Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 181 (2021) 247–254

Procedia Computer Science

www.elsevier.com/locate/procedia

CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies 2020

Evaluating Aspects of Usability in Video Game-Based Programming Learning Platforms

Jaime Díaz[a], Jeferson Arango López[b], Samuel Sepúlveda[a], Gabriel Mauricio Ramírez Villegas[c], Danay Ahumada[d], Fernando Moreira[e,f,*]

[a]*Universidad de La Frontera, Depto. Cs. de la Computación e Informática, Temuco, Chile*
[b]*Universidad de Caldas, Depto. de Sistemas e Informática, Manizales, Colombia*
[c]*Universidad Nacional Abierta y a Distancia, ECBTI, Bogotá, Colombia*
[d]*Universidad Católica de Temuco, Depto. Procesos Diagnósticos y Evaluación, Temuco, Chile*
[e]*REMIT, IJP, Universidade Portucalense, Porto, Portugal*
[f]*IEETA, Universidade Aveiro, Aveiro, Portugal*

Abstract

Teaching computer programming is an important topic. Due to Science and Technology initiatives, these topics are considered in different training cycles. For higher education, students must cultivate fundamental concepts for the development of software applications, which not only contribute to the knowledge of programming languages but also to opening guidelines for computational thinking. However, selecting a proper tool can be complex. Especially for the diversity of alternatives on the web. Further, not all of them meet basic usability requirements. In this study, we present a set of platforms that seek to develop programming skills based on video games. The search consisted of 4 stages: (i) definition of the research questions, (ii) scope review, (iii) execution of search and (iv) platform selection. Finally, we employ a usability heuristic evaluation for a novice programming system to determine best practices.

Keywords: Human-Computer Interaction; Usability; Video-games; Training; Computer Programming

* Corresponding author. Tel.: +351914163038.
E-mail address: fmoreira@upt.pt

10.1016/j.procs.2021.01.141

1. Introduction

Computer science-related initiatives are now being integrated into the undergraduate level in engineering. At this stage, students must acquire the concepts for software development that not only contribute to knowledge of programming languages, but also open new guidelines for computational thinking and problem solving.

Learning to program brings benefits at all levels [1]–[3], but it requires an additional effort: assimilating languages and tools is not always aligned with "*everyday*" aspects, where advances are also individual and do not encourage aspects of collaboration or inclusion.

This has been solved in various ways, with one of the study topics being the use of video games. These are considered good alternatives that reward "*productive effort*", a type of effort that results in attractive and motivating learning [4]–[6]. When games-based learning systems are compared with conventional assessment methods, the literature shows that games give students greater support in retaining knowledge, concentrating and performing at a higher level of execution [7]–[10]

Platforms based on video games, or that have the mechanics for these, have two important advantages in terms of self-learning: (i) they allow customized and adaptable learning, where the difficulty increases as progress is made, allowing the student to "*go back*" and review, and generating automatic feedback of their actions. Second, (ii) these are online and accessible via the Internet, i.e., students can access contents when they need to.

However, there has been a proliferation of tools aimed at developing programming skills. This availability of systems also makes it difficult to select an option that fulfills expectations. Moreover, for educators and professionals, comparing the quality of these tools in specific contexts is complex.

Usability is a key concept in the area of human-computer interaction (HCI). According to the ISO standard [9], this is defined as "*The extension where a product can be used by specific users to reach specific goals with efficiency, effectiveness and satisfaction in a context of certain use*". Thus, all interactive products have user interfaces, with their quality being a critical factor for meeting objectives [11], [12].

The aim of this study is to analyze the level of usability compliance on the basis of a heuristic evaluation, of 6 initiatives recommended by technology magazines obtained by search engines. These results seek to replicate the exercise of professionals or students in search of support alternatives.

The technique applied provides a practical methodology to determine the quality of systems by identifying potential problems. Therefore, the authors present a formal assessment of these systems as well as the findings of appraisals by end users (engineering students).

The results seek to be a support for decision-making with respect to levels of difficulty, usability and better practices for the tools of the future.

2. Methodology

This section presents the search of the recommended systems. An appropriate initial search is critical to carry out a subsequent analysis of these platforms. The aim was to capture those platforms available to teach programming through video games and then to assess the levels of usability compliance through a heuristic evaluation.

The initial search was done on three different engines - Google, Bing and DuckDuckGo - to reduce results bias. However, according to statistics tracker *NetMarketShare (https://www.netmarketshare.com/)*, the global breakdown of the use of search engines on portable / desktop computers / mobile devices is approximately 78% through Google.

If we take this information into account, the results from the platforms recommended by digital magazines and "*fast access*" should come from this platform, since it fulfills the overlap criterion of an average user.

2.1. Platform Selection

The search consisted of 4 stages: (i) definition of the research questions, (ii) scope review, (iii) execution of search and (iv) platform selection.

Definition of the research question. The research question was meant to determine the video game platforms that can provide computer programming skills. The proposed RQ is the following: (RQ1) *Which video game platforms*

offer options to teach computer programming skills?

Scope review. At this stage, the terms used to conduct the search were defined. Note that certain terms are accompanied by synonyms to expand the results. **Inclusion criteria:** Platforms available online, accessible via the Internet, that offer personalized and adaptable learning skills and that teach at least one programming language.

Exclusion criteria: Only supported for mobile platforms, do not have a free version or that were offline when the study was conducted. In this light, all the platforms that offer programming courses like traditional ones were beyond the scope here: CodeAcademy, Coursera, edX, Udemy, etc.

Execution of the Search The search consisted of the application of queries formed by the following key words: "*video games" "game*" "*teach*" "*learn*" "*programming*" "*code*". When executed in the Google search engine, 4 technology magazine listings were selected: Business Insider, Medium Mybridge, Makeuseof and Eduonix. All these appeared on the first results page, fulfilling the search parameters. The queries entered into the Google engine were the following: *"video game teach programming", "games for learning programming", "learn to code game".*

Platform selection. The application of the methodology is shown in Table 1. This shows the results of the platforms that stood out for having at least two technology magazines listings. A technology magazine is understood as any type of non-academic publication, in a news style, that reports on the contingency related to the world science and technology. All these appeared as results in the Google search engine.

Table 1. Platforms identified

Platform	Business Insider	Medium - Mybridge	Makeusof	EduOnix	**Total**
CodeCombat	X	X	X	X	**4**
CodeMonkey	X	X	--	X	**3**
CodinGame	X	X	X	--	**3**
VIMAdventures	X	X	X	--	**3**
Code Wars	X	X	X	--	**3**
CodeHunt	X	--	X	--	**2**
RoboCode	X	--	X	--	**2**
CheckIO	X	X	--	--	**2**
CyberDojo	X	X	--	--	**2**
ElevatorSaga	X	X	--	--	**2**
RubyWarrior	X	X	--	--	**2**
Screeps	--	X	X	--	**2**

12 platforms were selected, of a total of 250 that offer services to learn programming based on video games. In a second iteration, the inclusion/exclusion criteria were refined:

- It teaches at least one programming language: (i) *VIMAdventures* does not teach a language, but rather effectively uses a text editor known as VIM. (ii) *Screeps* does not teach programming; instead, the game is based on programming mechanics in order to be used. It also has no Internet version.
- Platforms available online: (iii) *RoboCode* is an application that must be downloaded for use.
- They are operational: (iv) The platform *CodeHunt* was offline at the time of the study.
- They offer personalized and adaptable learning skills: (v) CyberDojo and (vi) ElevetorSaga are platforms that present problems to be solved with programming algorithms, but do not have any follow-up elements for teaching-learning processes.

3. Results

Table 2 presents the platforms selected, in order of difficulty (statement by the platform creators themselves) according to the user's knowledge in programming skills: value (1): oriented to K12 (primary and secondary education), value (2) higher education students, not related to information technologies; and value (3) higher education students in programming subjects. The remainder of the columns describes the name, type of subscription, available languages and programming languages offered.

Table 2. Comparison of platforms selected.

Platforms	Subscription	Language	Languages	Difficulty
Code Combat	Free and paid	English, Spanish, French, German, others.	Python, Javascript, HTML, CSS, JQuery, Bootstrap	1
Code Monkey	Free and paid	English, Spanish, French, German, others.	Coffeescript	1
Ruby Warrior	Free	English	Ruby	2
CodeWars	Free	English	C, Java, Coffeescript, Python, Ruby, PHP, C#, Javascript, etc.	3
CheckIO	Free	English	Javascript, Phyton.	3
CodinGame	Free	English and French	C#, Javascript, Java, C++, Python3, etc.	3

Of the 6 resulting platforms, two groups were differentiated: (i) oriented to the introduction of basic concepts: CodeCombat, CodeMonkey, RubyWarrior and (ii) development of logic skills: CodeWars, CheckIO and CodinGame.

3.1. Tool assessment

At this evaluation stage, the heuristic proposals by Kolling and Mckay [13] were used to determine the level of usability compliance in novice programming systems. This proposal is based on the universal studies by Nielsen [14], but it incorporates specific points of view for programming learning platforms.

This approach makes possible a formal evaluation at low cost, since it is done by specialists in the area. The number of evaluators follows the cost-benefit model recommended by Nielsen and Landauer for such initiatives [15]. Table 3 presents the 13 heuristics used to evaluate the platforms.

Implementation of the evaluations. *Participants.* Three specialists in usability and Internet platform inspection mechanisms. All three had performed this exercise previously, have an academic degree in computer sciences, and are between 30 and 40 years of age. All the evaluators have given programming classes to undergraduate students in engineering. *Context.* Platforms were sought that help improve programming skills, centered especially on self-teaching for university undergraduate level. *Evaluation process*. Each participant was asked to assess all the platforms, using Kolling and Mckay's heuristics [13]: Code Combat; Code Monkey; Ruby Warrior; CheckIO; Code Wars; CodinGame.

Table 3. Heuristics of Usability Evaluation by Kolling and Mckay.

Heuristic	Heuristic
H1: Engagement	H8: Clarity
H2: Non-threatening	H9: Human-centric syntax
H3: Minimal language redundancy	H10: Edit-order freedom
H4: Learner-appropriate abstractions	H11: Minimal viscosity
H5: Consistency	H12: Error-avoidance
H6: Visibility	H13: Feedback
H7: Secondary notations	--

The evaluators entered the platform, registered if necessary, and used it on average for 15 - 20 minutes. This time range was defined, as this is the average period of engagement to capture potential users' attention. In the event of critical problems during this time, end users ultimately switch platforms. For each of the heuristics, they were evaluated with grades from 1 (poor fulfillment) to 5 (high fulfillment).

Finally, the evaluators made general observations for each of the platforms, thus making qualitative observations of the positive and negative aspects that will be discussed later. Table 4 provides the results of the evaluation.

Table 4. Evaluation Results.

H	Code Combat	Code Monkey	Ruby Warrior	CheckIO	Code Wars	Codin Game
H1	4.67	4.67	3.33	4.33	4.33	4.50
H2	4.33	4.67	4.00	4.33	5.00	4.50
H3	3.67	4.33	4.00	4.00	5.00	4.00
H4	4.33	4.67	**2.33**	2.67	4.00	4.00
H5	4.67	4.33	3.33	3.00	4.33	4.00
H6	4.67	4.33	3.33	3.33	4.33	5.00
H7	3.67	**2.33**	**2.67**	3.00	4.00	4.00
H8	3.00	4.67	4.00	3.33	4.33	5.00
H9	4.00	4.67	**2.67**	**2.67**	3.00	**2.00**
H10	3.67	4.33	3.67	**2.67**	3.33	**2.00**
H11	3.67	5.00	4.00	3.67	3.67	3.50
H12	4.33	4.67	**2.67**	3.00	3.67	3.00
H13	4.67	5.00	**1.67**	**2.67**	3.00	**2.50**

In general terms, all the heuristics had an average value over 3.17. This is something positive and is because we were evaluating the best selection of platforms. The standard deviation of the study stayed within a maximum range of 0 - 1.53. Those with the best overall performance were:

- *H1 - Engagement (4.31)*: The system must involve and motivate students. It must stimulate the students' interest or kindle a sense of fun. Each of the platforms uses mechanisms based on video games, or on aspects of "*gamification*", which develops a positive aspect of constant engagement.

- *H2 – Non-threatening (4.47):* The system must not seem threatening in its appearance or behavior. The users must feel safe knowing that they can experiment without breaking the system or losing data. Again, the platforms, regardless of their novice level of difficulty, were developed to improve skills in a simple and friendly way.

By contrast, the worst-performing heuristics were:

- *H7 – Secondary notations (3.28)*: The system should provide secondary notations automatically when they represent utility. The users should be able to add their own secondary notations when they add an additional value. However, only one platform (*CodeCombat*) offers this option by providing internal features (not those of the platform) in several languages. This may be because, being education platforms, personalization occurs on a second level.
- *H9 – Human-centric syntax (3.17)*: The platform notation must employ user-centered syntax. The syntactic elements must be easy to read, avoiding complex or technical terminology for the target audience. In this standard, a general problem is produced: each uses a real programming language employed in the industry. The programming language has its own syntax that is not always the friendliest for beginners.
- *H10 – Edit-order freedom (3.28)*: The interface must allow the user freedom in the order in which they choose to work. Users must be able to leave the tasks partially finished and return to them later. Although some platforms allow this feature (*RubyWarrior*), this function is not a standard feature for the rest. If the student does not finish the phase or the challenge, they must begin from zero in that section.

3.2. Preliminary adoption model

A preliminary survey was taken to evaluate two of the platforms: "*Code Combat*" and "*Code Monkey*". Both were available in Spanish, and targeted K12 (novice programming course). Of the 50 participants, half participated for each one. All the participants were aged between 18 and 20 years; 60% were men and 40% were women. All were in the first or second cycle of engineering.

Following a Method Adoption model [16], the individual's behavior was assessed, considering their beliefs, standards and intentions of use. The evaluated constructs were: perceived ease of use (PEU), perceived utility (PU) and use intention (UI).

Perceptions and intentions were captured through a questionnaire (adapted from [17]). Each of the responses was evaluated on a 5-point Likert scale. The score obtained for each of the constructs is equivalent to the average of each question. Table 5 shows the results of the experiment.

Table 5. MAM Experiment Results

	Average *Code Combat*	Standard Deviation *Code Combat*	Average *Code Monkey*	Standard Deviation *Code Monkey*
PU	4.00	0.96	3.48	1.39
UI	3.84	0.80	3.40	0.82
PEU	3.88	0.60	4.40	0.87

The highest values for each of the platforms were: Code Combat, PU (average: 4.00; standard deviation: 0.96). For Code Monkey, PEU (average: 4.40; standard deviation: 0.87).

The values of the constructs are the result of the first platform "Code Combat" having a more noticeable intention of the video game concept (it is medieval). By contrast, "Code Monkey" targets a younger audience (with the use of vibrant colors and simplicity in its interfaces).

Although both platforms achieved high values in their assessment (all above 3.40), it cannot be concluded that they will be relevant throughout the entire period of the course.

4. Conclusions and future work

We presented the analysis of a set of tools that are available online to gain programming skills for software engineering students. In comparison to traditional methods, this type of tools provides the student with a methodology based on gamification patterns that generate a better user experience. In addition, it increases the level of motivation in activities related to learning programming.

We must remember that this experiment was in the context of teaching programming to undergraduate students in engineering, where not everyone has a natural affinity for these sciences nor sees a real utility for them when they find examples very distant from their reality.

It is interesting to analyze the number and quality of the tools currently available on the Internet for the development of programming skills. The ability to teach computer science-related concepts on all levels is assumed to be a latent need. Likewise, the robustness and consistency of the platforms evaluated are worthy of note.

The results of review include two types of approaches: (i) those oriented to the introduction of basic concepts, with CodeMonkey standing out among the platforms evaluated, and (ii) those oriented to the development of logic skills, where CodeWars is noteworthy for proposing a learning community based on gamification concepts.

A disadvantage is related to language, where only 2 of the selected platforms have an option to deal with all the topics in Spanish. This could be an entry barrier for those who do cannot work in English. The optimal solution to this issue would be a combination of platforms that serve as a support to the current teaching-learning processes. It is not possible to place all the responsibility for the basic knowledge of the respective subject on an external platform.

Teachers can use these tools for educational purposes as a complement to traditional methodologies. It is important to be clear that these playful tools do not replace those methodologies.

As future work, we want to implement experiments in classrooms where the learning levels of certain programming topics can be measured by applying labs with these tools. In addition, we want to obtain information about motivation levels and define a catalogue of gamma patterns that can be applied to this area of knowledge. Also, we consider to generate a training platform that automatically measures the progress of a student who follows gamification processes in comparison with another student who does not.

Acknowledgements

The authors wish to thank all the participants involved in the experiments, particularly the members of the "Centro de Estudios en Ingeniería de Software, CEIS" and “User Experience & Game Design - Research Group, UXGD”. UXGD is member of HCI-COLLAB. This work was financed by FCT– *Fundação para a Ciência e a Tecnologia, I.P., project UIDB/05105/2020.*

References

[1] S.-C. Kong, M. M. Chiu, and M. Lai, “A study of primary school students’ interest, collaboration attitude, and programming empowerment in computational thinking education,” *Comput. Educ.*, vol. 127, pp. 178–189, Dec. 2018, doi: 10.1016/j.compedu.2018.08.026.

[2] A. Bray and B. Tangney, “Technology usage in mathematics education research – A systematic review of recent trends,” *Comput. Educ.*, vol. 114, pp. 255–273, Nov. 2017, doi: 10.1016/j.compedu.2017.07.004.

[3] R. van Roy and B. Zaman, “Need-supporting gamification in education: An assessment of motivational effects over time,” *Comput. Educ.*, vol. 127, pp. 283–297, Dec. 2018, doi: 10.1016/j.compedu.2018.08.018.

[4] J. Gonzalez, H. Pomares, M. Damas, P. Garcia-Sanchez, M. Rodriguez-Alvarez, and J. M. Palomares, “The use of video-gaming devices as a motivation for learning embedded systems programming,” *IEEE Trans. Educ.*, vol. 56, no. 2, pp. 199–207, May 2013, doi: 10.1109/TE.2012.2208194.

[5] D. H. Schunk, J. L. Meece, and P. R. Pintrich, *Motivation in education : theory, research, and applications*. Pearson, 2014, p. 436.

[6] R. Garris, R. Ahlers, and J. E. Driskell, “Games, Motivation, and Learning: A Research and Practice Model,” *Simul. Gaming*, vol. 33, no. 4, pp. 441–467, Dec. 2002, doi: 10.1177/1046878102238607.

[7] S. Diaz, J. Diaz, and D. Ahumada, “Virtual Reality in High School: A Systematic Mapping Study,” *2018 37th International Conference of the Chilean Computer Science Society (SCCC)*. 2018, doi: 10.1109/sccc.2018.8705266.

[8] R. J. Segura, F. J. Pino, C. J. Ogáyar, and A. J. Rueda, “VR-OCKS: A virtual reality game for learning the basic concepts of programming,” *Comput. Appl. Eng. Educ.*, vol. 28, no. 1, pp. 31–41, Jan. 2020, doi: 10.1002/cae.22172.

[9] C. González-González, P. Toledo-Delgado, C. Collazos-Ordoñez, and J. L. González-Sánchez, “Design and analysis of collaborative interactions in social educational videogames,” *Comput. Human Behav.*, vol. 31, pp. 602–611, Feb. 2014, doi: 10.1016/j.chb.2013.06.039.

[10] D. Schoene, S. R. Lord, K. Delbaere, C. Severino, T. A. Davies, and S. T. Smith, “A Randomized Controlled Pilot Study of Home-Based

Step Training in Older People Using Videogame Technology," *PLoS One*, vol. 8, no. 3, p. e57734, Mar. 2013, doi: 10.1371/journal.pone.0057734.
[11] Iso 9241-210, "ISO 9241-210:2010 - Ergonomics of human-system interaction - Part 210: Human-centred design for interactive systems," *International Organization for Standardization, Geneva, Switzerland.*, 2010. .
[12] J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons, 2015.
[13] M. Kölling and F. McKay, "Heuristic Evaluation for Novice Programming Systems," *Trans. Comput. Educ.*, vol. 16, no. 3, pp. 12:1–12:30, Jun. 2016, doi: 10.1145/2872521.
[14] J. Nielsen, *Usability Inspection Methods*. 1994.
[15] J. Nielsen and T. Landauer, "A Mathematical Model of the Finding of Usability Problems," in *Proceedings of ACM INTERCHI'93*, 1993, pp. 206–213.
[16] M. A. Fishbein and I. Ajzen, *Belief, attitude, intention and behaviour: An introduction to theory and research*, vol. 27. Reading, MA: Addison-Wesley, 1975.
[17] O. N. C. Fernandez, "Un procedimiento de medición de tamaño funcional para especificaciones de requisitos," Universitat Politècnica de València, 2007.