process for students. Existing work in the game-based learning area [30], [31] have proven the possibility of creating guidelines for content creators/Java instructors. Also other evidence could be found from the ACMs and IEEEs guidelines for teaching computer science [32]. This could be provided and implemented by the proposed authoring tool. Since the proposed game is optimized for mobile platforms, levels must be understandable and playable in a few minutes and not every existing tutorial is suitable for mobile learning. Therefore, it needs to consider how to make the Java code clearly arranged on small mobile phone screens. During the level design, it also needs to use a limited amount of keywords so that a keyword can be mapped to a single button on a screen.

Thirdly, at the moment, the game still has a prototypical user interface, which was useful for the first test mentioned in the above. But, the final version of the game needs to consider that most players are aged between 21 and 26 years (according to the finding of the initial survey). This means, a balance between effective and joyful user interface must be made to the final game.

Fourthly, the current prototype of the game only supports the Android platform as it has a more than 80% market share of mobile phones [33]. But students may use the mobile phones with Windows Phone or Apple iOS. Thus the final game also needs to support these platforms so that teachers can always use the proposed approach for a whole class, no matter which operating system is used by students' mobile phones.

Fifthly, the technical issue that mobile devices are having different screen sizes and resolutions needs to be thought. Initial tests of the game prototype have found that a display with at least 4.5 inches is required for precisely tapping each key on the coding keyboard.

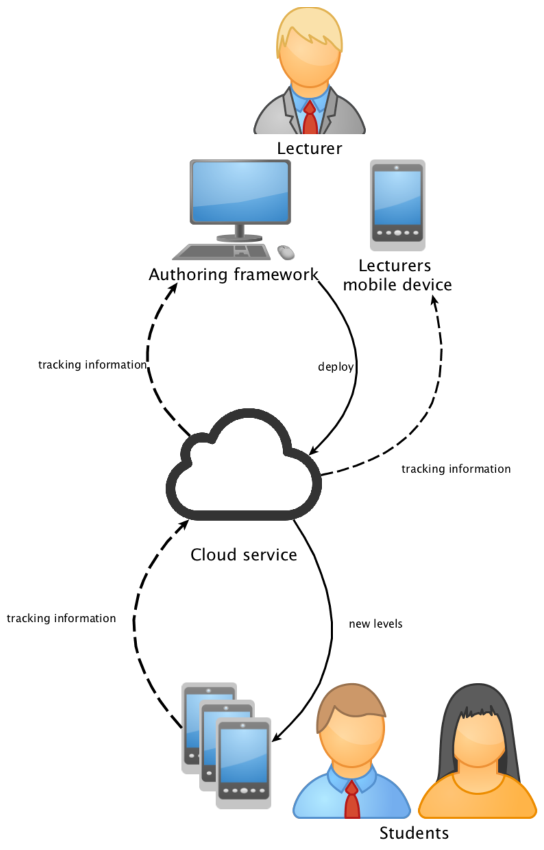


Figure 3. Proposed authoring tool

Finally, it is necessary for the game to provide another functionality that can send students' code to their teacher when they have any doubt about their code at each level.

VII. CONCLUSION AND FUTURE WORK

This paper has introduced a new mobile device based serious gaming approach for teaching and learning Java programming. It has shown existing works in Section II and summarized an initial survey in Section III that was conducted at the Stuttgart Media University at an earlier stage of a research project that is currently carried out by the authors. It then presented a prototype of the mobile game in Section IV that was created and implemented by the authors for the project. In Section V, it introduced an authoring tool that would be developed as a part of the game. Then it discussed the remained issues and improvements of the game in in Section VI.

In the next stage of the project a user study will be conducted at two universities of Germany and U.K. with two separate Java programming courses. The results of this study will show to what extent the Java learning game can support students in learning Java programming. This study will be carried out in two phases: the first phase will carry out the test of the current prototype of the game; the second phase will firstly analyze the result of the first test, then revise the code of the prototype, and finally test the revised game towards a final version.

In addition to the user study, a usability test will also be conducted in the next stage in order to find out the pragmatic and hedonistic qualities of the game. This will be done by using the AttraktDiff 2 test [34].

REFERENCES

[1] S. Khalaf, "Apps Solidify Leadership Six Years into the Mobile Revolution," Apr-2014. [Online]. Available: http://blog.flurry.com/bid/109749/Apps-Solidify-Leadership-Six-Years-into-the-Mobile-Revolution . [Accessed: 14-Jan-2015].

[2] D. Frohberg, C. Göth, and G. Schwabe, "Mobile Learning projects - a critical analysis of the state of the art: Mobile learning projects," *J. Comput. Assist. Learn.*, vol. 25, no. 4, pp. 307–331, Jul. 2009. http://dx.doi.org/10.1111/j.1365-2729.2009.00315.x

[3] "TIOBE Software: Tiobe Index." [Online]. Available: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html . [Accessed: 14-Jan-2015].

[4] S. O'grady, "The RedMonk Programming Language Rankings: January 2014," *tecosystems*. [Online]. Available: http://redmonk.com/sogrady/2014/01/22/language-rankings-1-14/ . [Accessed: 14-Jan-2015].

[5] R. E. Pattis, *Karel the Robot: A Gentle Introduction to the Art of Programming*, 1st ed. New York, NY, USA: John Wiley &amp; Sons, Inc., 1981.

[6] D. Sanders and B. Dorn, "Jeroo: a tool for introducing object-oriented programming," in *Proceedings of the 34th SIGCSE technical symposium on Computer science education*, New York, NY, USA, 2003, pp. 201–204. http://dx.doi.org/10.1145/611892.611968

[7] M. Dickerson, "Multi-agent simulation and netLogo in the introductory computer science curriculum," *J Comput Sci Coll*, vol. 27, no. 1, pp. 102–104, Oct. 2011.

[8] S. Cooper, W. Dann, and R. Pausch, "Alice: a 3-D tool for introductory programming concepts," in *Journal of Computing Sciences in Colleges*, 2000, vol. 15, pp. 107–116.

[9] K. Johnsgard and J. McDonald, "Using Alice in Overview Courses to Improve Success Rates in Programming I," in *IEEE 21st Conference on Software Engineering Education and Training, 2008. CSEET '08*, 2008, pp. 129–136.

[10] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai, "Scratch: programming for all," *Commun ACM*, vol. 52, no. 11, pp. 60–67, Nov. 2009. http://dx.doi.org/10.1145/1592761.1592779

[11] I. Paliokas, C. Arapidis, and M. Mpimpitsos, "PlayLOGO 3D: A 3D Interactive Video Game for Early Programming Education: Let LOGO Be a Game," in *2011 Third International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, 2011, pp. 24–31. http://dx.doi.org/10.1109/VS-GAMES.2011.10

[12] R. Tornero, J. Torrente, P. Moreno-Ger, and B. F. Manjón, "e-Training DS: An Authoring Tool for Integrating Portable Computer Science Games in e-Learning," in *Advances in Web-Based Learning – ICWL 2010*, X. Luo, M. Spaniol, L. Wang, Q. Li, W. Nejdl, and W. Zhang, Eds. Springer Berlin Heidelberg, 2010, pp. 259–268.

[13] J. Sánchez and A. Salinas, "Science Problem Solving Learning Through Mobile Gaming," in *Proceedings of the 12th International Conference on Entertainment and Media in the Ubiquitous Era*, New York, NY, USA, 2008, pp. 49–53.

[14] J. O'Kelly and J. P. Gibson, "RoboCode & problem-based learning: a non-prescriptive approach to teaching programming," *SIGCSE Bull*, vol. 38, no. 3, pp. 217–221, Jun. 2006. http://dx.doi.org/10.1145/1140123.1140182

[15] S. Esper, S. R. Foster, and W. G. Griswold, "On the nature of fires and how to spark them when you're not there," in *Proceeding of the 44th ACM technical symposium on Computer science education*, New York, NY, USA, 2013, pp. 305–310.

[16] J. K.-W. Chang, L. H. Dang, J. Pavleas, J. F. McCarthy, K. Sung, and J. Bay, "Experience with Dream Coders: developing a 2D RPG for teaching introductory programming concepts," *J Comput Sci Coll*, vol. 28, no. 1, pp. 227–236, Oct. 2012.

[17] P. Henriksen and M. Kölling, "greenfoot: combining object visualisation with interaction," in *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, New York, NY, USA, 2004, pp. 73–82.

[18] R. Hijón-Neira, Á. Velázquez-Iturbide, C. Pizarro-Romero, and L. Carriço, "Improving Students Learning Programming Skills with ProGames – Programming through Games System," in *Human-Computer Interaction – INTERACT 2013*, P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson, and M. Winckler, Eds. Springer Berlin Heidelberg, 2013, pp. 579–586.

[19] M. Jonas, "Teaching Introductory Progamming Using Multiplayer Board Game Strategies in Greenfoot," *J Comput Sci Coll*, vol. 28, no. 6, pp. 19–25, Jun. 2013.

[20] T. Vilner, E. Zur, and S. Tavor, "Integrating Greenfoot into CS1: A Case Study," in *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education*, New York, NY, USA, 2011, pp. 350–350.

[21] P. Lavin-Mera, J. Torrente, P. Moreno-Ger, J. Vallejo-Pinto, and B. Fernández-Manjón, "Mobile Game Development for Multiple Devices in Education," *Int. J. Emerg. Technol. Learn. IJET*, vol. 4, no. 2009, pp. 19–26, 20091024.

[22] E. B. Arias, D. G. Vico, S. A. Herrera, and J. Q. Vives, "A web tool to create educational content with gaming visualization," in *Frontiers in Education Conference (FIE), 2012*, 2012, pp. 1–6.

[23] R. J. Nadolski, H. G. K. Hummel, H. J. van den Brink, R. E. Hoefakker, A. Slootmaker, H. J. Kurvers, and J. Storm, "EMERGO: A methodology and toolkit for developing serious games in higher education," *Simul. Gaming*, vol. 39, no. 3, pp. 338–352, Jan. 2008. http://dx.doi.org/10.1177/1046878108319278

[24] S. Sina, S. Kraus, and A. Rosenfeld, "Using the Crowd to Generate Content for Scenario-Based Serious-Games," *ArXiv14025034 Cs*, Feb. 2014.

[25] S. Sauer, K. Osswald, X. Wielemans, and M. Stifter, "U-Create: Creative Authoring Tools for Edutainment Applications," in *Technologies for Interactive Digital Storytelling and Entertainment*, S. Göbel, R. Malkewitz, and I. Iurgel, Eds. Springer Berlin Heidelberg, 2006, pp. 163–168. http://dx.doi.org/10.1007/11944577_16

[26] S. E. Kirkle, S. Tomblin, and J. Kirkley, "Instructional design authoring support for the development of serious games and mixed reality training," in *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2005, vol. 2005.

[27] S. Cooper, W. Dann, and R. Pausch, "Teaching objects-first in introductory computer science," in *Proceedings of the 34th SIGCSE technical symposium on Computer science education*, New York, NY, USA, 2003, pp. 191–195. http://dx.doi.org/10.1145/611892.611966

[28] A. Ehlert and C. Schulte, "Empirical comparison of objects-first and objects-later," in *Proceedings of the fifth international workshop on Computing education research workshop*, New York, NY, USA, 2009, pp. 15–26. http://dx.doi.org/10.1145/1584322.1584326

[29] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining 'gamification,'" in *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, New York, NY, USA, 2011, pp. 9–15.

[30] S. Kelle, "Game Design Patterns for Learning," Nov. 2012.

[31] "Gameplay Design Patterns Wiki," 2014. [Online]. Available: http://gdp2.tii.se/ . [Accessed: 14-Jan-2015].

[32] "IT 2008 Curriculum — Association for Computing Machinery." [Online]. Available: http://www.acm.org/education/curricula/IT2008%20Curriculum.pdf/view . [Accessed: 14-Jan-2015].

[33] J. Ong, "Android Achieved 85% Smartphone Market Share in Q2," *The Next Web*, 2014. [Online]. Available: http://thenextweb.com/google/2014/07/31/android-reached-record-85-smartphone-market-share-q2-2014-report/ . [Accessed: 14-Jan-2015].

[34] M. Hassenzahl, M. Burmester, and F. Koller, "AttrakDiff: Ein Fragebogen zur Messung wahrgenommener hedonischer und pragmatischer Qualität," in *Mensch & Computer 2003*, Springer, 2003, pp. 187–196.

AUTHORS

T. Jordine received the B.Sc. and M.Sc. degree in computer science and media at the Stuttgart Media University in 2009 and 2011. In the beginning of 2013 he started his PhD studies in cooperation with the University of the West of Scotland and the Stuttgart Media University. Besides his research activities he is also an academic assistant at the Stuttgart Media University, Nobelstr. 10, 70569 Stuttgart, Germany (e-mail: jordine@hdm-stuttgart.de)

Y. Liang, is a senior lecturer in the School of Engineering and Computing, the University of the West of Scotland, Paisley PA1 2BE, U.K. (e-mail: ying.liang@uws.ac.uk).

E. Ihler is a professor at the Stuttgart Media University. He is teaching in the area of software engineering and theoretical computer science. Stuttgart Media University, Nobelstr. 10, 70569 Stuttgart, Germany (e-mail: ihler@hdm-stuttgart.de).

This article is an extended and modified version of a paper presented at the 2014 Frontiers in Education Conference (FIE2014), held from October 22-25, 2013, in Madrid, Spain. Submitted 25 December 2045. Published as resubmitted by the authors 25 January 2015