A Roadmap to Gamify Programming Education

Jakub Swacha
University of Szczecin, Poland
jakub.swacha@usz.edu.pl

Ricardo Queirós
CRACS – INESC-Porto LA, Portugal
uniMAD, ESMAD/Polythechnic of Porto, Portugal
ricardoqueiros@esmad.ipp.pt

José Carlos Paiva
CRACS – INESC-Porto LA, Portugal
DCC – FCUP, Porto, Portugal
up201200272@fc.up.pt

José Paulo Leal
CRACS – INESC-Porto LA, Portugal
DCC – FCUP, Porto, Portugal
zp@dcc.fc.up.pt

Sokol Kosta
Aalborg Universitet, Denmark
sok@es.aau.dk

Raffaele Montella
Università Degli Studi Di Napoli “Parthenope”, Italy
raffaele.montella@uniparthenope.it

Abstract

Learning programming relies on practicing it which is often hampered by the barrier of difficulty. The combined use of automated assessment, which provides fast feedback to the students experimenting with their code, and gamification, which provides additional motivation for the students to intensify their learning effort, can help pass the barrier of difficulty in learning programming. In such environment, students keep receiving the relevant feedback no matter how many times they try (thanks to automated assessment), and their engagement is retained (thanks to gamification).

While there is a number of open software and programming exercise collections supporting automated assessment, up to this date, there are no available open collections of gamified programming exercises, no open interactive programming learning environment that would support such exercises, and even no open standard for the representation of such exercises so that they could be developed in different educational institutions and shared among them. This gap is addressed by Framework for Gamified Programming Education (FGPE), an international project whose primary objective is to provide necessary prerequisites for the application of gamification to programming education, including a dedicated gamification scheme, a gamified exercise format and exercises conforming to it, software for editing the exercises and an interactive learning environment capable of presenting them to students. This paper presents the FGPE project, its architecture and main components, as well as the results achieved so far.

2012 ACM Subject Classification Social and professional topics → Computer science education

Keywords and phrases gamification, programming, learning, automatic assessment, programming exercises

Digital Object Identifier 10.4230/OASIcs.ICPEC.2020.26

Funding This paper is based on the work done within the Framework for Gamified Programming Education project supported by the European Union’s Erasmus Plus programme (agreement no. 2018-1-PL01-KA203-050803).

1 Introduction

Skilled programmers are in high demand in the European Union countries, and the EU has been focusing on different initiatives to increase the number of such experts, calling coding *the 21st century skill* [5]. One of the key obstacles to satisfy this demand is the difficulty in learning programming [3]. We believe a progress in this area can be attained with the combined use of automated assessment, which provides fast feedback to the students experimenting with their code, and gamification, which provides additional motivation for the students to intensify their learning effort. In our opinion, the availability of programming courses based on such an approach may not only improve the effectiveness of learning programming, but also extend the group of people feeling capable of effectively learning it. Nonetheless, to the best of the authors' knowledge, there are yet no available open collections of gamified programming exercises, no open interactive programming learning environment that would support such exercises, and even no open standard for the representation of such exercises, so that they could be developed in different educational institutions and shared among them.

In this paper, we present a work-in-progress on a framework for application of gamification to programming education, realized as an international project within the scope of the Erasmus+ programme, key action: *Cooperation for innovation and the exchange of good practices* [7]. The scope of this project includes the specification of the gamification scheme and the exercise definition format, and the development of a collection of gamified exercises for several popular programming languages and software: a toolkit for editing the exercises and an interactive learning environment so that they could be given to the students. The target group of the project are programming instructors and students learning programming (also self-teaching). All the project outputs will be freely available on the Internet under open-source licenses. The expected impact of the project is an improvement in the efficiency of programming education and its student-perceived experience.

The remainder of this paper is organized as follows. Section 2 presents an overview of the existing private platforms and open source tools related to gamified programming. Then, Section 3 provides a brief insight of the proposed framework, including details on its architecture, design, and implementation. Also, it enumerates and describes the intellectual outputs (IOs) of the project and their current status. Following, Section 4 presents the first results achieved by the project team. Finally, Section 5 summarizes the main contributions of this research.

2 Related Work

Many companies and educational institutions are investing in solutions to teach programming to young students. A good example of this is the European Coding Initiative [11], a project supported by the European Commission that aims at *promoting coding and computational thinking at all levels of education.* This initiative provides a series of documents and guidelines for teachers on how to teach programming at different student levels and for students on how to practice and learn coding. Google has introduced the *Code with Google* initiative [8], where students can attend remote coding classes and work in teams to learn programming. Moreover, Google has different competitive programming contests, where coders solve a set of problems in a certain amount of time and they get ranked based on their performance [9].

Similar to Google's Coding Competitions, many other online tools exist, such as HackerRank [10], TopCoder [20], LeetCode [14], among others. They all present students with hundreds or thousands of problems and rank them using several metrics, such as number of problems solved, total points collected by solving each problem, efficiency of the solution, among others. Quite often, recruiters use these platforms to identify skilled programmers to offer them a chance for a job interview [1]. However, these successful platforms are owned and managed by private companies, meaning that there is no collaboration among them. One direct consequence of this factor is that the same programming problems are being rephrased and entered multiple times in the different platforms.

Open-source solutions for competitive programming contests exist and have been adopted by different universities to teach coding. Some of the most prominent ones are Kattis [12], DOMjudge [6], DMOJ [4], and Mooshak [13]. Being open-source, educators can install these tools in their servers and provide a means for students to compete with each other. Unfortunately, each tool uses a self-established format for storing exercises [15, 16, 13, 12, 18], without adhering to a common format, which hinders the sharing of these problems. As a consequence, an open collection of problems is not yet available to educators, so they will have to spend their time creating the problems in the tool of their choice.

In addition to that, competition is the only motivational element typically included with automated assessment tools out-of-the-box. However, too much focus on competition can be harmful to those who lose [21], which is commonly the case of students with learning difficulties. For instance, gamification, which includes but is not limited to competition, is an area of great interest towards the engagement of students in (programming) education [2]. Notwithstanding, there is a complete lack of formats/specifications for its application in educational contexts, increasing the disparity of implementations.

3 Framework Architecture and Planned Outcomes

The five IOs planned for the FGPE project are the following: (IO1) Gamification Scheme for Programming Exercises; (IO2) Data Exchange Format for Gamified Programming Exercises; (IO3) Tools Supporting Editing and Conversion of Programming Exercises; (IO4) Programming Courses featuring Gamified Exercises; and (IO5) Programming Learning Environment featuring Gamified Exercises.

In Fig. 1, we present the conceptual architecture of the framework based on these outputs and their relations.

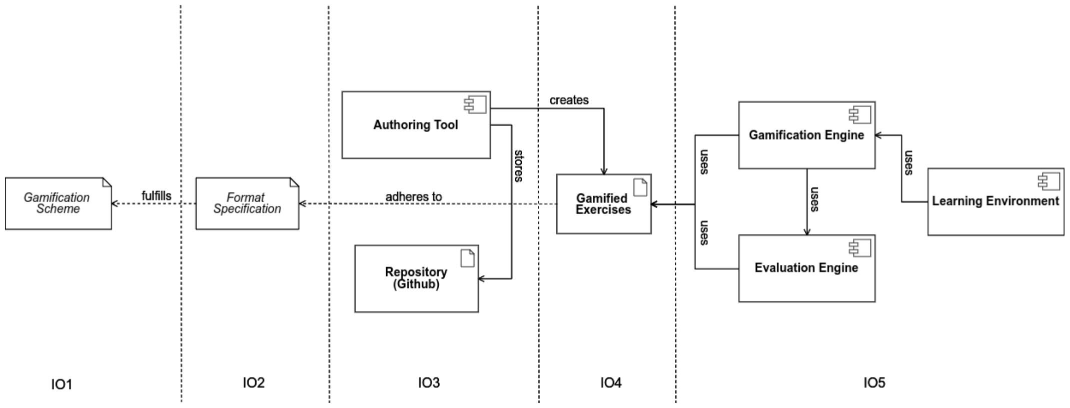


Figure 1 Conceptual architecture of the framework.

The IO1 – Gamification Scheme for Programming Exercises – aims to provide easy-to-follow guidelines on how to apply gamification to programming courses. The expected result of this IO is a reference scheme for gamification of programming courses.

In IO2 – Data Exchange Format for Gamified Programming Exercises –, the goal is to foster data reusability and interoperability in a gamified programming education framework, more precisely: programming exercises' data and gamification-related data. This will be achieved through the definition of specification formats that fulfil the gamification scheme created in the previous IO.

The IO3 – Tools Supporting Editing and Conversion of Programming Exercises – aims to simplify both the process of authoring and storing exercises and the binding of the gamification layer to a set of exercises. In order to accomplish these goals, a tool to author content adhering to the formats designed in IO2 will be created. This tool will integrate with Github to store files and data.

The IO4 – Programming Courses featuring Gamified Exercises – will provide a base set of ready-to-use exercises, created with the tools developed in IO3 and conforming to the formats defined in IO2, that addresses the needs of the most popular programming courses taught by the project partners.

Finally, the IO5 – Programming Learning Environment featuring Gamified Exercises – consists of the development of a free and open-source learning environment that will use the gamification and the evaluation engines to engage and assess students' performance. This learning environment will consume the exercises developed at IO4.

4 First results

Currently, the first three IOs are completed. The remaining two are still in progress, and not yet suitable to be presented.

The first IO involved the development of the reference scheme for gamification of programming courses. It resulted in the identification of gamification concepts and techniques applying them to programming education, as well as the use cases for gamified programming exercises and their matching to the respective gamification techniques. The results of this study were validated with a survey administered to students of five distinct universities, and described in a previous publication [17].

Having selected the gamification concepts of interest for programming education, a study to gather the requirements bound to those concepts for the specification of a gamified exercise format has been conducted [19]. As a preliminary result, this study unveils the importance, for reusability and interoperability, of separating the two main kinds of data present in a gamified programming education framework: programming exercises' data and gamification-related data. The former includes data relating to the full life-cycle (creation, selection, presentation, solving, and evaluation) of a programming exercise, and even though a standard format or, at least, a consensus is yet to be found, it is well-grounded in the literature due to its extensive use by automated assessment tools [15, 16, 13, 12, 18]. The latter conveys all the elements that should be included to foster the motivation and enjoyment of students during the realization of the programming exercises (e.g., challenges and rewards), and is the novelty of the study as there is a complete absence of formats for its application in education (not only programming education).

This distinction led to the development of two independent formats for IO2, namely YAPExIL and GEdIL, as well as a tool to support the authoring of content adhering to these formats, the AuthorKit, for IO3.

4.1 YAPExIL

Yet **A**nother **P**rogramming **Ex**ercises **I**nteroperability **L**anguage (YAPExIL) is a JSON schema for describing a programming exercise package, based on PExIL [15] – an XML dialect that aims to consolidate all the data required in the programming exercise life-cycle. YAPExIL aims for simplicity while covering most features of a task package as described by Verhoeff's model [22], i.e., a unit for collecting, storing, archiving, and exchanging all information concerning with a programming task.

YAPExIL is composed of four facets: metadata, presentation, evaluation, and tools. Metadata facet contains identification information about an exercise, such as title, author, keywords, module in which it is contained, among others. Presentation facet includes what is presented to the student when he/she opens the exercise as well as instructions displayed to teachers for reusing exercises. Evaluation facet encompasses all the components used in the automated assessment of the exercise. Finally, the tools facet holds extra external scripts that are not strictly necessary at any phase of the programming exercise life-cycle.

4.2 GEdIL

Gamified **Ed**ucation **I**nteroperability **L**anguage (GEdIL) is a JSON schema designed to describe gamification layers for educational contexts. Although designed to fulfil the specific requirements of gamification applied in programming courses [19], GEdIL is sufficiently generic to be applied to any other educational subjects as it only delineates a layer with the gamification elements that should lay on top of another layer describing activities.

GEdIL defines a hierarchy of challenges where each level (including the root) may have rules, rewards, and leaderboards. The root node has metadata for identifying the layer. Leaf nodes refer to activities, bridging to bottom layers. In this way, the gamification layers are easily replaceable and may reuse the same activities.

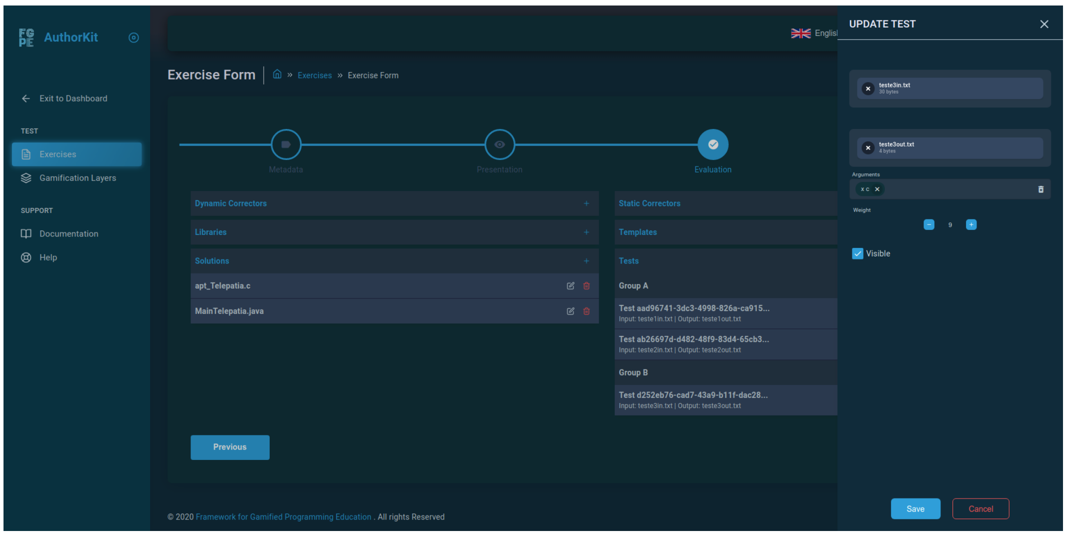
4.3 AuthorKit

AuthorKit is a web application to author educational content adhering to the formats of the FGPE (YAPExIL and GEdIL). Each author can create/edit/view educational content inside projects he/she owns and projects shared with him/her by other authors.

The user interface, presented in Fig. 2, is characterized by two separate form wizards nested under a project: one to create programming exercises and another to create gamification layers. The former maps each facet of YAPExIL to a step of the wizard, having the possibility to upload all the necessary files for a programming exercise (e.g., problem statement, usage instructions, input/output files for tests, and solutions). The latter has steps to (1) manage the metadata of the gamification layer, (2) add rewards that are not linked to a challenge, (3) wire the rules of the course, (4) create the leaderboards by defining their metrics, and (5) build the challenge tree, where each challenge may have child challenges and/or local scoped rules, leaderboards, and rewards. All data, including files, is synchronized to a repository within the GitHub account linked to the project owner's account of AuthorKit.

5 Conclusion

In this paper, we presented our work-in-progress in the area of programming education, particularly targeting the aspect of student engagement. Our contribution will advance the state of the art by addressing the gap of the lack of open collections of reusable gamified programming exercises by providing: *(1)* a frame of reference for programming



apt_Telepatia.c
MainTelepatia.java

Group A
Test aad96741-3dc3-4998-826a-ca915...
Input: teste1in.txt | Output: teste1out.txt
Test ab26697d-d482-48f9-83d4-65cb3...
Input: teste2in.txt | Output: teste2out.txt
Group B
Test d252eb76-cad7-43a9-b11f-dac28...
Input: teste3in.txt | Output: teste3out.txt

teste3in.txt 30 bytes
teste3out.txt 4 bytes
Arguments: x c
Weight: 9
Visible ✓

© 2020 Framework for Gamified Programming Education . All rights Reserved

Figure 2 Editing a test on the Evaluation step of the exercise form wizard (dark theme).

course gamification (including featured gamification concepts and the intended area of their application); *(2)* the specification of a format for exchanging gamified programming exercises based on the above frame of reference; *(3)* tools for authoring exercises in the above format; *(4)* a programming learning environment allowing to set up and manage gamified programming courses making use of such exercises. These four components constitute the scope of the Framework for Gamified Programming Education project [7].

References

1 Bharat Adibhatla. Top 5 coding websites companies hire from, 2019. accessed on 20 Jan 2020. URL: `https://analyticsindiamag.com/top-5-coding-websites-companies-hire-from/`.

2 Manal M. Alhammad and Ana M. Moreno. Gamification in software engineering education: A systematic mapping. *Journal of Systems and Software*, 141:131–150, 2018. `doi:10.1016/j.jss.2018.03.065`.

3 Yorah Bosse and Marco Aurélio Gerosa. Why is programming so difficult to learn? patterns of difficulties related to programming learning mid-stage. *SIGSOFT Softw. Eng. Notes*, 41(6):1–6, January 2017. `doi:10.1145/3011286.3011301`.

4 Guanzhong Chen. Dmoj, 2020. accessed on 20 Jan 2020. URL: `https://dmoj.ca/`.

5 European Commission. Coding - the 21st century skill, 2018. accessed on 20 Jan 2020. URL: `https://ec.europa.eu/digital-single-market/en/coding-21st-century-skill`.

6 Jaap Eldering, Nicky Gerritsen, Keith Johnson, Thijs Kinkhorst, and Tobias Werth. Domjudge, 2020. accessed on 20 Jan 2020. URL: `https://www.domjudge.org/`.

7 FGPE Project Consortium. Framework for gamified programming education, 2018. accessed on 20 Jan 2020. URL: `http://fgpe.usz.edu.pl`.

8 Google. Code with google, 2020. accessed on 20 Jan 2020. URL: `https://edu.google.com/code-with-google/`.

9 Google. Google's coding competitions, 2020. accessed on 20 Jan 2020. URL: `https://codingcompetitions.withgoogle.com/`.

10 HackerRank. Hackerrank, 2011. accessed on 20 Jan 2020. URL: `https://www.hackerrank.com/`.

11 European Coding Initiative. All you need is code, 2020. accessed on 20 Jan 2020. URL: `http://www.allyouneediscode.eu`.

12 Kattis. Kattis, 2019. accessed on Jan 2020. URL: `https://open.kattis.com/`.

13 José Paulo Leal. Mooshak, 2018. accessed on Jan 2020. URL: `https://mooshak.dcc.fc.up.pt/`.

14 LeetCode. Leetcode, 2015. accessed on 20 Jan 2020. URL: `https://leetcode.com/`.

15 Ricardo Queirós and José Paulo Leal. PExIL: Programming exercises interoperability language. In *Conferência Nacional XATA: XML, aplicações e tecnologias associadas, 9. ^{a}*, pages 37–48. ESEIG, 2011.

16 Erik Scheffers, Tom Verhoeff, Stefan Geuns, Marijn Kruisselbrink, Paul Wagener, Robert Leenders, Iosif Macesanu, and Maikel Steneker. peach3, 2017. accessed on Jan 2020. URL: `https://peach3.nl`.

17 J. Swacha, R. Queirós, and J. C. Paiva. Towards a framework for gamified programming education. In *2019 International Symposium on Educational Technology (ISET)*, pages 144–149, July 2019. `doi:10.1109/ISET.2019.00038`.

18 Jakub Swacha. SIPE: A Domain-Specific Language for Specifying Interactive Programming Exercises. In *Towards a Synergistic Combination of Research and Practice in Software Engineering*, pages 15–29, Cham, Switzerland, 2018. Springer.

19 Jakub Swacha, Ricardo Queirós, José Carlos Paiva, and José Paulo Leal. Defining requirements for a gamified programming exercises format. *Procedia Computer Science*, 159:2502–2511, 2019. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019. `doi:10.1016/j.procs.2019.09.425`.

20 TopCoder. Topcoder, 2001. accessed on 20 Jan 2020. URL: `https://www.topcoder.com/`.

21 Maarten Vansteenkiste and Edward L. Deci. Competitively contingent rewards and intrinsic motivation: Can losers remain motivated? *Motivation and emotion*, 27(4):273–299, 2003. `doi:10.1023/A:1026259005264`.

22 Tom Verhoeff. Programming task packages: Peach exchange format. *International Journal Olympiads In Informatics*, 2:192–207, 2008.