# Comilona: the game of balanced diet menus

Approaching Nutrition and Computational Thinking

Rafael Herrero, Coromoto León
Dpto. Ingeniería Informática y de Sistemas
Universidad de La Laguna
Apto. 456, 38200 La Laguna, Tenerife, Spain
comthink@ull.es, cleon@ull.es

***Abstract*—This work describes the development and implementation of the Comilona game. It has been crafted from the beginning trying to emphasize the educational content and the players enjoyment at the same time. The users can practice and develop their skills in Computational Thinking with little or no programming knowledge. The game target audience is 8 to 12 years and it aims is cover programming concepts through the development of menus to a balance diet. Also, it discusses the potential benefits of this kind of games as a support tool to foster student motivation and abilities in problem solving.**

***Keywords—computational thinking; game based learning; introductory programming, games and learning***

## I. Introduction

Computational Thinking (CT) has increasingly gained attention in the educational field in the past decade, following a short paper by J. Wing [1], who used this expression to indicate thinking as a computer scientist (i.e., using an analytic and algorithmic approach to formulate, analyze and solve problems). In her article, Wing claimed that CT is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child analytical ability. Ten years after this seminal work, hundreds of papers on this topic have been published in the academic literature with the added value of CT to foster thinking and problem solving skills [2].

A mutual influence between CT and coding/programming is recognized. The acquisition of CT does not to necessarily need computer programming, being a (conceptual) approach to problem solving that uses strategies such as algorithms, abstraction and debugging [3]. However, programming illustrates in concrete terms otherwise-abstract concepts and can therefore be an effective and practical way to foster the development of CT skills. On the other hand, teaching word processing and how to create slideshows does not lead students to do a deep analysis that allows them to think creatively and critically [4].

Digital Literacy, which is usually identified with the Information and Communication Technology (ICT) school subject, in which students learn to use computers and other electronic equipment to store and send information, is seen to differ from CT, even though connected to it. Few question the potential value of CT in schools but it is less clear how traditional coding tools can be successfully employed for Computer Science Education without introducing difficult overhead. Particularly in subject areas where coding is not the focus, coding overhead quickly turns into an insurmountable educational obstacle.

Moreover, many initiatives have been launched to promote programming among the population, especially among children and young people who have a broad digital culture. The Hour of Code [5] is a global initiative consisting of one-hour introduction to computer science. It was designed to demystify code and show that everyone can learn the basics. The goal is not to teach everybody to become an expert computer scientist in one hour. Only one hour is enough to learn that computer science is fun and creative, that it is accessible at all ages, for all students, regardless of their background. Similar initiatives are: Made With Code, Code Club, CoderDojo, Code Week, among others. Computer Science for All [6] is a project promoted by the White House which intends to empower a generation of American students with the computer science skills they need to thrive in a digital economy. Google CS First [7] is a project which is intended to inspire kids to create with technology through free computer science clubs. Google is also promoting computational thinking by the creation and dissemination of materials and courses for educators [8].

There is an extensive bibliography related to Game-Based Learning (GBL) for computing. Games can be a valuable tool for enriching computer science education, since they can facilitate a number of conditions that promote learning: student motivation, active learning, adaptability, collaboration, and simulation. Additionally, they provide the instructor the ability to collect learning metrics with relative ease [9]. In this work we present the digital game “Comilona” designed to teach knowledge, skills, and/or attitudes related to computer science. The systematic review by Wang [10] found no empirically validated games to support education in health care training. We found three games focus primarily on nutrition area [11]– [13] but not in computer science education.

The paper is organized as follows: Section II presents the progress of the digital game through examples. Section III locates the target audience. Section IV describes the implementation process and the software-tools used to develop the application. Section V presents the concluding remarks and the future works.

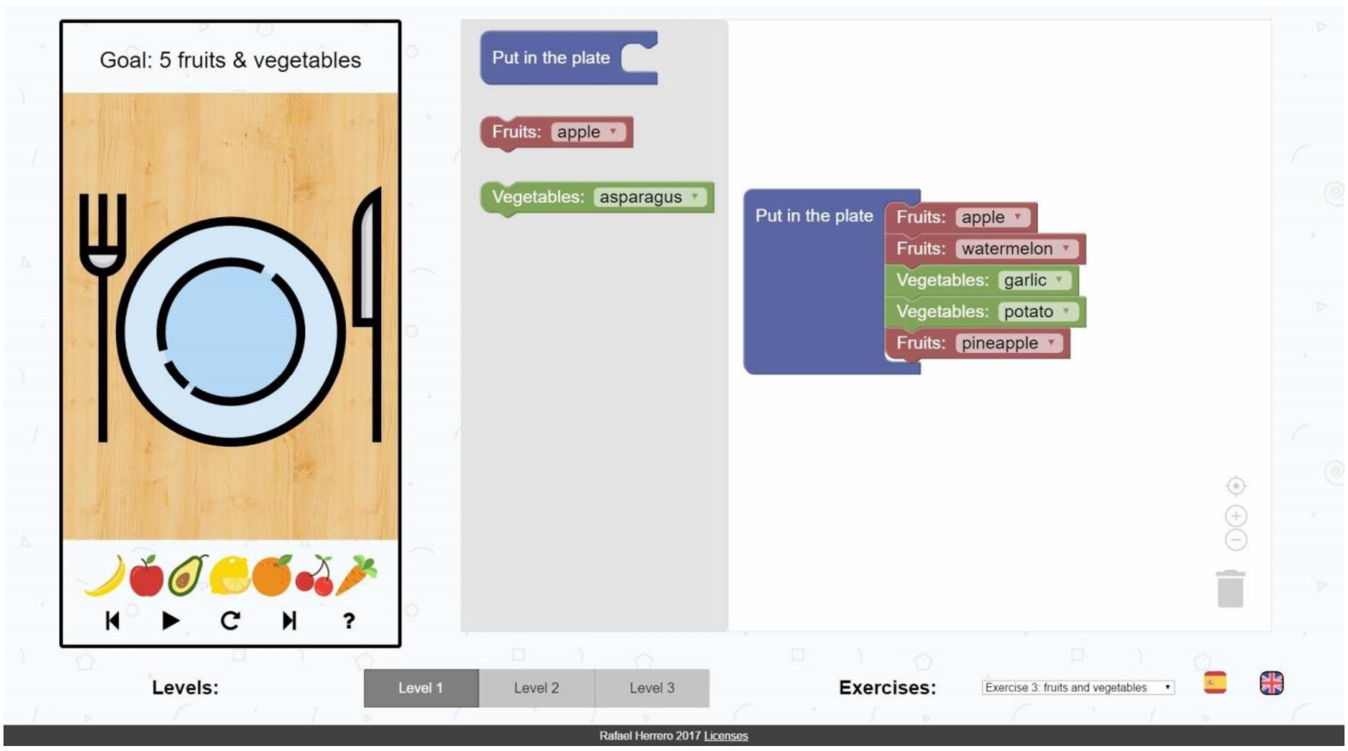

Fig. 1. Current version of Comilona prototype

## II. Game Development

The angular theme of the game is nutrition, where the goal is always to create a balanced menu that meets the nutritional values recommended by nutritionists [14]. However, the objective is to learn the concepts of CT in a gradual, never exponential way. The game is divided into levels with several challenges each.

The game simulates a puzzle where players control different kind of foods by giving various commands to them. The goal of the game is to prepare a dish where each level contains only one recipe. During the game-play, players need to design a solution algorithm through using programming and symbolic representations in order to mix the foods to cook the recipe. The commands players can give to the foods are divided into two as action commands and programming commands. While action commands are used to choose and move the foods in an environment consisting of a bowl, programming commands indirectly affect these actions and facilitate designing algorithms. Both types of commands are dragged from their associated toolbars and can be dropped into the programming specific area. Players play the game by dragging and dropping any number of commands into these programming area in any sequence they choose, for as long as they have foods in their game-play. To complete a level, players need to move their foods on the bowl, activate the blender, and this will then allow them to proceed to the next level. As players progress through the levels, the bowl environment expands and the recipes increases in complexity. In each level, players also encounter items that can be used to cook. These cookware items are randomly scattered every time players start to play a level, and thus this kind of approach ensures that the problem presented to a player at one level is significantly different from a problem presented to another player playing at the same level, or indeed the same player repeating the level to consolidate their learning. The randomness of cookware items is also controlled in order to guarantee the complexity of levels remains consistent. The game rewards players with new features (such as new cookware items, new recipes and new diet constraint) as players advance through the game.

Figure 1 shows the user interface for the first level where the user will learn simple concepts such as aggregation, which leads to the theory of sequence. For example, how many fruits are needed to reach a recommended nutritional value as fats or proteins. In the next level, it will approach more advanced concepts like loops. The first challenges are the same as the first level, but now, the player can not, for example, select fives apples one by one, they have to do it using loops. The last level will be a mix of all the concepts studied previously, but now going further in the programming concepts with conditionals and functions. In this case, we will point out the user if the result is not the optimal answer.

## III. Target Audience

This introductory computer science digital game was designed for ages 8 to 12, and might be useful for a wide range of ages and a variety of courses.

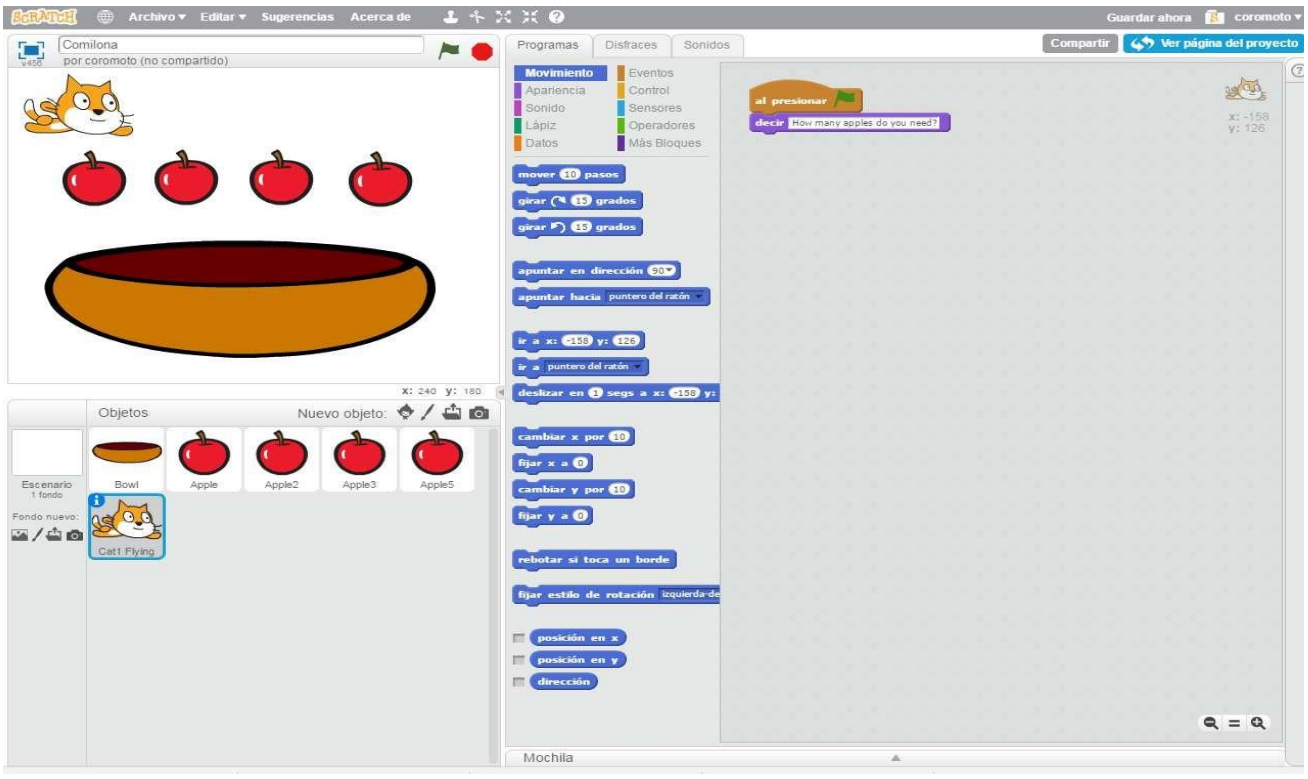

Fig. 2. Comilona story on Scratch

## IV. Implementation

A stand-alone, cross-platform desktop application has been developed. It works on the major operating systems on the market and is totally portable, so it can be copied to any storage device, such as a pendrive or a CD, without installing anything on the guest machine. No Internet connection is needed, but the development was based on web technologies. A client has been programmed in the Javascript programming language [15]. The native application was created using the Electron [16] framework. It is an open source project maintained by Github [17] and uses Chromium to assure that the development will be fully compatible with most browsers. Since Comilona is based on the block philosophy [18], the library for building visual programming editors Blockly [19] has been used. Blockly allows to define a workspace within the page where we will have the blocks on one side and the area on which to place them on the other. Each of the blocks represents a code fragment, we can create some custom ones or use the already predefined ones. These elements can be dragged from one area to another to form a puzzle that becomes code in different languages. The Comilona code is in charge of analyses the generated code to determine if the puzzle formed by the player is correct or not.

Comilona is not a platform to develop games like Scratch [20] or Kodu [21]. Scratch is a visual programming environment that lets users create interactive, media-rich projects. People have created a wide range of projects with Scratch, including animated stories, games, online news shows, book reports, greeting cards, music videos, science projects, tutorials, simulations, etc. [22]. Scratch builds on the constructionist ideas of Logo [23] and Etoys [24]. It makes easy to import or create many kinds of media (images, sounds, music). The Scratch Web site provides a social context for Scratch users, allowing users to share their Scratch projects, receive feedback and encouragement from their peers, and learn from the projects of others. On the contrary, Comilona is a stand-alone desktop application. The goal of Comilona is to offer a close exercise to develop CT skills for student for ages 8 to 12, and its story implementation using Scratch is out of this range. Figure 2 shows a possible implementation of Comilona's story using Scratch.

## V. Conclusions

This work has illustrated an approach to learn through digital games in an effort to integrate computational thinking with learning of nutrition concepts. The prototype game has now reached the stage where a detailed and structured evaluation can be carried out. A structured evaluation exercise with young students will be performed, and the empirical evidence from that exercise will be analyzed and used to validate our research.

## Acknowledgment

This work has been supported by the Spanish Ministry of Economy, Industry and Competitiveness inside the program I+D+i Orientada a los Retos de la Sociedad with contract number TIN2016-78410-R.

## References

[1] J. M. Wing, “Computational thinking,” Commun. ACM, vol. 49, no. 3, pp. 33–35, Mar. 2006.

[2] S. Bocconi, A. Chioccariello, G. Dettori, A. Ferrari, K. Engelhardt, P. Kampylis, and Y. Punie, “Exploring the field of computational thinking as a 21st century skill,” in EDULEARN16 Proceedings, ser. 8th International Conference on Education and New Learning Technologies. IATED, 4-6 July, 2016 2016, pp. 4725–4733.

[3] T. Bell, J. Alexander, I. Freeman, and M. Grimley, “Computer science unplugged: School students doing real computing without computers,” The New Zealand Journal of Applied Computing and Information Technology, vol. 13, no. 1, pp. 20–29, 2009.

[4] A. Collins and J. Brown, What’s Worth Teaching?: Rethinking Curriculum in the Age of Technology, ser. Technology, Education–Connections (the TEC Series) Series. Teachers College Press, Teachers College, Columbia University, 2017.

[5] “Code.org,” http://code.org, accessed: 2017-04-27.

[6] “Computer Science For All,” https://www.whitehouse.gov/blog/2016/01/30/computer-science-all, accessed: 2017-04-27.

[7] “Google CS First,” https://www.cs-first.com/, accessed: 2017-04-27.

[8] “Google for Education,” www.google.com/edu/resources/programs/exploring-computational-thinking/, accessed: 2017-04-27.

[9] C. Johnson, M. McGill, D. Bouchard, M. K. Bradshaw, V. A. Bucheli, L. D. Merkle, M. J. Scott, Z. Sweedyk, J. ´Angel, Z. Xiao, and M. Zhang, “Game development for computer science education,” in Proceedings of the 2016 ITiCSE Working Group Reports, ser. ITiCSE ’16. New York, NY, USA: ACM, 2016, pp. 23–44.

[10] R. Wang, S. DeMaria Jr, A. Goldberg, and D. Katz, “A systematic review of serious games in training health care professionals,” Simulation in Healthcare, vol. 11, no. 1, pp. 41–51, 2016.

[11] V. J. Green, R. B. Parnes, L. M. Montuori, T. Mardigan, J. Gerald, and D. Friedman, “Fresh minds, from farm to classroom: A nutrition and agriculture game,” Journal of nutrition education and behavior, vol. 35, no. 5, pp. 271–272, 2003. [12]

[12] J. M. Lacey, J. R. McGinn, R. L. Albino, and T. Ferro, “The nutritional scattergories R game: adding zest to a nutrition course,” Journal of nutrition education and behavior, vol. 35, no. 6, pp. 333–334, 2003.

[13] M. T. Burns, M. Benoit, and D. Bulvan, “Nutrition jeopardy,” Journal of nutrition education and behavior, vol. 34, no. 2, pp. 117–118, 2002.

[14] K. E. Drummond and L. M. Brefere, Nutrition for foodservice and culinary professionals. J. Wiley, 2004.

[15] C. Wootton, JavaScript Programmer’s Reference. Wrox Press Ltd., 2001.

[16] “Electron,” electron.atom.io, accessed: 2017-04-27.

[17] “Github,” github.com, accessed: 2017-04-27.

[18] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, “The scratch programming language and environment,” ACM Transactions on Computing Education (TOCE), vol. 10, no. 4, p. 16, 2010.

[19] “Blockly,” developers.google.com/blockly/, accessed: 2017-04-27.

[20] “Scratch,” https://scratch.mit.edu/, accessed: 2017-04-27.

[21] “Kodu,” https://www.kodugamelab.com/, accessed: 2017-04-27.

[22] J. Maloney, et al. "The scratch programming language and environment." ACM Transactions on Computing Education (TOCE), vol. 10, no. 4, pg. 16. 2010.

[23] S. Papert. Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc. 1980.

[24] C.J. Bouras, et. Al. “Squeak Etoys: Interactive and Collaborative Learning Environments” in I. Management Association (Ed.), Gaming and Simulations: Concepts, Methodologies, Tools and Applications, pp. 898-909, 2011