HOW TO TEACH PROGRAMMING WITH ELECTRONIC GAMES

Ronald Carvalho Boadana, Cristina Souza de Araujo, Jucimar Maia da Silva Junior,
Clairon Lima Pinheiro and Luis Cuevas Rodriguez
Escola Superior de Tecnologia - EST
Universidade do Estado do Amazonas - UEA, Manaus, Brazil

ABSTRACT

This article describes the process of development of electronic games using the programming language Python which aims to focus on upgrading the student's programming skills. This way to learn turns out to be interesting by the fact that using arcade games that are daily plenty popular among the young who dedicate a lot of their time to playing. The process of game creation is more exciting to apply development tools than programming lists by college subjects.

KEYWORDS

Programming, Games, Skills, Learning

1. INTRODUCTION

Programming is a task which it requires much time and dedication to be learned correctly [Nilo, 2019]. At the university, there's specific programming's subjects which are very important at development of course, besides being present in all engineering courses. It is complete natural that the students ask themselves about the benefits with studies of programming languages concepts [Sebesta, 2011].

Electronic games can be categorized as the first touch with a computer of a young person that can be with PCs, consoles and smartphones, besides also being entertainment software, which focus on holding attention for a long period of time. Many players have the wish of learning how to create a game and especially, how to create their own games in many times they thought that their game ideas are better than those of most commercial gamer's designers [Schell, 2015]. When it comes to learning programming, creating games can be a highly effective and engaging way to learn. Games are interactive, fun, and can help learners to understand the logic behind programming in a practical and engaging way [Sweigart, 2015].

2. OBJECTIVES

The programming language Python is very interesting as the first language programming which a person has touch by its simplicity and clarity [Nilo, 2019]. Writing a clean and organized code makes the code better to understand and keep [McConnell, 2004]. Here there is some of main programming good practices:

- Use descriptive names for variables, functions, and classes. This helps make the code easier to understand [Martin, 2008];
- Avoid code duplication. This helps reduce complexity and makes the code easier to maintain [Fowler, 2018];
- Use a version control system to track changes made to the source code [Chacon et.al, 2014];
- Write automated tests to ensure that your code is working correctly [Beck, 2002].

Software engineering is a branch which works with software development of several kinds and having good practices is very essential for a future professional. On game development, there is good practices that must be followed as well as any development environment. According to Sommerville, rigorous software testing and

iterative and incremental software development are important practices which must be adopted by developers because it can make the software development easier and adaptable to handle difficulties what can happen.

According to Martin, there is some techniques and practices which can used to improve the teammate is collaboration as an agile software development such as Scrum or Kanban. Also adopting a coding standard can be good for the project. Knowing how to handle with conflicts that can be emerge or problems among teammates is important, because at harmonic environment there's a higher level of enjoying, satisfaction and productivity increases.

3. JUSTIFICATION

Programming teaching at universities is widely used on graduation courses which involves technology like all engineering areas, computer science and information systems for application development is projects. Also, students cannot see how it relates to their interests or personal goals, because must be a way to make coding relevant and interesting by connecting it with their interests [Payne, 2015]. It is easier to notice the student's difficulties at high dropout rates at graduation courses previously mentioned.

In Brazil, the biggest evasion rate is realized on courses related to technology areas [G1, 2012], being that for each 4 students enrolled only one gets the graduation. This is a big problem, because there are always missing professionals to meet the market demand, a lot of jobs are created and some of them are filled.

The education must be unbounded of standard thought that is still the traditional axis of schools [Shaffer, 2006-a]. So, the electronic game development as a learning process looks ideal. Besides this strategy, also can highlight the active teaching methodologies which are very used in projects as the way to fix the learning.

4. METHODOLOGY

During the game development is necessary to use the active teaching methodologies as a form to get the enough learning to solve complex problems and the possibility to work with a team [Bacich el. al, 2018].

This section describes the materials and methods used during the project development:

- Python: During this process, Python was adopted because the students had familiarity with them and there's graphics libraries to develop.
- Turtle Graphics: This graphics library Python is used to draw on screen using simple commands. It was used for the first game, Pong, by its simplicity.
- PyGame: Python library specific to game development, when compared to Turtle with its functionality and commands, PyGame becomes very interesting by its quantities of commands which makes the code more simple and more functional. It was used in all games mentioned earlier.
- Linux: It's an operational system very used by programmers and made for them. It was used by students to get more immersion on programming and, it's a standard operating system used at universities to teach programming.
- Git: Also known as the version control system which helps the programmer to manage the code of a project. Is very used by programmers to team software development and keeping the code available to everyone from any computer.
- PyCharm: It's an IDE (Integrated Development Environment), so, a development integrated environment that assembles several development tools in one single program. It was developed by JetBrains, and it is very used by students for game development.

The actives methodologies are other pedagogy way which put the focus of teaching process and learning learner involving by discovery learning, investigation or problem is solving [Bacich el.al, 2018]. Learning by Doing is an active methodology, applied to the project, which focuses on hands-on projects, coding challenges, and interactives simulations as this way a programming educator can help the students to develop skills and knowledge they need [Kolb, 2014].

5. DEVELOPMENT

This section will explain how the development process for the game chosen and how learning techniques have applied.

5.1 Pong

The Pong [Atari, 1972] version using the Turtle graphics was the first game developed. This version had errors (bugs). These errors were: the ball was stuck behind the players rackets making it impossible to play and it was the main characteristic of the original game, the ball has velocity and angulation constants. The problem of fixing code also is known as debug which is technique that the programmer verifies each code lines on trying to find errors. Besides this, there was a program good practice to be applied: refactoring (technique used to rewrite/organize a code according by the programmer opinion).

On Pong 2.0, new possibilities have been presented to students with the Pygame library that improves the performance of the game and code, besides making available new more practical functionalities. A good example is the relationship to game sounds, using Turtle it is necessary to use another library that changes according to the operational system. Beyond it was applied software engineering techniques to show the students the importance and how this can make the development easier. Worth highlighting one specific function on this version that was implemented: the second racker was controlled by an AI simulation (Artificial Intelligence).

5.2 Snake

Different from the two previous Pong versions which were given a code with errors to fix, this Snake [Blockade, 1976] version the game was created by zero by students. Also, it started the process to make the code organized using Python functions. On game development, knowing how to handle deadlines is essential to the ethical posture of a developer, starting at this point it should create a weekly plan to show everything will be developed during the week.

The technique applied to this Snake 2.0 was modular programming (instead of writing the code on one file, this technique allows to separate the code in many files which it can communicate itself using an import statement). This kind of program focuses on:

- Joining functions and variables related in single file.
- Game division in modules (which module is responsible of a part of game, per example, the snake module is responsible for the snake on game).
- Making the code maintenance easier.

The introduction of this technique had the goal to prepare for Oriented Object Programming (OOP). The OOP is a programming paradigm that uses objects – data structures that contain data and code to design applications. The OOP promotes modular code, code reuse (it's a technique which the programmer can reuse a code already used previously), encapsulation (it's a technique which protects data and prevents it from being accessed or modified by other part) and a natural way to model real-world concepts and systems [Lafore, 2010].

Beyond the module programming, the Snake 3.0 should use OOP to create the new Snake version, but this time the development process was made with a team. Besides, it should have a snake which has autonomy to fight with the player, it should avoid all the obstacles and eat all fruits.

5.3 Asteroids

From all games developed until now, the Asteroids [Atari, 1979] was the best because it brought all the techniques learned with OOP, modular programming and refactoring. At this game, it should be applied a *Design Pattern* which is a recurring solution to problems that arise when designing software within a particular context and it is a way to reusing knowledge about how to solve a particular type of problem [Gamma et.al, 1994]. The design pattern adopted for this game was *Abstract Factory* which is way to incorporate easily new types of objects (this case these objects actually is the asteroids) or families of objects without requiring major changes to the system itself [Gamma et.al, 1994]. The most common way to create was using arrays and

counters, but the optimization was necessary therefore the *Abstract Factory* is better to simplify the code and make the system more flexible [Gamma et.al, 1994].

6. CONCLUSION

The programming teaching at universities is looked at as a complex and tiring task by many students, but it is essential to graduation. So, finding a way to become something bad seen in something funny and interesting is essential. Thus, using game development to teach looks promising at university, but it is not limited to teaching the basics, but also improving the skills already existing. The arcade games look very good to start with their simplicity when compared to present games and by its simple mechanics as Pong, Snake, Asteroids were very important in it is golden age at last century and until now as teaching tools. How there is different ways to create games, the library specific use from a programming language is very used. As an example, the Turtle graphics used to make draws with geometric forms for the first game, but there are specific libraries to game development as Pygame which it offers more functionalities.

ACKNOWLEDGEMENT

The authors thanks to the Universidade do Estado Amazonas (UEA), Transire Eletrônicos, Tec Toy S.A. and Ludus Lab for their support. The results were published through the research and development activities of project ARKADE AD ASTRA, sponsored by Transire Eletrônicos and Tec Toy S.A, with the support of SUFRAMA under the terms of Federal Law No.8.387/1991.

REFERENCES

Bacich L., Moran J., (2018). *Metodologias ativas para uma educação inovadora: uma abordagem teórico-prática.* Penso Publisher, Porto Alegre, Brazil.

Beck K., (2002). *Test Driven Development: By Example*. Addison-Wesley Publisher, USA.

Chacon S., Straub B., (2014). *Pro Git.* Apress, USA.

Fowler M., Beck K., (2018). *Refactoring: Improving the Design of Existing Code.* Addison-Wesley Publisher, Boston, USA.

G1, (2022). Indice de evasão de alunos é maior na área de tecnologia da informação. Available on https://g1.globo.com/sp/sao-carlos-regiao/noticia/2012/09/indice-de-evasao-de-alunos-e-maior-na-area-de-tecnologia-da-informacao.html

Gamma E., Helm R., Johnson R., Vlissides J., (1994). *Design Patterns: Elements of Reusable Object-Oriented Softwares.* Addison-Wesley Professional Publisher, Indianapolis, USA.

Kolb D. A., (2014). *Experiential Learning: Experience as the Source of Learning and Development*. Pearson Education, New Jersey, USA.

Lafore R., (2010). *Object-Oriented Programming in C++*. Sams Publishing, USA.

Martin C. M., (2008). *Clean Code: A Handbook of Agile Software Craftmanship.* Prentice Hall Publisher, Boston, USA.

Martin R. C., (2003). *Agile Software Development, Principles, Patterns and Practices*. Pearson, USA.

McConnell S., (2004). *Code Complete.* Microsoft Press, Washington, USA.

Menezes C. N. Nilo, (2019). *Introdução a Programação com Python: Algoritmos e lógica de programação para iniciantes.* Novatec Publisher, São Paulo, Brazil.

Payne, B. (2015). *Teaching Kids to Code: An Introduction to Python Programming*. No Starch Press, San Francisco, USA.

Sebesta W. R, (2018). *Concepts of Programming Languages.* Pearson, USA.

Shaffer D. W., (2006). *How Computer Games Help Children Learn*. Palgrave MacMillian, New York, USA.

Sommerville I., (2015). *Software Engineering.* Pearson, Indianapolis, USA.