Engaging learning to program using real data

Álvaro Nuno Santos
Department of Informatics Engineering
Coimbra Polytechnic - ISEC
Coimbra, Portugal
ans@isec.pt

Anabela Gomes
Department of Informatics Engineering
Coimbra Polytechnic - ISEC
Centre for Informatics and Systems
University of Coimbra
Coimbra, Portugal
anabela@isec.pt

António José Mendes
Centre for Informatics and Systems
Department of Informatics Engineering
University of Coimbra
Coimbra, Portugal
toze@dei.uc.pt

***Abstract* — This Innovative Practice Work in Progress presents an auxiliary tool that can be used in programming teaching in order to increase students' motivational levels and, hopefully, reduce the failure of related curricular units. Programming learning is neither simple nor a quick task to accomplish. On the contrary, learning to program requires a lot of effort, persistence and, above all, time for assimilation and training of all the related skills. Furthermore, students are usually exposed to amazing application interfaces and advanced application capabilities and, on the contrary, when they attend the first programming courses, they are asked to develop small programs, usually with no practical use and with black and white screen-based interfaces. Therefore, it would be surprising that students, who didn't previously have any motivation for programming, suddenly will get motivated to develop boring (traditional) programming exercises. In this context and underlining the idea that it is more motivating to work with genuine data, this work presents a software library that allows students to solve introductory programming exercises based on real data of their day-to-day life, like data obtained from their smartphones. The library is easily integrated into the development environments used by students, being a useful complementary tool to use in the classroom context.**

Keywords—Programming Learning, Real Data, Student Motivation

I. INTRODUCTION

The literature includes many references about the high failure and dropout rates in introductory programming courses in many high education institutions worldwide [1-4]. Many causes for the learning difficulties associated to these courses have already been identified [5-9]. It is possible to argue that difficulties are mainly related with the students' background knowledge, study methods and attitudes, the nature of programming, and the pedagogical strategies commonly used in introductory programming courses [10-11]. Different solutions have been proposed in literature, but the situation remains mostly unchanged, as many novices continue to struggle to learn basic programming. Research in this field includes efforts in several areas in order to engage students in the process. There are different pedagogical approaches that encourage an active role of the student, such as problem-based learning [12-14] or collaborative learning [15-16]. Nevertheless, the actual results obtained from the application of these methods are often disappointing. Although some students have success, it is normally a smaller number than expected.

II. DEMOTIVATION: FROM THE DREAMS TO THE NIGHTMARES

From our experience students usually begin their programming courses, apparently, motivated to work with a goal of getting a good mark. However, lesson by lesson, we notice that the number of students attending the classes decreases and the student motivation in the class also decreases. We also notice that, in class, some students are more motivated and participative while others keep quiet most of the time and simply wait for a solution. When confronting the quieter ones with direct questions, such as asking to suggest some improvement or alternative solution for the problem being discussed, often they are not able to make any contribution due to several misunderstandings concerning essential basic concepts or simply not having an idea about how to answer. Many of these students soon get lost for the course and they dropout or fail in the exams.

Many students have difficulty to understand the objective of each programming concept and how they may be useful to solve interesting and real problems. Moreover, they have to work with "complex applications" (development environments) with hundreds of options, that are difficult to install and configure, that will only allow them to create programs that appear in a black window (console) to show information that, for them, is useless. This generation is used to amazing applications with stunning interfaces, that they use in their day-to-day life, and, in the classes, they work with a set of unrelated numbers or other information to reach a result that is not useful and not interesting for them. Of course, as teachers, we have the perception that they don't work enough to overcome the difficulties associated with programming. But, a demotivated student will always be a predictable failure.

In the real world or in a practical sense, motivation is highly valued because it helps to produce positive results [17]. Motivation is a core factor in the learning-teaching process to improve active learning [18]. The literature shows a high diversity of terms and approaches about motivation [19]. In the literature we can find many strategies intended to motivate and engage students in programming learning. In [20] "...an intelligent assistant has been proposed to capture student's attention and to increase motivation and the time that students spend in learning programming and to make learning process more interesting". Some authors use robots to motivate and involve students in algorithm design, trying to overcome the

978-1-5386-1174-6/18/$31.00 ©2018 IEEE

student's lack of problem solving skills, allowing them to manipulate real entities, making the algorithm concrete instead of completely abstract [21-25]. Different types of game approaches have also been developed to engage and motivate students in the process of programming learning. Several authors discuss computer video games and game-like environments as one strategy to use as a motivational tool to engage students in learning programming [26-27]. Serious games are also being used to improve learning motivation [28]. In [29] is reported the increasing of students' motivation for learning the basics of programming by writing programs to manipulate the units of a real time strategy game (RTS)". A more recent work proposes program animations and automatic evaluation of programs to support students' motivation in programming courses [30].

III. REAL DATA FOR PROGRAMMING (RDP)

A. General Objectives

We believe that the exercises commonly used in introductory programming courses are not very motivating for most students. Student motivation could be improved if they develop programs that are interesting for them, instead of programs that simply manipulate numbers without a specific meaning. So, we thought that it might be interesting to use data that the students recognize and with which they identify themselves. The use of real data can also allow students to easily evaluate the program results, verifying in which situations they begin to diverge from what is expected. Nevertheless, this approach introduces a problem related to the time needed for the programs development and/or the time needed to collect and input test data for program validation. This may make the process too long, limiting the number of problems that can be developed in the class.

Bearing in mind this difficulty, we began to think about a system that might allow the use of real data in initial programming courses. This new system must be planned so that it wouldn't be prejudicial to the normal class plan. In our case, since the C language is used for the introductory courses, the solution to this problem could be to create a C library of simple functions that would allow any student to take advantage of its functionalities to access and use real data in his/her programs. In order to keep the library relatively simple, it was intended to start with data from just one source of information. Thus, it was necessary to identify a source of data that had real information, recognized by all the students, and with which they had some connection and interest. It was therefore decided to use social network data as the main source, more specifically one of the social networks most used by our students – Facebook.

At the beginning of the development of this project access to Facebook data was relatively simple, using "Facebook Login for devices" [31] and the "Facebook Graph API" [32], allowing access to "friends" user data, such as name, email and date of birth. However, since 2014, Facebook has been introducing security mechanisms which has made it unfeasible to use as a viable data source for our purposes. We considered using "Google Contacts API" [33] and "Google People API" [34] to access similar data, but the difficulties to access information from console applications were identical.

Thus, while maintaining the objective of using information related to student's contacts in the context of their programming exercises, a slightly different solution was chosen: the user consciously shares, for a limited time, information about the contacts on his/her smartphone to be used in one or more programs that she/he is developing. This solution will be described in the following sections.

B. Information sharing

To be useful, the solution must work in Android and iOS smartphones, as they are normally used by our students. This is not a problem, as APIs are available in both of them which allow access to contacts information that we consider minimal for use in introductory programming exercises, namely: name, birthday, phone number and email address. It should be stressed that some fields may not be available for all contacts. However, the system must be sufficiently robust and flexible to deal with any errors that may arise from the lack of information.

There was a need to define how contacts were sent to/received by the programs to be developed in the context of students' programming studies. To assist in this process, we needed to create a library, but it was necessary to define how this library could establish a direct connection with the mobile devices through the IP network. This solution, although functional, had some problems in its generalized use. The exchange of direct information with the mobile devices could fail due to connection problems. The connection between classroom computers and smartphones was not always allowed, due to restrictions of the local network or the mobile network operator of the mobile phone in question. As all mobile devices (smartphones) have access to the internet, we decided to use an external server. The external server solution had the problem of transmitting information through a third-party system (the server) that we intended to remain private to each user. To minimize threats, the information that is transferred to the final computer through the server is double-encrypted. The first encryption is done through a private key of the application and the second through a password chosen by the student. For this, when the user chooses to share the information of the existing contacts in his/her mobile phone, he/she has to indicate a password to protect the data. After gathering the data in JSON format [35], this information is encrypted using, first, a private key defined for the application. Once encrypted, the password provided by the user is used to encrypt it again. For security reasons the algorithm used for encryption is not explained here. A message with the encrypted data is then sent to the server, which associates a unique identifier and a time stamp before temporarily storing the information. The identifier is returned to the mobile application and presented to the user. According to the current settings the information is kept on the server for 3 hours (maximum time for a class in our institution). These definitions as well as the encryption mechanisms may be revised in future versions.

Some screenshots of the iOS app can be seen in Fig. 1. The Android version has a similar interface. In both applications, Android and iOS, permission to access contacts is asked the first time the application is used.

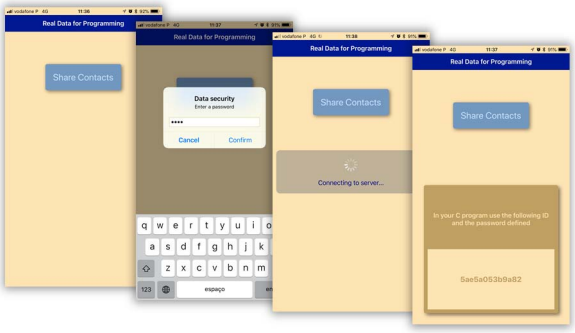

Fig. 1. Real Data for Programming iOS app

C. Library Usage

To use the information stored in the server, the proposed library must be used in the projects. After including the library, when the program is executed the student is asked for the unique identifier and the password entered in the mobile device. With this information a connection is established with the server to obtain the message with the encrypted information. Locally, the message is decrypted and will be ready for use according to the mechanisms explained in the next section. To avoid constant ID and password input and queries to the server, the information is temporarily stored locally.

The library is included in the projects similarly to the remaining C libraries, according to the settings for each development environment used. A header file with the prototypes of all available functions is also provided. Thus, in order for the library to be properly compiled and integrated into the projects, a `#include "rdp.h"` must be included in the program. To access the data shared by the mobile application is it necessary to start by calling the function `RDP_init(int reset)`. The parameter of this function can be 0 or 1 indicating, respectively, whether previously transferred data should be used or if a new process of downloading information should be started. If the user chooses to refresh such data by downloading new information, the user will be prompted for both the identifier displayed in the mobile application and the password defined. If the `RDP_init` function is not directly invoked, it will be called at the beginning of any other library function that is invoked.

To use the data, adequate functions are available to access each contact information. The contacts can be accessed through indexing or by an enumeration process. In Table I a summary of the library most relevant functions can be consulted. It is possible to verify that the same information can be accessed using different approaches (different functions), but this is intentional in order to allow their use in different contexts or needs in the classroom. In the next section several contexts in which these functions can be used will be presented.

TABLE I. RDP FUNCTIONS REFERENCE

`int RDP_init(int reset);`	Initializes Real Data Programming library. If *reset* is 0 the offline copy of the information taken from the server is used. If *reset* is 1 then the user is asked for the ID presented on the smartphone followed by the user password.
`int RDP_SetFilter(int filter);`	Activates a contact filter in order to use only a subset, based on the information that is available for each user. Can be: `RDP_FILTER_ALL`, `RDP_FILTER_NAME`, `RDP_FILTER_AGE`, `RDP_FILTER_PHONE`, `RDP_FILTER_EMAIL`.
`void RDP_PrintAllData();`	Prints all the JSON data received from the server
`void RDP_PrintProcData();`	Prints all the data processed that can be accessed by the RDP library
`int RDP_GetItemsCount();`	Returns the number of contacts available
`char *RDP_GetName(int index);`	Returns the name of the contact at the specified index
`int RDP_GetAge(int index);`	Returns the age of the contact at the specified index
`int RDP_GetBirthdayYear(int index);`	Returns the birth year of the contact at the specified index
`int RDP_GetBirthdayMonth(int index);`	Returns the birth month of the contact at the specified index
`int RDP_GetBirthdayDay(int index);`	Returns the birth day of the contact at the specified index
`int RDP_GetPhonesCount(int index);`	Returns the number of phone numbers of the contact at the specified index
`char *RDP_GetPhone(int index, int n);`	Returns the n^{th} phone number of the contact at the specified index
`int RDP_GetEmailsCount(int index);`	Returns the number of email addresses of the contact at the specified index
`char *RDP_GetEmail(int index, int n);`	Returns the n^{th} email address of the contact at the specified index
`void RDP_EnumBegin();`	Initializes (or reinitializes) the enumeration process among all filtered contacts
`int RDP_EnumNext();`	Goes to the next contact in the enumeration process
`int RDP_EnumPrev();`	Goes to the previous contact in the enumeration process
`int RDP_EnumLast();`	Goes to the last contact in the enumeration process
`int RDP_IsFirstPosition();`	Returns true (1) or false (0) if the current item is the first one or not
`int RDP_IsLastPosition();`	Returns true (1) or false (0) if the current item is the last one or not
`int RDP_isAfterEnd();`	Returns true (1) or false (0) if the enumeration process already ended or not
`int RDP_GetCurPosition();`	Returns the current position in the enumeration process
`char *RDP_GetCurName();`	Returns the name of the current contact in the enumeration process
`int RDP_GetCurAge();`	Returns the age of the current contact in the enumeration process
`int RDP_GetCurBirthdayYear();`	Returns the birth year of the current contact in the enumeration process
`int RDP_GetCurBirthdayMonth();`	Returns the birth month of the current contact in the enumeration process
`int RDP_GetCurBirthdayDay();`	Returns the birth day of the current contact in the enumeration process
`char *RDP_GetCurPhone();`	Returns the first phone of the current contact in the enumeration process
`char *RDP_GetCurEmail();`	Returns the first email of the current contact in the enumeration process

IV. RDP CLASSROOM USAGE

As mentioned, RDP can be used in different situations in the classroom context. RDP is not intended to replace the traditional exercises that each teacher plans for his/her classes, but as an additional method for programming skill training, consolidating programming knowledge and as a motivational tool in the study of programming. Their use in class may be planned, but it can also be used as a plan B for special situations that can arise in class. For example, students who finish their regular exercises more quickly or to try to reach students who don't show interest by regular exercises. The RDP can also be used to perform extra activities: homework or challenges given by the teacher throughout the course. Although RDP can be used from the early stages of programming learning, it will be more beneficial for students who already know the basics of using C-language functions. In order for the teacher to begin the use of RDP as soon as possible, special care has been taken not to include functions with pointer parameters. There are only a few functions that, given the nature of the data, need to return a character pointer (`char *`) in order to give access to the contact's name, email, or phone.

RDP can be used within the initial contents of introductory programming, such as sequential exercises or selections, but will gain more relevance for exercises where data sets are processed. This type of exercise has to use loop instructions and, as such, it is considered that the RDP should be used as soon as the concepts of loops have already been taught. RDP can appear as an auxiliary tool in solving knowledge consolidation exercises about loops. In this context there are several typical exercises, such as the calculation of averages, sums, maximum values, among others, which may become more interesting if student provided real data is used. For example, the following exercises may be proposed: find the oldest person of your contacts (maximum value); calculate the average age of your contacts; count the number of contacts whose ages are between 18 and 25. In the case of the first exercise, the solution to find the oldest age using RDP may be:

```
#include <stdio.h>
#include "rdp.h"

int main(int argc, char** argv) {
    int i,age,max;
    RDP_SetFilter(RDP_FILTER_AGE);
    max = RDP_GetAge(0);
    for(i=1;i<RDP_GetItemsCount();i++){
        age = RDP_GetAge(i);
        if (age > max)
            max = age;
    }
    printf("Max age: %d\n", max);
    return 0;
}
```

In the solution presented, some care was taken in order to filter the contacts, so that, only those with birthday/age information were used, and the algorithm was optimized initializing the variable *max* with the first known age (it was assumed that there is at least one contact with an associated age). There are several ways to solve each exercise, it is not intended that the RDP is a limiting factor of the preferred methods of each teacher or student.

With the introduction of the String concept we can also start to take advantage of other functions, such as those that give access to the name, phone number or email address. With these functions numerous exercises can be solved, such as: list the name and phone number of each of your contacts; find the phone number of a certain contact; display a list of contacts in the format: "*<family_name>, <first_name>: <phone_number>*". These are only some examples that are relatively simple, although traditionally they begin to highlight some students' basic difficulties. For example, in the last exercise, students usually have difficulty in processing, individually, the words of the full name to present them in the requested format.

When more advanced C programming topics start, RDP is a valuable tool that can be used more often, at least as a data source to test developed programs. When we start explaining *struct* concepts, dynamic array vectors or linked lists, there are lost periods in the classes with successive tests that require a time-consuming introduction of data. With RDP, we can solve static or dynamic structures exercises with automatic filling, taking advantage of the contacts of each student. Perhaps, more important than speeding up the data entry process is that students are working with data they recognize. This way, they can predict the expected outputs for the programs and from which they can derive their own benefits. For example, students can create contact lists with certain characteristics to use in their day-to-day lives.

V. CONCLUSIONS

As programming learning is a complex task, we believe that most people agree that it is impossible to overcome all the problems related to this subject. Several methods that can help to mitigate this problem have been tackled with different levels of success. Based on our experience and the belief of several authors that motivated students get better results, we presented in this paper a system which take advantage of user personal data as an auxiliary tool in the programming skills training. With this system, students can develop computer programs, that may help them in their studies in the diverse subtopics of programming, using and processing data that they know and understand. When using their own data, they can easily predict the results expected and can identify eventual mistakes in the developed programs.

In the actual version, only the information that the students have in their smartphone contact list is used, but we are planning to include other data sources in the system. The earlier version of this system used social network data (Facebook) but this was abandoned because of well-known problems that led to restrictions on their use. Nevertheless, we are considering taking advantage of other social networks that are being more often used by the students. With less concerns on privacy or ethics issues we want to include data of public sources with public information, like: sport results, weather conditions, geography data, EuroMillions numbers, among others.

Taking into consideration other learning programming environments different from ours, we want to make available in the near future adapted versions to be used in Python and Java.

Next school year we intend to perform some tests to prove the effectiveness of our system and the possibility of its generalized use in regular classes.

REFERENCES

[1] T. Jenkins, "On the difficulty of learning to program," in *Proceedings of the 3rd Annual LTSN_ICS Conference (Loughborough University, United Kingdom, August 27-29, 2002), The Higher Education Academy*, 2002, pp. 53-58.

[2] E. Lahtinen, K. Ala-Mutka, and H-M Järvinen, H-M., "A study of difficulties of novice programmers," in Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (Monte de Caparica, Portugal, June 27-29, 2005), ACM New York, NY, USA, 2005, pp. 4-18.

[3] R. Lister, B. Simon, E. Thompson, J. L. Whalley, and C. Prasad, "Not seeing the forest for the trees: novice programmers and the SOLO taxonomy," *SIGCSE Bulletin*, vol. 38, no. 3, pp. 118-122, 2006.

[4] C. Watson, and F. W. Li, "Failure Rates in Introductory Programming Revisited," in *Proceedings of the Innovation and Technology in Computer Science Education Conference (ITiCSE2014), Uppsala, Sweden, ACM*, 2014.

[5] P. Byrne, and G. Lyons, "The effect of student attributes on success in programming," *SIGCSE Bulletin*, vol. 33, no. 3, 49-52, 2001.

[6] I. Milne, and G. Rowe, "Difficulties in learning and teaching programming – views of students and tutors," *Education and Information Technologies*, vol. 7, no. 1, pp. 55–66, 2002.

[7] J. Bennedsen, and M. E. Caspersen, "Abstraction ability as an indicator of success for learning object-oriented programming?," *SIGCSE Bulletin inroads*, vol. 38, no. 2, pp. 39-43, 2006.

[8] W. D. Gray, N. C. Goldberg, and S. A. Byrnes, "Novices and programming: Merely a difficult subject (why?) or a means to mastering metacognitive skills*?," Journal of Educational Research on Computers*, vol. 9, no. 1, pp. 131-140, 2007.

[9] J. Stachel, D. Marghitu, T. Brahim, R. Sims, L. Reynolds, and V. Czelusniak, "Managing Cognitive Load in Introductory Programming Courses: A Cognitive Aware Scaffolding Tool," *Journal of Integrated Design and Process Science, Computer Science*, vol. 17, no. 1, pp. 37-54, 2013.

[10] A. Gomes, and A. Mendes, "Learning to program – difficulties and solutions," in *Proceedings of the International Conference on Engineering Education, Coimbra, Portugal*, 2007.

[11] A. Gomes, W. Ke, M. Marcelino, S. Lm, S. Siu, and A. Mendes, "A Teacher's view about introductory programming teaching and learning – Portuguese and Macanese perspectives," in *Proceedings of 45th ASEE/IEEE Frontiers in Education 2017 - FIE'17, Indianápolis, Indiana, USA,* 2017.

[12] A. Ellis, L. Carswell, A. Bernat, D. Deveaux, P. Frison, and V. Meisalo, "Resources, tools, and techniques for problem based learning in computing," in *Working group reports of the 3rd annual SIGCSE/SIGCUE conference on integrating technology into computer science education, New York, NY: ACM Press*, 1998, pp. 41–56.

[13] F. Dochy, M. Segers, and P. Van den Bossche, (2003). Effects of problem-based learning: A metaanalysis," *Learning and Instruction*, vol. 13, pp. 533–568, 2003.

[14] B. F. Jones, C. Rasmussen, and M. C. Moffitt, *Real-life problem solving: A collaborative approach to interdisciplinary learning*. Washington, DC: American Psychological Association, 1997.

[15] P. Dillenbourg, "What do you mean by collaborative learning?," in *Collaborative learning: Cognitive and computational approaches*, P. Dillenbourg, Ed. Oxford, UK: Elsevier, 1999, pp. 1-19.

[16] Harvard Business School, (2013). "Hardware Case Method." [Online]. Available: http://www.hbs.edu/mba/academic-experience/pages/the-hbs-case-method.aspx

[17] R. M. Ryan and E. L. Deci, "Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being," *American Psychologist*, vol. 55, pp. 68–78, 2000.

[18] P. Pintrich, "Motivation and classroom learning," in *Handbook of psychology: Educational psychology*, W. Reynolds and G. Miller, Ed. Hoboken, NJ: John Wiley & Sons, 2003, vol. 7, pp. 103–122.

[19] P. Murphy, and P. Alexander, P. "A motivated exploration of motivation terminology," *Contemporary Educational Psychology*, vol. 25, pp. 3–53, 2000.

[20] M. Konecki, N. Kadoic, and R. Piltaver, "Intelligent assistant for helping students to learn programming," in *Proceedings of the 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO),* 2015, pp. 924-928.

[21] J. Davis, B. Wellman, M. Anderson, and M. Raines, "Providing robotic experiences through objectbased programming (PREOP)," in *Proceedings of the 2009 Alice Symposium, Durham, North Carolina, USA: ACM New York, NY, USA*, 2009, pp. 1-5.

[22] T. Lauwers, I. Nourbakhsh, and E. Hamner, "CSbots: design and deployment of a robot designed for the CS1 classroom," in *Proceedings of the 40th ACM technical symposium on Computer Science Education (SIGCSE'09), Chattanooga, Tennessee, USA: ACM New York, NY, USA*, 2009.

[23] W. McWhorter, and B. O'Connor, "Do LEGO® Mindstorms® motivate students in CS1?," in *Proceedings of the 40th ACM technical symposium on Computer Science Education (SIGCSE'09) , Chattanooga, Tennessee, USA: ACM New York, NY, USA*, 2009, pp. 438-442.

[24] C. Martin, and J. Hughes, "Robot dance: edutainment of engaging learning," In *Proceedings of the 23rd Psychology of Programming Interest Group (PPIG'11), York, UK*, 2011.

[25] M. M. Mcgill, "Learning to Program with Personal Robots: Influences on'Student Motivation," *ACM Transactions on Computing Education*, vol. 12, no. 1, pp. 1–32, 2012.

[26] C. Kazimoglu, M. Kiernan, L. Bacon, and L. MacKinnon, L. "Understanding Computational Thinking before Programming: Developing Guidelines for the Design of Games to Learn Introductory Programming through Game-Play," *International Journal of Game-Based Learning*, vol. 1, no. 3, 2011, pp. 30-52.

[27] P. Molins-Ruano C. Sevilla, S. Santini, P. A. Haya, P. Rodriguéz, and G. M. Sacha, "Designing videogames to improve students' motivation," *Computers in Human Behavior*, vol. 31, no. 1, pp. 571–579, 2014.

[28] A. Vahldick, A. J. Mendes and M. J. Marcelino, "Analysing the Enjoyment of a Serious Game for Programming Learning With two Unrelated Higher Education Audiences", in *Proceedings of European Conference on Games Based Learning*, 2015.

[29] M. Muratet, E. Delozanne, P. Torguet, and F. Viallet, "Serious game and students' learning motivation: Effect of context using Prog&Play," in *Proceedings of the 11th International Conference on Intelligent Tutoring Systems, Lecture Notes in Computer Science*, 2012, pp. 123–128.

[30] P. Tavares, P. Henriques, and E. Gomes, "A Computer Platform to Increase Motivation in Programming Students – PEP," in *Proceedings of the 9th International Conference on Computer Supported Education (CSEDU 2017)*, 2017, vol. 1, pp. 284-291.

[31] Facebook, "Facebook Login for devices". [Online]. Available: https://developers.facebook.com/docs/facebook-login/for-devices

[32] Facebook, "Facebook Graph API." [Online]. Available: https://developers.facebook.com/docs/graph-api

[33] Google, "Google Contacts API." [Online]. Available: https://developers.google.com/google-apps/contacts

[34] Google, "Google People API." [Online]. Available: https://developers.google.com/people

[35] ECMA International. (2017). "Standard ECMA-404: The JSON Data Interchange Syntax." [Online]. Available: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf