CodeLab: Designing a Conversation-Based Educational Tool for Learning to Code

Enric Mor[1], Francesc Santanach[2](✉), Susanna Tesconi[1], and Carlos Casado[1]

[1] Computer Science, Multimedia and Telecommunication Studies, Universitat Oberta de Catalunya, Barcelona, Spain
{emor,stesconi,ccasadom}@uoc.edu

[2] eLearn Center, Universitat Oberta de Catalunya, Barcelona, Spain
fsantanach@uoc.edu

Abstract. This work presents the design and architecture of an educational tool for learning to code. The CodeLab tool is based on skill practice and assessment and is targeted for non-STEM students to develop computational thinking. The tool is designed to provide a lab experience and environment based on exercises to practice through a conversational interface.

Keywords: Conversation-based interaction · Dialogue-based interfaces Interaction design · Virtual learning environments · Learning to code Learning labs · Learning tools

1 Introduction

An educational lab is not only a repository of material and digital resources, but a multidisciplinary space. In this space, knowledge is built through to social interaction, self-management, self-training, informal research and learning, as well as participation in expanded educational communities. In this sense, the laboratory is the space for experimentation for students, as well as the place where teaching practices are exchanged and peer training processes are activated. The laboratory is an ideal place to develop creativity, a flexible space for testing and experimentation that allows learning from mistakes and allows the acquisition of learning by practicing skills and learning by doing [1].

Learning to code can be considered as one of the most representative examples of learning by doing [2]. Accordingly, a laboratory-based learning environment seems to be the place where learning in programming take place more easily, especially for students from non-STEM (Science, Technology, Engineering and Mathematics) contexts. Learning to code in a laboratory-based environment allows students to build code-related knowledge through practice and social interaction with peers and experts. Also, a laboratory-based setting can help teachers create scaffolding strategies in order to support the acquisition of code-related concepts and practice [3]. The creation of those scaffolding strategies in online, virtual and asynchronous learning environments presents several problems, among them the difficulty in recreating the complexity of social and cognitive interactions taking place in a laboratory-based setting.

C. Stephanidis (Ed.): HCII Posters 2018, CCIS 852, pp. 94–101, 2018.
https://doi.org/10.1007/978-3-319-92285-0_14

Research in TELE (Technology-Enhanced Learning Environments) [4] in order to remove those barriers is moving forward from GUI (Graphical User Interfaces) interfaces and WIMP (Windows, Icons, Mouse and Pointer) environments toward the development of new interface formats and novel interaction paradigms such as wearable and tangible interaction as well as bots and conversational interfaces.

The use of artificial intelligence in education constitutes a research area with extensive background. For this research project, it is particularly interesting to note the advances in intelligent tutoring systems (ITS) and autonomous agents [5]. In recent years, conversational systems or agents, also called bots or chatbots (referring to the conversation-based bot), have experienced a large growth. We can find voice assistants in the main mobile and desktop operating systems. Conversation-based assistants can also be found in intelligent speakers and commerce-based web pages. In addition, we can also find bots in specific chat spaces such as Facebook Messenger (https://es-es.messenger.com/) or Slack (https://slack.com/). The current research in technology enhanced learning suggests that the impact of chatbots in the educational field will lead to a significant change in learning tools. Chatbots can provide, for example, teacher support, student guidance or perform formative assessment [6].

This work presents the design and architecture of the CodeLab learning tool. CodeLab is an ongoing project that aims to offer students a learning code laboratory tool that simulates the interaction that is carried out in a face-to-face programming laboratory. Thus, the system mimics a human tutor in a physical lab. Through the communication interface, a chatbot suggests a series of exercises and activities that the students have to solve or complete. Also, it is available to students to help them solve questions and monitor their progress; and if it identifies learners experiencing difficulties, it proactively addresses to them practice and solve the proposed activities.

2 Teaching and Learning to Code

This project takes place at UOC, Universitat Oberta de Catalunya (http://www.uoc.edu), a fully online university based in Barcelona, Spain. UOC offers a wide range of educational programs, and some of them are STEM oriented. Currently, the programming courses of the STEM area are based on three main elements: (1) learning materials; (2) teacher-student interaction and; (3) practice-based activities.

Learning materials with different formats and goals are used in order to introduce and help students to understand the basic programming concepts and ideas. Text-based resources are used as main material for understanding basic programming concepts. Video contents are used in order to foster a better understanding of advanced programming features and interactive videos are used to show students how code is executed and how variables are changing value along the process.

The teacher-student interaction is mainly based mail-based forums where teachers answer questions and solve problems in a way somehow similar to a traditional face-to-face course. In the case of theoretical questions, the teacher provides an answer based on the learning materials. But in the scenario with a more practical code-related questions is the teacher providing answers by examining and executing the code written by students, fixing it, if needed, and by delivering written feedback to students.

Hands-on activities are carried out autonomously by the student. Depending on the programming language, instructions are offered in order to download and install the IDE (Integrated Development Environment), if it's available, or the editor and the compiler or the editor and the interpreter. During the course several activities are proposed to students with the aim to foster their self confidence in programming and the acquisition of basic programming concepts.

Student assessment takes place through an exam and a practice-based test. The exam is theory oriented and students must show some understanding of the programming concepts included in the course learning itinerary. In the practice-based test students have to prove programming skills. Both assignments are reviewed and graded by the teacher. In the case of the practice-based test the teacher has to download, review and execute the code of every student in order to provide feedback and a mark.

This approach presents several issues for student's learning as well as for teacher's monitoring and assessment: (1) for novice students installing an IDE can be a difficult task; (2) collaborative coding tasks can be difficult because they require to interchange code by email or to install and learn version control tools; (3) in order to review the code the teacher has to download all the activities of every student and execute them, and; (4) for the teacher it is difficult to track the student's learning process, review all the activities or getting information about students practice. To address these issues and improve the experience of learners and teachers, a new approach is needed. With the CodeLab research project we aim to solve some of these issues and improve the experience of teaching and learning to code through a practice-oriented learning environment with a conversation-based interface.

3 The CodeLab Learning Tool

The CodeLab approach focuses on learning to code in a laboratory learning environment. The research is also focused on the definition and design of the most appropriate interface and interaction style for providing chatbot-based support. This is an ongoing interdisciplinary research project that combines ethnographic research, human-computer interaction research and technology enhanced learning, with the aim of designing and developing chatbots for educational environments.

Starting from the idea that learning to program is not easy [7, 8] and above all, it requires a lot of practice, we envision a learning environment that could foster the understanding of programming concepts as well as the acquisition of programming skills. The process we envision is quite simple: the student is provided with some basic concepts, then, he's invited to practice as much as possible in order to assimilate them correctly. Once the concepts are acquired, he can move on to more advanced ones. Whatever the teaching system and approach, this process is not easy to implement. It requires the teacher to stop and verify that students have acquired sufficient fluency with the concepts and lessons presented before progressing with new ones. In distance learning, the student is quite self-sufficient in determining both the number of hours he or she devotes to study and the times when he or she does so. This complicates the work of the teacher who cannot set a previously defined learning itinerary. Therefore, the teacher is only able to control the assimilation of concepts with the tests and assignments that are carried out throughout the course.

The main functionalities of the CodeLab learning tool are: (1) learner can write code and visualize its execution; (2) activities for practice and learn are presented in a consistent and logic order; (3) students can solve, visualize execution and save activities; (4) the tool can monitor the progress of learners as they practice and solve activities; (5) teachers can follow up on the student's activity and performance and, if necessary, step in and provide recommendations or feedback; (6) a dialogue-based interface allows learners and teachers to interact in order to answer questions and share comments; (7) in the conversational interface, a chatbot behaves as a participant in the laboratory, offering answers to questions or helping in specific aspects of algorithmic learning or programming language.

Currently, a number of programming tools to write and execute code can be found. There are both open source and paid tools that are being used in online educational settings where learners can write code, debug it and visualize its execution in real time. Nevertheless, most of these tools are coding tools but not learning tools. That is, they do not provide learning scaffolding for students. The aim of the CodeLab tool is to provide each student a collection of learning activities to be solved within the tool. Thus, the student can access to the theory and also to a set of activities designed to practice the key concepts of programming. The activities are presented in form of a sequence that is related to a previously defined learning itinerary. The development of activity-based learning itineraries are part of the teacher's educational work and it takes into account the skills to be developed on each learning stage. Learning itineraries can have different types of activities: predefined or template-based, that can automatically generate instances. The template-based activities allow CodeLab to provide different activities for each learner where the same learning concepts can be practiced.

CodeLab will track the programming practice carried out by each student as well as the activities that he or she solves, recording the time spent on the practice and the number of activities completed. This will allow both the system and the teacher to follow up on each student, see their progress and make relevant recommendations. Activity monitoring must also allow learners to deliver exercises directly from the CodeLab to perform a training evaluation. For example, a submission could take the form of solving a series of activities in a given time-period. Once the assessment period ends or the learner finishes, the teacher can review the work and, if necessary, provide a mark. This allows to advance towards a formative evaluation approach, especially indicated in areas as programming, that is based on practice and incremental learning.

Based on the monitoring and analysis of the activity carried out by learners (solving exercises, interaction with the environment, etc.), the CodeLab tool will provide feedback based on the learning itineraries and recommendations around activities and exercises to be solved. It will also recommend reading educational content if a learner needs to review basic concepts of programming. These recommendations will be similar to the ones provided by an intelligent tutoring system, but here will presented taking advantage of the dialogue-based interface mentioned above.

One of the design challenges of CodeLab is the design of a conversation-based interface that allow conversations to take place in the context where learners practice code. The goal is to facilitate conversations as they would happen in a face-to-face laboratory: when a student has a question, she asks the teacher or a colleague while she is able to point where the problem is, in a coding activity. Additionally, with the CodeLab project we are exploring the introduction of educational bots in the conversation-based interface of the laboratory. The interaction design of the chatbots is another design challenge that is being addressed. It takes into account ethnographic studies [9] of current programming labs and the state of the art from the HCI field [6]. We identified a set of functionalities that bots can support, from simply providing help with the programming language syntax to a more complex like providing feedback based on the monitoring and analysis of the activity carried out by learners (solving exercises, interaction with the environment, etc.). Chatbots can provide feedback based on the learning itineraries and recommendations around activities and exercises to be solved. They can also recommend learners to read educational content in the case they need to review basic concepts. Recommendations will be presented through the dialogue-based interface and chatbots will behave as any other participant in the learning laboratory.

Therefore, with the CodeLab project, we are facing several research and design challenges such as: (a) the conversational interaction within the tool; (b) chatbot design and training; (c) workspace for code practice and visualizing; (d) learning activities and itinerary and progress design; (e) formative assessment approach; and (f) dashboards and data for the instructors.

4 Architecture

CodeLab is a technology-based learning tool that can be used in both blended and online educational settings. In both cases, the learning tool needs to be integrated in a virtual learning environment where students can also interact with other courses and services. From the learner point of view, the dialogue-based interface of CodeLab needs to provide an integrated interaction experience with the different elements of the virtual learning environment. Therefore, the challenge here is to design and deploy the chatbot conversation-based interaction into the main application contexts of the virtual learning environment. In our case, a fully online higher education institution, we identified three main contexts: (a) learning materials or digital learning resources context; (b) virtual classroom context; and (c) communication interface.

In the digital learning resources context (a), the chatbot is deployed inside the learning materials, acting as a learner assistant, providing, depending on the topic or objective of each material section, the most appropriate exercises, activities and assessments. In the virtual classroom context (b), the chatbot is deployed inside the virtual classroom environment, it acts as an assistant, providing, through the chat interface, access to the resources and assessments available. In the communication interface context (c), the chatbot is deployed using Slack as communication environment. Slack is a communication tool that can act as an external space that unifies all learning and teaching communications in one place. In the case of CodeLab, there is a Slack that brings all code learners together in the same place, there are channels for each programming language and for promoting the community engagement. Here, the chatbot acts as an automatic lab assistant, solving the students' code questions, providing the appropriate lab resources for solving activities, providing assessments and transferring to the real lab instructor those questions that it doesn't understand or needs to be answered by a human teacher.

The chatbot is also an applications/systems aggregator and each of the application/ system in that ecosystem has also an interface with several design challenges that need to be faced. The number and distribution of that systems or applications could change depending on the products to integrate, but from the logical point of view, we can distinguish the following modules: (a) workspace for code practice; (b) assessment management system; (c) grading & feedback; (d) dashboards & analytics; and (e) the chatbot itself (Fig. 1).

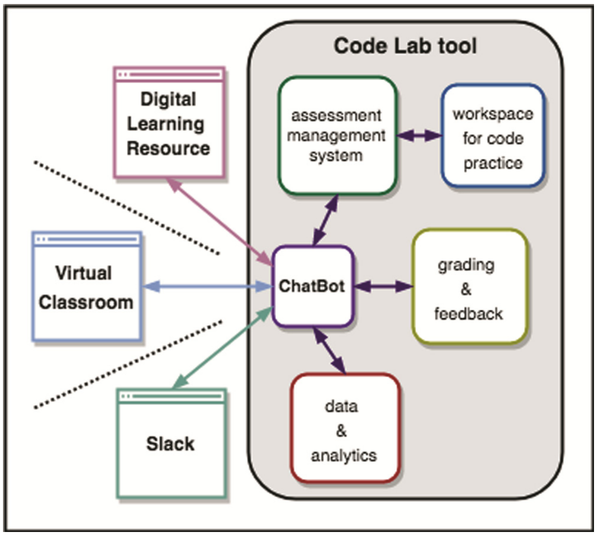
Code Lab tool
Digital Learning Resource
assessment management system
workspace for code practice
Virtual Classroom
ChatBot
grading & feedback
Slack
data & analytics

Fig. 1. CodeLab architecture.

The workspace for code practice (a) is essentially a cloud-base IDE for learning programming. In its first iteration, it provides P5.js as a programming language to learn. Since it is based on JavaScript, it does not require a server to compile and execute the code but can be directly interpreted and executed in any web browser.

The Assessment management system (b), is a system for managing the authoring and the deployment of the code exercises into the workspace for code. Through that module, is possible to connect with the authoring tools - like the workspace for code practice - and to set up specific learning activities for both, to be performed by students in specific time and to catalog for subsequent reuse (Fig. 2).

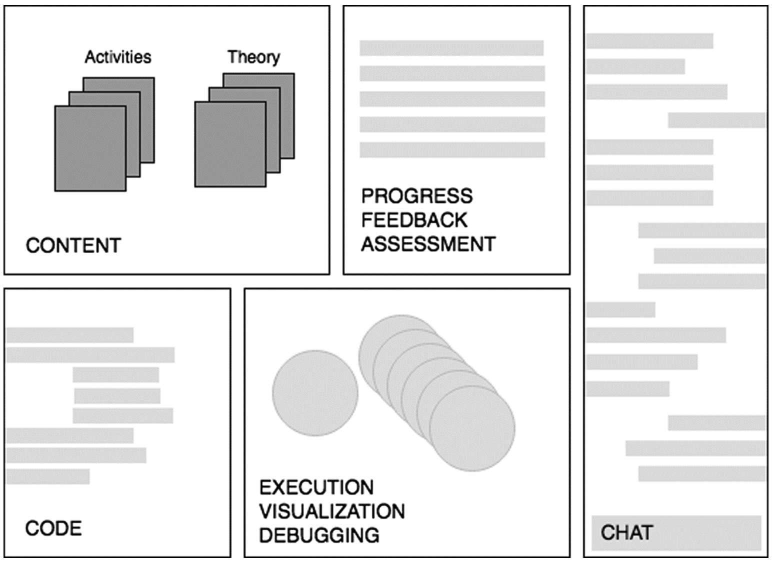

Fig. 2. CodeLab workspace

The grading & feedback (c), is a module that includes all the interfaces and services for grading and providing feedback to students. This is a complex issue, not only for the interaction with the chatbot and the other tools but also for the complexity of the grading process. It needs to provide a way to define learning itineraries and rubrics that could be very diverse such as the code style, the test cases, the legibility, the accuracy, the performance, and others.

The data & analytics module (d), includes interfaces and services related to the data gathered by the chatbot and by the other tools and modules in the ecosystem. It allows the chatbot provide recommendations for learners based on learning itineraries. This module also provides teachers with a dashboard to understand student progress.

The chatbot (e) is itself a module or application that provides dialogue-based interaction for students and instructors. For instructors, the chatbot provides access to modules above for the authoring, delivering, grading and monitoring processes. For students, it provides assistance in context, recommending the most appropriate exercises to them. An important aspect here is the dialog-based interface language processing component. During the project we plan to evaluate different levels of chatbot communication capacities, from a simple structured language to a natural language processing approach with the goal of measuring the importance of the human language proximity variable.

5 Discussion

With the CodeLab project, our goal is to provide a different approach to learn programming online, solve some common problems and provide a better learning experience through practice laboratories, learning by doing and conversation-based interactions. Therefore, we have two main goals: setting the architecture and design for the code learning tools and designing the chatbot that provides support to the tool. Introducing chatbots in online educational settings can be very challenging. In online learning the communication between teachers and students is a key element of its success and chatbots provide interesting opportunities to guide and support students. Also, chatbot deployment in virtual learning environments implies that different modules should be redesigned in order to introduce conversational interaction.

With our design and architecture, we are in the process of setting up a first version of the CodeLab learning tool. This tool will be evaluated with arts and design students in a P5.js course. Usually, non-STEM learners need more support and guidance when learning to code and we expect the CodeLab tool can provide it. From this pilot, we expect to get rich information to iterate the design of the learning environment and also the chatbot design in terms of guidance capabilities and its conversation design.

References

1. Romero, M., Lepage, A., Lille, B.: Computational thinking development through creative programming in higher education. Int. J. Educ. Technol. High. Educ. **14**(1), 42 (2017). https://doi.org/10.1186/s41239-017-0080-z
2. Hassinen, M., Mäyrä, H.: Learning programming by programming: a case study. In: Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006, pp. 117–119. ACM, February 2006. https://doi.org/10.1145/1315803.1315824
3. Van Merrienboer, J.J., Paas, F.G.: Automation and schema acquisition in learning elementary computer programming: implications for the design of practice. Comput. Hum. Behav. **6**(3), 273–289 (1990). https://doi.org/10.1016/0747-5632(90)90023-A
4. Veletsianos, G., Heller, R., Overmyer, S., Procter, M.: Conversational agents in virtual worlds: bridging disciplines. Br. J. Educ. Technol. **41**(1), 123–140 (2010). https://doi.org/10.1111/j.1467-8535.2009.01027.x
5. Roll, I., Wylie, R.: Evolution and revolution in artificial intelligence in education. Int. J. Artif. Intell. Educ. **26**(2), 582–599 (2016). https://doi.org/10.1007/s40593-016-0110-3
6. Song, D., Oh, E.Y., Rice, M.: Interacting with a conversational agent system for educational purposes in online courses. In: 2017 10th International Conference on Human System Interactions (HSI), pp. 78–82. IEEE (2017). https://doi.org/10.1109/HSI.2017.8005002
7. Gomes, A., Mendes, A.J.: Learning to program-difficulties and solutions. In: International Conference on Engineering Education–ICEE, vol. 2007 (2007)
8. Lahtinen, E., Ala-Mutka, K., Järvinen, H.M.: A study of the difficulties of novice programmers. ACM SIGCSE Bull. **37**(3), 14–18 (2005). https://doi.org/10.1145/1067445.1067453
9. Fink, R.D., Weyer, J.: Interaction of human actors and non-human agents. A sociological simulation model of hybrid systems. Sci. Technol. Innov. Stud. **10**(1), 47–64 (2014)