# CODE CONTROL: DEVELOPING A SERIOUS GAME TO REINFORCE INTRODUCTORY CODING CONCEPTS

Devorah Kletenik
Brooklyn College
kletenik@sci.brooklyn.cuny.edu

Deborah Sturm
College of Staten Island
Deborah.Sturm@csi.cuny.edu

**ABSTRACT:**
**WE REPORT ON THE DEVELOPMENT OF A 3D SERIOUS GAME TO TEACH AND REINFORCE FUNDAMENTAL CODING CONCEPTS. AN INNOVATIVE FEATURE ALLOWS INSTRUCTORS TO CREATE CUSTOMIZED PROGRAMMING CHALLENGES THAT STUDENTS SOLVE IN THE CONTEXT OF THE GAME. THE GAME WAS DESIGNED TO ENGAGE AND MOTIVATE ALL STUDENTS, WITH A SPECIAL FOCUS ON WOMEN AND UNDERREPRESENTED GROUPS IN COMPUTER SCIENCE. A PRELIMINARY EVALUATION OF THE GAME AT THREE COLLEGES IN THE CITY UNIVERSITY OF NEW YORK INDICATES PROMISE IN THE USE OF THE GAME AS A TEACHING TOOL.**

**Keywords: serious games, programming, coding, computer science**

## INTRODUCTION

Majoring in computer science fields offers a number of benefits to students, including a growing and lucrative job market. However, introductory programming courses often have drop-out and failure rates of over 30% – 50% (e.g. Kinnunen & Malmi, 2006). The attrition rate is often significantly higher for female students. It is commonly observed that learning to code is difficult and that difficulty may account for the poor retention rates in the introductory class (e.g. Kinnunen & Malmi, 2006). One factor that may contribute to the poor retention rates is a lack of sufficient programming practice (Beaubouef & Mason, 2005).

The literature on *serious games* discusses strategies to motivate students to practice programming. *Serious games* describes those games that accomplish a goal in addition to entertainment. Research on *serious games for education* shows that, compared to traditional methods of instructor, game-based learning is superior at teaching subject matter and increases both long-term retention and student motivation (Papastergiou, 2009).

Our game, *Code Control*, is an innovative platform where instructors create customized coding challenges that students solve in the context of the game. An automated compiler and testing environment gives students instant feedback, and completing successful programming challenges advances students in the game.

## CODE CONTROL

*Code Control* is a digital game developed using the Unity 3D engine that reinforces and assesses programming concepts. It allows instructors to create customized programming challenges for their

students as well as provide correct solutions. The programming challenges are then posed to the players in the context of the game. We use the Judge0 API (Došilović, 2016) which is a free open-source API for code compilation and execution. Code windows give players code snippets with missing sections to complete; once the players fill in the missing code, the code is compiled. The students' solutions are then checked for correctness by comparing the output to that of the instructor's version. Compiler errors, execution errors, or positive feedback are also provided. Customized scripts provide syntax highlighting. Judge0 supports 42 programming languages so any introductory language will most likely have support in our game. Furthermore, with Judge0 support for multi-file programs, *Code Control* can be used in coding courses at all levels.

While a number of serious games for computer science have been created, including some to teach introductory programming (e.g., (Barnes, Richter, Powell, Chaffin, & Godwin, 2007, Lee & Ko, 2011), all of these games involve fixed coding challenges. In contrast, our approach allows instructors to specify their own challenges that best reinforce what is being taught in class. Additionally, many of these games (e.g. *Gidget* (Lee & Ko, 2011)) use a simplified programming language that was created for the game, and are designed to teach abstract programming concepts. In contrast, our game uses standard programming languages and is designed to augment existing programming courses. It is targeted to students who are learning the basics and want practice to solidify their skills.

The game's storyline involves a woman looking for endangered animals who are missing from an animal rescue. The digital name tags contain code that was corrupted. The player needs to find the lost roaming pets by solving the code to fix their name tags, to save the pets. In designing our game, we consulted with students, both gamers and non-gamers, men and women, and varied races to appeal to a broad audience. It has been particularly designed for a female audience, featuring a female lead character, avoiding violence and focusing on social goals. The avatar's customizable skin/hair colors avoids specifying a specific racial group and gives players co-ownership.

A database connection allows us to collect detailed analytics of all users' gameplay (e.g. time spent per challenge, successes and mistakes made, levels completed, score, etc.) This provides a wealth of information to instructors; instructors who adopt this game as a course material will be able to use the game as an informal evaluation mechanism to see which concepts their students are successful with and with which they need additional practice. *Code Control* is deployed as a WebGL and can be played in a browser without requiring installation.

A pilot study among students in Brooklyn College, College of Staten Island, and Kingsborough Community College, all colleges of City University of New York (CUNY), indicated that students were enthusiastic about the game, even though it was still in beta mode. Survey responses included "It seems like a fun and engaging way to teach programming," "Fun and helpful!" and "Brought a twist to coding that made programming a lot funner than the cut and dry homework assignments." In our pilot, a number of students could not complete the entire game due to browser incompatibilities. Although our focus was on measuring student *interest*, we note that the game had a measurable effect on student *performance*: students who played the entire game had statistically significantly higher scores on post-tests than on the pre-tests (average of 6%); students who did not complete the game showed no such learning gains. We are confident that when the game is further developed, student interest and performance will correspondingly increase.

## ACKNOWLEDGEMENTS

This work was supported by a SIGCSE Special Project Grant. We thank the undergraduate students who coded the game: Ruslan Pantaev, Kwan Holloway, and Mike Williams.

## REFERENCES

Barnes, T., Richter, H., Powell, E., Chaffin, A., & Godwin, A. (2007). Game2Learn: building CS1 learning games for retention. In *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education.*

Beaubouef, T., & Mason, J. (2005). Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin, 37*(2), 103–106.

Došilović, Herman Zvonimir (2016). Judge0 Web API, version 1.2.0. https://judge0.com.

Kinnunen, P., & Malmi, L. (2006). Why students drop out CS1 course? In *Proceedings of the Second International Workshop on Computing Education Research.*

Lee, M. J., & Ko, A. J. (2011). Personifying programming tool feedback improves novice programmers' learning. In *Proceedings of the Seventh International Workshop on Computing Education Research* (pp. 109–116).

Papastergiou, M. (2009). Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation. *Computers & Education, 52*(1), 1–12.