This is the published version of a paper presented at *European Conference on Games Based Learning 2017*.

Citation for the original published paper:


N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-31798
Learning to Program by Building Learning Games

Peter Mozelius1 and Marie Olsson2
1Department of Computer and Systems Sciences, Mid Sweden University, Sweden
2Department of Computer and Systems Sciences, Stockholm University, Sweden
Peter.Mozelius@miun.se

Abstract: Digital games and digital gaming have had a fast expansion in the 21st century and today the vast majority play some kind of digital games. From a teacher’s perspective an interesting type of games are the so called learning games and the didactic idea of Game-based learning (GBL). In the last decade GBL has been an emerging field and several researchers have pointed out its strong learning potential. The aim of this study is to describe and discuss the idea of learning to program by game construction. The overall research strategy has been action research where the case study approach has been combined with a content analysis of submitted learning games. Data has been collected from course evaluation questionnaires, postings in online discussion fora and game analyses. Involved informants have been kept totally anonymous and game analyses are also presented with respect for personal integrity. Findings show that game construction can be an appreciated as well as stimulating way of designing course assignments, especially in programming courses involving multimedia and graphical user interfaces. The quality of students’ submissions show a wide variety and are strongly dependent on students pre-knowledge. Some of the student built learning games were found to have a potential for reuse as additional learning tools in introductory programming courses. However, to what degree and how needs to be assessed in a future study.

Keywords: game-based learning, game construction, constructionism, programming education, technology enhanced learning

1. Introduction

Programming education has in several studies been pointed out as a problematic area with a need for redesign and new teaching techniques (Lahtinen, Ala-Mutka & Järvinen, 2005; Gomes & Mendes, 2007; Olsson & Mozelius, 2015). Researchers have in the 21th century reported about low motivated students and high drop-out rates in programming courses at university level (Wiedenbeck Labelle & Kain, 2004; Bennedsen & Caspersen, 2007; Mozelius, Torberg & Calderon Castillo, 2015). Students seem to have problems with basic syntax, as well as with more abstract programming concepts (Gomes & Mendes, 2007). Today, after the casual revolution analysed and described by Juul (2010), the major part of the population is playing different kind of digital games. The so called casual gamers and the hardcore gamers both spend considerable time in front of screens playing casual games and hardcore games (Juul, 2010). Research studies have reported that games can have strong motivational effects (Malone, 1981; Yee, 2006; Wiklund & Mozelius, 2013) and reactivate the students that have lost their motivation for traditional teaching (Wiggins, 2016).

From a programming teacher’s perspective the didactic idea of Game-based learning (GBL) is interesting to reinforce the current instructional design. Several researchers have pointed out the learning potential of combining games and learning (Ebner & Holzinger, 2007; Tobias, Fletcher & Wind, 2014). GBL today is an integrated part of different educational concepts where four interesting branches are:

- GBL with tailor-made educational games (Malliarakis, Satratzemi & Xinogalos, 2013)
- GBL by playing commercial of-the-shelf games (Van Eck, 2006)
- GBL to support social inclusion of disaffected youth (Wiklund et al, 2014)
- Learning to program with the idea of game construction (Olsson & Mozelius, 2016)

This study has explored the opportunity to combine points 1 and 4, in an attempt to use the described potential of GBL to motivate students. In two programming courses for third and fourth year university students, the project assignments should be solved by constructing learning games. Hopefully, some of the learning games could be used as tools for self-learning in introductory programming courses.

1.1 Problems

- There is an identified problem with high drop-out rates for low motivated students in programming courses at university level (Wiedenbeck Labelle & Kain, 2004; Bennedsen & Caspersen, 2007; Mozelius, Torberg & Calderon Castillo, 2015).
At the same time as research studies highlight the strong potential in GBL (Gee & Hayes, 2012; Tobias, Fletcher & Wind, 2014) it still is hard to find educational programs that can be used in programming courses and especially for the Python programming language.

1.2 Research questions

The problem described above under A. and B. could partly be solved by a study providing answers to:

- In what ways could GBL and game construction stimulate learning in programming education at university level?
- How might student constructed learning games be reused as resources for self-learning in introductory programming courses, and are they appropriately designed?

2. Extended background

One difficult aspect of learning programming is to learn different coding concepts and at the same time implement the concepts with a correct executable syntax for the actual programming language. There are several ways to achieve this, but this study had a focus on GBL, game construction and constructionism.

2.1 GBL for programming education

Game-based learning has been a hyped research field in the 21st century and has been tested with promising results in various educational contexts (Tobias, Fletcher & Wind, 2014). GBL should not be seen as a standalone concept, but rather a didactic idea that preferably could be combined with other learning methodologies (Sierra et al., 2016). In programming education there is a tradition of problem oriented learning (O'Kelly & Gibson, 2006) and also of constructionism as the fundamental didactic idea (Brennan & Resnick, 2012).

A straight-forward idea of including GBL in programming courses is to purchase and use so called commercial-of-the-shelf (COTS) games (Van Eck, 2006). For a History teacher or a subject matter expert in natural languages this could be easily implemented, but when it comes to programming languages in general and Python in particular, there are not that many games to buy. The concept of academic staff designing and developing their own-designed tailor-made educational games is interesting (Malliarakis, Satratzemi & Xinogalos, 2013), but also depending on resources and costs.

Authors' GBL approach is to use game construction in programming courses, with the idea that the process of designing and building games will include most of the fundamental programming (Olsson & Mozelius, 2016) and at the same time stimulate students' creativity (Mozelius, 2016).

2.2 Game construction for programming education

Since programming courses essentially are about increasing students' programming skills and knowledge, the concept of students constructing their own digital games seems natural and congenial (Shabalina et al., 2012). Some researchers have suggested to start out at university level by learning programming concepts without actually writing any code (Kazimoglu et al., 2012), but the most common approach in Computer science programmes still is to start writing coding as early as possible.

This paper is a study on students' game construction, and if the artefacts they create in project assignments might be of use as learning games in introductory programming courses.

2.3 Constructionism for programming education

Constructionism as a specialisation of constructivism is an idea originating from Seymour Papert, but as he has stated himself: "... it would be particularly oxymoronic to convey the idea of constructionism through a definition since, after all, constructionism boils down to demanding that everything be understood by being constructed." (Papert & Harel, 1991). However, the constructionist approach to learning has a strong emphasis on the importance of students engaging in the development of artefacts (Kafai & Resnick, 1996).

Compared to Jean Piaget's ideas of constructivism, Papert's constructionism is more focused on the art of learning, or 'learning to learn', based on the making of things in learning (Ackermann, 2001). With the
constructivist approach learning should be seen as a process of active construction, rather than a passive transmission of knowledge. New knowledge would better be constructed by the learners themselves, through an active process based on their previous knowledge (Hadjerrouit, 1999).

Papert tested his constructionist approach as a new way to teach and learn Mathematics, but constructionism has also been a frequently used idea in Programming education (Boyer, Langevin & Gaspar, 2008; Konecki, 2014), and in the learning of Computational thinking for younger students (Brennan & Resnick, 2012 ; Bers et al., 2014). Constructionism is a serious learning theory, but as highlighted by Papert it should include playfulness as well (Papert & Harel, 1991).

3. Methods and data collection

The overall research strategy has been action research, with the definition of action research as a systematic study combining action and reflection with the intention to improve and reform the existing practice (Cohen, Manion & Morrison, 2007). In this study the existing practice to reform is the instructional design in programming courses at university level. The way to improve instructional design in programming education is to study and analyse programming education. To gather the data necessary to answer the research questions a case study approach has been combined with a content analysis. The actual research design is inspired by the book 'Practical research: Planning and Design' by Leedy & Ormrod (2016).

According to Leedy & Ormrod, a content analysis should be "a detailed and systematic examination of the content of a particular body of material for the purpose of identifying patterns, themes or biases". In this study the particular body consists of learning games submitted as solutions to project assignments in two programming courses, where one of the authors is the subject matter expert. Content analyses can be found in a variety of disciplines such as arts, education, history, journalism and political science (Leedy & Ormrod, 2016).

Data can in case studies preferably be gathered with multiple sources and in a mix of methods (Creswell, 2009). Case studies should also have a focus on one, or a few instances of the selected phenomenon, with the aim to result in an in-depth description of activities and processes in the particular instances (Denscombe, 2014). The case study researcher should also record and describe details about the context surrounding the cases, and the history of the studied cases (Leedy & Ormrod, 2016).

Data has been collected in a combination of answers in course evaluation questionnaires, postings in online discussion fora and analyses of learning games in the courses 'Multimedia programming in Python' and 'Game-based learning', during 2015 and 2016. Earlier both courses have had substantially larger groups, but due to regulations of the intake, 86 students registered for Multimedia programming in Python and 29 students registered for Game-based learning in 2015. In 2016, 85 students registered for Multimedia programming in Python and 36 students for the course on Game-based learning. For Multimedia programming in Python only around 40 students actually started the course in 2016, and for Game-based learning as many as 34 started the course. The differences in completion rates between the courses could probably be explained by the fact that Multimedia programming in Python is an independent, standalone summer course, and that Game-based-learning is a mandatory course for third year programme students.

Informants have been kept as anonymous as possible and game analyses are also presented with respect for personal integrity. Authors have tried to follow the ethical guidelines described by Leedy & Ormrod (2016) to avoid exposing course participants.

4. The programming courses

What both programming courses have in common is that game-based learning and game construction is a core theme involved in the major part of the course content. In a final project assignment, an educational game should be constructed with a given educational topic. Examples of educational game topics in earlier course batches are Mathematics, IT-security and natural language training. In the 2015 course batches all student projects was about building an educational game on IT-Security.

In course evaluations students have classified IT-Security as a dry topic and asked for themes that better could stimulate the construction of imaginative game worlds. For the 2016 course batches the themes were IT-
4.1 Multimedia programming in Python

Practically the entire syllabus is built on the idea to learn multimedia programming in Python by game construction. Lectures, tutorials and the course book, all have a focus on analysis and synthesis of various kinds of classic digital games. Almost all assignments are based on building games or creating game mechanics, initially basic game ideas but later with an increased complexity. Graphical user interfaces are initially created with Tkinter and later most students build their learning games with the use of PyGame.

Course batches have the last five years had a variation of groups between 50 – 400 students from universities all over the nation. There have been large variations in the age groups with the youngest students below 20 years old, and the oldest so far 65. The course is given as a summer course with a 50% study pace from early June to late August. There is an initial face-to-face lecture, but all activities can be completed in distance mode with online facilitation, online discussion and recorded lectures and tutorials.

4.2 Game-based learning

This course is also given with a 50% study pace with a course syllabus based on ten lectures, five assignments and a final project, where content and examination is a mix of theory and practical work. Compared to the multimedia programming course there is more of theoretical input, such as research papers on game construction and a course book explaining various motivational models and learning theories. Guest lecturers introduce various techniques for design of educational games that also involves pedagogical, didactical and accessibility aspects. Assignments are given as a mix of essays on theoretical subjects and concrete programming exercises where students build learning games.

Course batches have had a variation span between 30 to 80 students from two Bachelor’s programmes on Digital media and Digital game construction. During the two last years it has only been students from the Digital game construction programme with minor changes in the syllabus. In the final project assignment students should design, discuss and implement a learning game prototype with the use of HTML5, CSS, JavaScript and jQuery.

5. Findings and discussions

Findings confirm that GBL and game construction is a didactic idea that can motivate students. The games built in the projects also includes advanced programming techniques as well as creative solutions. Several findings are common for both courses, but there are also variations that motivate the division below in 5.1 and 5.2 in this first parts of the chapter, that strive to answer the first research question.

5.1 Multimedia programming in Python

Out of around 300 applicants the intake was only 90 students this year, which is a result of new limitations for the maximum amount of students that Swedish universities get reimbursement for. Not more than 40 students started the course, 30 students completed all assignments, and 21 students completed all assignments and the project to the second deadline. The relatively low completion rate might be a result of the project requirements, where the learning game also should be complemented with a demonstration video and detailed documentation. However, the submitted learning games have a technical standard that is the highest ever if compared to the six earlier course batches, and the completion rate is relatively high for a summer course given on distance.

The course was initially designed to be a promotion course for programming, with relatively low prerequisites and average grades for enrolment. According to new enrolment rules students’ pre-knowledge has increased during the last years. One student had learnt to program ten years earlier with no later practice, she did not complete the course and had problems to solve even the initial basic assignments. Several course participants were from technical universities with good programming skills, but not with experience of game construction.

The most common answer to the question about the idea to build learning games in Python was "Well, why not?". A student from a technical university wrote that, "A fun topic where it is easy to see the results of your effort, if compared to signal handling in a programming language". But there were also answers like, "A fun topic,
but I had problems to find a design idea for a game where I could apply the programming techniques I now”. The problem of game design was also brought up online in a discussion forum. To help students with the design phase research articles were posted in the facilitation forum.

To the question about if the construction of educational games is a good way to learn new programming techniques, course participants were more clear and positive with answers like: "Absolutely", "Definitely" and "Yes, I believe you can learn a lot by building games". Students also had a general positive attitude towards Python and PyGame compared to other programming languages and add-on libraries that they had experienced earlier. The response rate was, as in earlier course batches, relatively low. One plausible explanation is that students from other universities are not familiar with the tailor-made evaluation system used at the department.

The quality of the constructed learning games were surprisingly high and the best ever for this course. One third of the students met all grading criteria for the highest grade of A and all submitted project solutions were approved. Of the given game themes, History was the most popular one, Programming second and with only two games on IT Security. What seems like a probable explanation for the popularity of games on historical eras is the rich possibilities for creating imaginative gaming environments. The games on IT Security had more strict user interfaces and backstories.

Finally, most of the learning games for programming were hybrids combining classical games like Asteroids, Memory and Snake, with basic programming learning content. Five games were selected for further analysis based on learning content, usability and gameplay.

5.2 Game-based learning

This years’ course group consists, like last year, of students exclusively from a game construction programme were practically all participants are familiar with most types of games (Mozelius, 2014). In earlier course versions there have only been students from the design track of the game construction programme, but this year there were also students from the construction track. Students from the construction track has taken more programming courses and are in general on a higher level of programming proficiency. These students added value two the project groups and the games in the project assignment were, like in the multimedia course, of higher quality than previous years.

For these students, games and game construction is a core concept in their Bachelor’s programme. From this viewpoint it was surprising to find answers in the evaluation questionnaire like, “I find this approach good, especially for us game programme students, that seldom have to build games in other courses”. Authors have contacted subject matters on the game constructing programme to check to what degree and what types of games that are built on students’ previous courses, since this also was brought up in some of the examination seminars. Anyhow, these students have a pre-training that make them suitable for building educational games, even if this is the first course where theory on educational games is presented and discussed.

There were also answers like, “I found it relevant to build learning games” but also students that wrote: “Not a bad idea but a bit of a compromise” and “It should have been more focus on GBL than on programming, but I find the course design good”. Most students agreed on that game construction is a way to learn programming techniques, and almost all had a positive attitude towards the choice of HTML5, CSS, JavaScript and jQuery. Some students argued for a use of game engines like Unity and HTML5 frameworks like Phaser. Unity has been tested in earlier course batches but been removed since it does not really support the learning objectives.

Like in the multimedia programming course the most popular theme was learning games on History, with Programming as number two, and that the least chosen topic was IT-Security. Many of the games on History have sophisticated graphics and thrilling backstories based on ancient Greek and Roman History. One learning game on Greek Philosophy would probably fit in well in Philosophy lessons in upper secondary school. However, there were less games on Programming, but one creative solution was a prototype for a game on programming, where the player also must consider real world phenomena like, food, sleep and money to support the programming activities
Peter Mozelius and Marie Olsson

5.3 Content analysis of submitted games

The learning games were of surprisingly high quality in both batches of both courses but with the variations described above. Results from the content analysis were grouped into four categories that are relevant to answer the second research question. Found categories were Backstory, Game mechanics, Usability and Learning content.

Backstory

Comparing the various given game topics the games on History by far had the most engaging backstories and often with tailor-made graphics and music. Backstories are rich and from various eras from 'Ancient Greece' to '1960s Politics' where students have spent quite a lot of time on narration and checking historical facts. Not surprising, but anyhow obvious how introductions, narration and backstories were much shorter and dryer in games on IT-Security and Programming. In general the students in the Game-based learning course had richer backstories which can be explained by their previous courses on game construction.

Game mechanics

The game complexity and the technical level of game mechanics show wide variations in both courses. Students with less programming skills often go for 'quizzes in disguise' where multiple-choice questions are combined with multi-media effects. Advanced game mechanics often provides a rich gaming experience but there are also games that are built by complex programming where the gameplay is quite boring. In general the games from the Multimedia programming course had more advanced game mechanics which probably can be explained by the rich resources in the PyGame add on package in combination with students high programming skills. Several participants in this course were on their summer leave from programmes on Software engineering.

Usability

To answer the second research question usability is an important aspect in learning games should be reused in introductory programming courses. Some of the analysed games with engaging backstories and challenging game mechanics were problematic even to start up and play. This can often be solved by reading the required manual or watching the demonstration video. However, if games should be possible to reuse in programming courses they need to have good usability and an understandable gameplay. Students taking the course on Game-based learning were mostly hardcore gamers preferring more complex games that should be played in longer time spans and where the player have to try hard to find a winning strategy.

Learning content

Last, but not least, learning content and learning outcomes must always be essential. Whatever engaging backstory, brilliant game mechanics or excellent usability games have the overall objective must be games that have a potential to improve students programming skills or programming knowledge. Important also with games involving fundamental concepts.

5.4 Game selection for reuse in introductory programming courses

Based on the four categories presented above five games were selected for further evaluation. They all have relatively thin backstories but high usability and relevant learning content where students can improve their knowledge about fundamental programming concepts. All games are developed in the Python programming language and also presenting techniques and concepts in Python code. We find this important since the introductory course were the games later will be tested is given in Python. Most game have a medium level of game mechanics, and some of them are at an advanced level, but there are never any usability issues since all but one game are variations on well-known games such as Memory, Asteroids and Snake. They also have in common that the game span is relatively short, which is important if the gaming sessions should be part of relatively short workshops.
6. Conclusions

GBL based on game construction seems like a promising concept, especially for both these courses where students have good pre-knowledge of programming. However, students lacking basic programming skills might find it stressful. Authors’ GBL idea for novice programming students is to play learning games instead of building them, and that these games should be relatively short games, with high usability and relevant learning content. Rich backstories and ambient game worlds are of course seen as positive game design, but not necessarily as main criteria in a selection for introductory programming courses.

Findings confirm earlier findings that GBL is a good concept for programming courses (Shabalina et al., 2012; Olsson & Mozelius, 2016), but students with poor programming proficiency might find it problematic to design games at the same time as learning basic programming techniques. Authors’ GBL idea for students with less pre-knowledge and poor programming skills is instead to play learning games on fundamental programming concepts. With a basic idea similar to what has been presented in earlier research as ‘Brain-Compatible Learning’ (van Niekerk 2016). Provided with the basic building blocks of programming, learners will be prepared for further self-learning.

7. Future work

Next step for the authors is to further analyse and chose some of the selected games for use in an introductory programming course. Results will be evaluated and presented in another paper that we hope can be presented at the same conference as this paper. Both studies are part of the project "Learning to Program by Playing Learning Games" that can be followed at ResearchGate: https://www.researchgate.net/project/Learning-to-Program-by-Playing-Learning-Games

A more general continuation would be to let students build more learning games in other programming courses, but with another course design and with more input on game design. Two ideas are to have extra guest lectures on game design, and to make research articles like this one part of the course content.

References