Game Development with a Serious Focus

Devorah Kletenik*
Brooklyn College, City University of New York
kletenik@sci.brooklyn.cuny.edu

Deborah Sturm*
College of Staten Island, City University of New York
Deborah.Sturm@csi.cuny.edu

ABSTRACT

We report our experience teaching elective game development courses at two colleges at a public university. Over the past nine years these courses have been taught in a variety of languages on several platforms. As the courses evolved we introduced serious games with game-based-learning as a focus for the projects and ultimately offered a special topics elective in serious game development. In this paper, we discuss the merits of using serious games as a focus in game programming, including the benefits for students without a strong interest in gaming. We also describe the novel restructuring of one college's Computer Science elective sequence in response to recommendations from students, alumni, and an advisory board of computing professionals. By introducing 200-level electives, students are able to sample advanced topics including game development early in their academic sequence. This has led to involving more students in game-based undergraduate research which can result in increased interest and retention in Computer Science. We discuss our curriculum design and lessons learned including challenges and successes, and data from student surveys indicating student motivation and engagement.

CCS CONCEPTS

• **Social and professional topics** → **Model curricula**; **Computer science education**; • **Applied computing** → **Computer games**;

KEYWORDS

game development, serious games, game-based learning, curriculum design

ACM Reference Format:
Devorah Kletenik and Deborah Sturm. 2018. Game Development with a Serious Focus. In *SIGCSE '18: The 49th ACM Technical Symposium on Computing Science Education, February 21–24, 2018, Baltimore , MD, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3159450.3159588

1 INTRODUCTION

Many Computer Science curricula now include a course in game development; such courses have been shown to motivate and engage a wide range of students. [3, 5, 26]. Game development has also been incorporated into other courses and has been used to teach the introductory programming course [1, 4, 9, 23], Software Engineering [7, 25], Artificial Intelligence [2, 34]), Computer Graphics [33] and other topics (e.g. [22]). Game development courses typically cover at minimum, game genres, sprites, movement, assets, animation, collisions, and user interaction. Additional topics include AI, effects, shaders, physics, and 3D development. They also usually include a major project that, because of its size, is often designed and coded as a team. In our experience in teaching game development we found that it is useful to have a common project theme. We proposed that the students create serious games with a purpose, specifically those that employ game-based learning techniques [16] to teach a college-level topic. Part of the project included playtesting and assessing whether the game met learning outcomes.

In this paper, we discuss the game development courses at two senior colleges of City University of New York (CUNY). CUNY is the public university system of New York City and the largest urban university system in the United States; we focus on two senior colleges of CUNY, The College of Staten Island (henceforth, "Institution 1") and Brooklyn College ("Institution 2").

We first review other attempts to teach game development courses both with and without a serious game focus. We discuss platforms and languages that we tried in our courses, along with our final recommendation for the use of Unity. We describe the curriculum of the courses and list the required deliverables. Finally we present data from student surveys.

2 BACKGROUND

Game development courses are popular at many colleges because they engage students by allowing them to use their programming skills in real-world applications that interest them. The use of games motivates students across a range of courses, from introductory programming through research capstones [26] and many college now include courses in game programming [4, 15, 27, 29]. The field is so large that at this point, it is now a concentration within the CS major [10].

Game programming courses have been taught using a variety of different languages. Doss et al. [14] present a number of them, including XNA, Game Maker, and Scratch. More recently, Dickson [13] and Ivanov [19] reported on the use of Unity in game design courses.

Serious games is the term for games that accomplish a goal in addition to entertainment. Examples of serious games include games used for crowd-sourcing problem solving (e.g. Foldit[1]), recruitment (e.g. America's Army[2]), simulations (e.g. Zero Hour: America's Medic[3]), or to encourage exercising (e.g. Wii-Fit[4]). Probably the

*Both authors contributed equally to the paper

SIGCSE '18, February 21–24, 2018, Baltimore , MD, USA

ACM ISBN 978-1-4503-5103-4/18/02…$15.00
https://doi.org/10.1145/3159450.3159588

[1] https://fold.it/portal
[2] https://www.americasarmy.com/
[3] http://www.virtualheroes.biz/zerohour
[4] http://wiifit.com

most popular subset of serious games is games used for education, or Game Based Learning (GBL). Computer games for education (Digital GBL, or DGBL) have been popular since the dawn of personal computers and are used in a variety of fields and levels of education. We describe our experiences in using serious games, specifically DGBL, as the focus of a game development course.

Involving students as creators of educational games has a long history. As early as 1992, Ritchie and Dodge report on their creation of an adventure game authoring template that allowed middle-school students to design educational games in a variety of disciplines [28].

More recently, Chaffin and Barnes [8] presented a detailed informative paper about their creation of a serious games course, including a report on successes and failures in their pilot version. Kapralos wrote a comprehensive paper about a serious game course that was offered as part of a Game Development and Entrepreneurship undergraduate degree program [20] and followed up in 2015 in [21]. A difference in our approach is that students who took the serious games courses described in the aforementioned work already had a solid background in creating games. We, on the other hand, recommend offering a serious games course at an early stage of the major, or *as a replacement for* the game development course. In a degree program that offers only one game development course, we have found that a project-based learning approach to the course centered around serious games has been successful.

Brown et al. [6] describe a capstone project in creating educational video games. This project was offered to Computer Science, Education, and Digital Media students in the hopes of creating multidisciplinary teams. The games were created using Scratch, and the emphasis was on non-technical "soft skills," such as teamwork and presenting. This differs from our approach, which is to use serious games to teach a complete course in game programming.

3 WHY SERIOUS GAMES?

We identify the following benefits to focusing on serious games in a game development course rather than focusing on recreational games:

(1) It offers students a meaningful goal for their project. It is our belief that students engaged in learning that has a socially-minded component are more engaged and more motivated, and develop the perspective to think in bigger pictures. As one student commented, *"Before this class, I honestly thought games were a waste of time. I had no idea that you could use games for good. [I learned] that you could accomplish a lot with games."*
(2) It offers students a clear focus for their work, giving students a clear place to begin and a means to structure their game.
(3) It encourages creativity and discourages recycling code from online tutorials, articles, assets and scripts.
(4) We hypothesize that this focus appeals to non-gamers, including some women. Non-gamers who take game design/ programming courses are often at a disadvantage due to their limited exposure to the medium. It has been observed that there are fundamental differences in the ways that male and female players play computer games: for example, males tend to play games more frequently and for longer durations than do their female peers [11]. In our experience, our gamer students tend to be predominantly male. As a result, female students can feel marginalized from the beginning. We suggest that the use of serious games levels the playing field.
(5) It can serve as a hook to engage undergraduate students in research. Involving undergraduates in research contributes to student retention, interest in careers in STEM, and graduate school enrollment [24, 30]. However, it can be challenging to attract students to research. We find that game development is a major draw for student research: it's engaging, fun, and appeals to what they already know. In our approach, we recommend introducing research-based games early in the curriculum and to encourage students to design games that could be used as part of a research project.

4 EARLY ELECTIVES

The NSF's Integrative Computing Education and Research (ICER) Initiative recommends five strategic initiatives; the third Strategic Initiative includes: "Improve retention by recasting introductory courses as pumps, not filters" [12]. These recommendations motivated restructuring the elective sequence at Institution 1 [31].

These changes were further informed by several issues reported by the Departmental Outcomes Assessment Committee during a periodic review of assessment data. Several constituencies, including graduating seniors, alumni, and computing professionals, recommended introducing new courses to keep pace with the rapidly changing computing field. They specifically mentioned Game Development as an example of a course that would be helpful to graduates.

As a result of this recommendation, in 2007, Institution 1 introduced several new electives at the 200-level that only required CS1 as a prerequisite. By positioning them early in the major sequence, advanced and applied topics were introduced earlier in the major in line with the ICER recommendation to create "pump" rather than "filter" courses. The early electives include Hacking Revealed, Robotics, Web Database Applications, Introductory Game Development and Introduction to High Performance Computing. In order to make the coursework more practically oriented and lower the barrier to entry, courses such as Game Development make use of pre-built scripts and prefabs.

The restructuring of the elective sequence resulted in three positive outcomes: (1) potential Computer Science majors can sample advanced and/or applied topics early in their academic career; (2) students from other disciplines can take interesting and popular courses without completing several prerequisites; (3) Computer Science students are able to take electives (two early electives = one full elective) earlier and not overload themselves with several courses in their last semester. Additionally, moving the game electives earlier in the pipeline allows students to be hooked and begin working on research immediately, rather than waiting until they complete a 400-level course, when they are already poised to graduate. We discuss some of our student research projects in Section 7.2.

Game development was a perfect way to engage students in programming. As one student commented, *" This course had me more motivated to program without really realizing that I was programming. Meaning, that I got so immersed in the creation process*

that I stopped thinking about programming as this formal entity that is presented in other classes. It almost felt like I was taking an art class and I was looking for new techniques to create something in front of me, when in fact later I realized I had written hundreds of lines of code. . . . It was one of the few classes where you literally saw your code come to life."

5 CURRICULUM AND DELIVERABLES

Our game programming with a serious games focus at both colleges have a similar curriculum and for ease of presentation, we discuss both curricula as one. We begin with an overview of known games with a purpose. We discuss and play several serious games including Foldit and Eyewire[5]. We then showcase game-based learning games such as Oregon Trail, Zoombinis[6] and Spent[7]. We concluded the semester with TED talks by game designer Jane McGonigal[8][9], which demonstrate the power of serious games.

We used James Paul Gee's Principles of Learning as a framework for how to design the learning components of the games [16]. Dr. Gee proposes designing games to produce "Empowered Learners." We emphasized his principles including co-design, customizing, identity and presenting well-ordered problems. We emphasized that the games should be pleasantly frustrating so that the player wants to play (i.e. learn) and persist. The game shouldn't be too easy so that the player is bored and stops playing, nor should it be too difficult to avoid quitting because of frustration. Instead the game should get more difficult at a pleasant pace to increase learning and persistence. The feedback to the player should be positive when learning is occurring and not overly negative when the player doesn't succeed. We also suggested his "on demand" information and "just-in-time" instructions. Our rubrics for peer-evaluation asked about whether the game incorporated these principles and whether it was working for the player.

The student group assignments were organized around designing, creating, and evaluating a serious game. The first group assignment was to research a game-based learning game, test it and report on its objectives and learning outcomes by completing a provided rubric. Each group then chose a college-level topic to teach and a genre. The next deliverable was a pitch to their peers where they also presented how each learning objective would be assessed. They then prepared a game design document (based on a template) with most of the chosen assets, including graphics, music, sound effects. Students demoed the project to their peers again after completing a teaching and assessment level. We didn't dictate content or genres but we encouraged students to design games to teach material at the introductory to advanced college level. We specified that each game have both teaching and assessment components. We also encouraged including high scores (using PlayerPrefs in Unity) to show the player their progress. The penultimate week of the semester was reserved to give each group a chance to evaluate the games of all other groups, using the same game evaluation rubric that they used in the first assignment. The students then had the chance to incorporate the feedback that they received into their final game submissions.

Each group prepared pre-surveys to assess the prior knowledge of their players and post-surveys to test whether their game met their learning outcomes. For example the pre- and post- surveys for the Linear Algebra game had dot-product problems to fill in. The final project included the completed surveys with a report summarizing the results.

Our students produced an array of impressive serious games for education with educational content that spanned the fields of biology, chemistry, physics, health, English, foreign languages, music, computer science, math, and history. We present a list of our student projects (from students at both colleges) in Table 2.

6 DEVELOPMENT TOOLS - WHY UNITY?

There are many excellent free game development tools available and our introductory and advanced courses have been taught using a variety of environments. As organized by user interface, Scratch and Appinventor are both visual. XNA and Xcode/objective C are primarily code-based. The Unity game engine is the most integrated in that the visual environment can be manipulated using drag-and-drop (including code) yet the more advanced developer can code their own scripts.

For the 200-level course we initially used Scratch and Appinventor. These are both excellent at conveying gaming principles without getting sidetracked by syntax and deployment issues. We later added XNA and Xcode using Objective C. In both cases we gave the student templates to get their projects started in this introductory course. In the 400-level electives we used XNA and Xcode/Objective C. Student feedback at both levels was overwhelmingly in favor of XNA and this enthusiasm was matched by stellar projects. However, when Microsoft stopped supporting XNA we researched other options. Unity was an appealing choice since it is used to produce professional games and can still be programmed by relative beginners. It uses a combination of a drag-and-drop user interface and C# scripting, which enables students of all levels to develop games; it supports both 2D and 3D game creation; and it can create games for multiple platforms, including Web players and mobile apps. Students are very aware of the commercial success of Unity and were excited and motivated to learn the interface. The Mac and Windows versions are very similar and there were few deployment issues. In our post-course survey, students unanimously reported that they enjoyed Unity. In their list of reasons, they included that they found it easy to learn and that they liked coding with the C# scripts.

At Institution 2, a game design course has also taught using Processing. Processing is based on Java, and was therefore easy for our students (all of whom had taken a course in Java) to learn. It easily supports graphics and is useful for highly visual programs. It also supports many external libraries (such as the `minim` audio library that one group of students used for a music-based game). However, it is not built for game development and many students found it clunky and difficult to use for that purpose. Due to student demand, we switched to Unity for the game programming course.

[5] http://eyewire.org/explore
[6] https://itunes.apple.com/us/app/zoombinis/id961739806?mt=8
[7] http://playspent.org/
[8] https://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world
[9] https://www.ted.com/talks/jane_mcgonigal_the_game_that_can_give_you_10_extra_years_of_life

Week #	Topics covered	Student assignment/deliverables
1	Intro to game development, serious games, principles of GBL	game evaluation
2	Intro to Unity (assets, sprites, prefabs, scenes, gravity, tags)	
3	movement, user controls, transformation & rotations, velocity	game concept document "pitch" (and peer evaluation)
4	collision detection, audio effects & volume control, spawning	
5	sprite sheets & animations	
6	text, scenes	midterm
7	buttons and time	first submission of game (and peer evaluation)
8	data persistence: static classes	
9-10	data persistence, particle systems, Google Analytics	
11	2.5D games, models & textures, camera projections	final games due
12	peer evaluation of final games	submission of pre-/post-tests for review, lab assignments due
13	final review, summary of serious games	evaluations (peer) and pre- and post-test evaluations
14	final	final projects due with user data (scores on pre- and post- tests)

Table 1: Curriculum and Student Deliverables

Game name	Topic	Genre
American Revolution	History	Turn-based Strat.
Balance	physics	Physics puzzle
Binary Battleship	binary numbers	Turn-based Tact.
Egg Bash!	decimal ↔ binary	Drag/Drop
The GREat Escape	GRE vocabulary	Platform
HangukeoBaeUgi	Korean characters	FPS
The Happiness Crusade	drugs & alcohol	RPG
Khemicalz	chemistry	Puzzle-platform
Loop()	programming	Puzzle
Lunario	chemistry	Platform
Math Tac Toe	vector operations	Strategy
Miss Match	pattern matching	Platform
Night Of The Cancer	cancer facts	Action-adventure
Orchestroids	pitch	3D FPS
Recycle Hero	recycling	Platform
Siege of Bastogne	WWII history	Tower Defense
Spell Invaders	homonyms	FPS

Table 2: Student Game Projects

Inst.	Course Title	Level	Environment / Lang.
2	Game Development	400	Processing
2	Game Programming (Serious games)	400	Unity
1	Intro Game Devel.	200	XNA (C#) XCode (Objective C)
1	Serious Game Devel.	200	Unity
1	Advanced Game Devel.	400	Unity

Table 3: Languages used in game courses at both colleges

7 STUDENT OUTCOMES

At the end of the most recent semester that this course was taught, students were given a survey measuring their attitude towards the course and its aims. Twelve out of the twenty-one students in Institution 1 and all thirty-six out of thirty-six in Institution 2 completed the survey. The results of the survey were anonymous. About half of the students in both courses preferred to create a serious game over one purely for entertainment. Three-quarters of the students in Institution 2 and over 90% in Institution 1 would be interested in creating a serious game for research. In both schools, almost all the students replied that they would recommend the course to a friend and that they enjoyed the use of Unity. In Table 4, we present all the questions responses.

We also gave an open-ended question as a follow-up for question #1: *Why did you enjoy creating a serious game or why would you have preferred to create one purely for entertainment?* Here is one representative response:

"A serious game is meaningful. When we were creating the game, we were thinking. We were thinking how to make people learn from our game and enjoy our game. Making a serious game is beneficial for the creator and players. Players learn stuff creators want to teach them. Creators learn those stuff while they create their games."

7.1 Non-gamer students

Most but not all of the students took the games courses because they already had an interest in games. Some of the non-gamers reported that developing a game with a purpose leveled the playing field for them. We hope that this serious focus will encourage more non-gamers to take the game development courses, especially women. The following are quote from female students who took the course: *"I preferred that the class focused on serious games. I'm not a game player at all. I never played video games or computer games as a child. I barely ever play games on my iPhone. Even if someone has not played many games, almost everyone has played an educational game at some point in their life (whether in school or at home). Therefore, being assigned a serious game made the task more relatable and less daunting. If we had to create an adventure game I would find it difficult to figure out all the concepts involved because I don't have extensive experience playing video games."*

"I rarely play computer games. . . . I liked having the serious requirement for our group project. Giving the requirement of an educational game helped to focus the group and narrow down ideas. I think that in terms of gender parity, this is the way to go. In my experience, most

General questions about the course

	Institution 1, $n = 12$			Institution 2, $n = 36$		
Question	**yes**	**no**	**no opinion**	**yes**	**no**	**no opinion**
Did you prefer to have a serious focus instead one purely for fun?	41.7%	33.3%	25%	48.5%	27.3%	24.2%
Did you enjoy using Unity?	100%	0%	0%	93.9%	3.0%	3.0%
Would you be interested in creating a serious game for research?	91.7%	0%	8.3%	78.5%	3.0%	21.2%
Would you recommend this course to a friend?	100%	0%	0%	97.0%	0%	3%

Specific questions about Unity and the format of the course. Multiple choices allowed.

I liked Unity because	**Inst. 1**	**Inst. 2**
It's easy to learn	91.7%	57.58%
It's used in commercial games	41.7%	63.64%
It has the capability to do lots of cool things.	66.7%	78.788%
Other	8.3%	3.0%

What did you like about the format of the course?	**Inst. 1**	**Inst. 2**
Peer feedback	91.7%	36.7%
Working in groups	66.7%	51.5%
Using Unity	75%	93.9%
Testing the educational objective through pre- and post- tests	33.3%	21.2%
Having a Web player version	41.7%	54.5%

Table 4: Student responses to survey

educational games are fairly gender neutral, whereas many entertainment games are marketed towards men (e.g. first person shooter games that sell so well, but very rarely feature female protagonists)."

We suggest that using a serious games focus to the course will ultimately encourage more female students to be interested in the course and will offer a way to recruit more female students into a very heavily male-dominated course.

7.2 Student Research

As we pointed out in Sections 3 and 4, one major goal of both focusing on serious games and positioning this course as an early elective is to use the course as a hook to involve students in research. We have successfully recruited students who enjoyed the course to engage in research in serious games, including in the creation of two games for players on the autism spectrum. One game for the iPad assesses emotion recognition [32]; the second is a Kinect game that assesses and encourages collaborative learning [17]. For each we are working with domain experts from our Psychology department. One novel aspect of this project is that during the design phase, we are getting feedback from college students on the autism spectrum. Another area of undergraduate research in serious games was in gamifying the statistics lab for the Psychology department. The students collaborated with the instructors to design and develop a game-based set of labs that are currently in use in the course [18]. Yet another team is working on measuring the impact of game design elements such as storyline, sound effects, and power-ups in serious games for CS education. Students have presented their research at three game-based-learning conferences, one game expo, and at nine college undergraduate research conferences. One student was hired as as a Unity Developer for an educational gaming company based directly on his course and research experience.

8 LESSONS LEARNED

Our combined 9 years of teaching game development to over 200 students has led to several take-away lessons. In this discussion we include our experience in having students work on game-related undergraduate research projects during and following the courses.

8.1 Challenges

It is challenging to cover all the material and have the students produce a project they are proud of in a one semester course. We have had several students interested in continuing to work on independent study projects in order to deepen their knowledge of gaming and to get research experience. However if the game development course is an advanced elective, students will typically take the course in their last or penultimate semester. They graduate just as they are prepared for a substantive research project. As in other team-based projects, another issue that we deal with is managing team dynamics. In the institution where we have early electives, the intro to game development has two challenges that we needed to address. One is that some students only have the minimal prerequisite of introductory programming whereas others have completed data structures and object oriented development. We dealt with this disparity by making sure the teams were balanced so that no group had only beginners.

Finally we had to address assessment issues since the typical student did not have any background in this area. We provided examples of pre- and post-tests that meaningfully assess student learning, taught students how to create customized tests to assess the impact of their games, and had students submit their tests separately for our critique before using them for evaluation.

8.2 Successes

Although having students on varying technical levels was a challenge, having diverse teams was very beneficial. Students learned from each other especially when one team member was an expert in the subject being taught. For example in the group that produced the game to teach elements of Korean, the student who did not know Korean made sure that players learned the basics. Having students taking an early elective in game development allowing for

early training in this area. Students were then more prepared to work on research projects spanning more than one semester. By choosing college level topics to teach, the students were able to easily find friends and relatives who did not know the topic and to then assess if their game taught the material. For example one group wrote a game to teach elements of Linear Algebra such as the dot product.

When a group demoed their game to the class, we assigned a different member of the class to play the game. In this way the group could see whether the game-play was intuitive. Also, towards the end of the semester, we told the students that we were asking our faculty colleagues to play the games to get their feedback. This motivated them to add polish to their games. Additionally, the level of the games produced rose significantly when we moved the web versions of the games onto the college server.

Another suggestion that we felt worked well was to allow our students to create an educational game on any topic of their choice. Our students enjoyed applying their unique backgrounds and interests to their projects, and created games on a broad array of topics.

We had excellent attendance (even when one section ran from 8 to 10:00 pm) and responsible team-work. Student satisfaction was high and most significantly, several students continued working on projects after the semester ended and presented their work at game-based learning conferences. We think that our experiences can offer guidance to other instructors who teach game development courses.

9 CONCLUSIONS

We found teaching game development with a serious focus to be a successful strategy in teaching gaming concepts, motivating high-level projects and generating interest in game-based research for both gamer and non-gamer students. The use of Unity in the course motivates students and provides a widely-used and flexible platform for game development. We successfully implemented the course in two colleges, and as a 200-level course in one institution. Having a 200-level game development course introduced students to this topic earlier in their academic careers and prepared interested students to work on game-based research projects. In the future, we would like to introduce the course at the 200-level at Institution 2 as well, and we recommend this to others who would like to try this course. Offering the course as an early elective helps flatten the curriculum, thereby making computer science more available for all.

REFERENCES

[1] Jessica D Bayliss. 2007. The effects of games in CS1-3. In *Microsoft Academic Days Conference on Game Development in Computer Science Education*. 59–63.

[2] Jessica D. Bayliss. 2012. Teaching game AI through Minecraft mods. In *Games Innovation Conference (IGIC), 2012 IEEE International*. IEEE, 1–4.

[3] Jessica D. Bayliss and Sean Strout. 2006. *Games as a Flavor of CS1*. Vol. 38. ACM.

[4] Katrin Becker and JR Parker. 2007. Serious games+ computer science= serious CS. *Journal of Computing Sciences in Colleges* 23, 2 (2007), 40–46.

[5] Kevin Bierre, Phil Ventura, Andrew Phelps, and Chris Egert. 2006. Motivating OOP by blowing things up: an exercise in cooperation and competition in an intro. Java programming course. In *SIGCSE Bulletin*, Vol. 38. ACM, 354–358.

[6] Quincy Brown, Frank Lee, and Suzanne Alejandre. 2009. Emphasizing soft skills and team development in an educational digital game design course. In *Proceedings of the 4th international Conference on Foundations of Digital Games*. ACM, 240–247.

[7] Nergiz Ercil Cagiltay. 2007. Teaching software engineering by means of computer-game development: Challenges and opportunities. *British Journal of Educational Technology* 38, 3 (2007), 405–415.

[8] Amanda Chaffin and Tiffany Barnes. 2010. Lessons from a course on serious games research and prototyping. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*. ACM, 32–39.

[9] Daniel C. Cliburn and Susan Miller. 2008. Games, stories, or something more traditional: the types of assignments college students prefer. *ACM SIGCSE Bulletin* 40, 1 (2008), 138–142.

[10] Ron Coleman, Mary Krembs, Alan Labouseur, and Jim Weir. 2005. Game design & programming concentration within the computer science curriculum. In *ACM SIGCSE Bulletin*, Vol. 37. ACM, 545–550.

[11] T.M. Connolly, E.A. Boyle, M.H. Stansfield, and T. Hainey. 2007. The potential of online games as a collaborative learning environment. *Journal of Advanced Technology for Learning* (2007).

[12] Judy Cushing, Rob Bryant, Genevieve Orr, Sylvia Spengler, Sharon Tuttle, and Ken Yasuhara. 2006. NSF'S Integrative Computing Education and Research initiative: whither the northwest. *J. of Computing Sciences in Colleges* 22, 2 (2006), 49–51.

[13] Paul E. Dickson. 2015. Using Unity to Teach Game Development: When You've Never Written a Game. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*. ACM, 75–80.

[14] Kathlyn Doss, Valerie Juarez, Daniel Vincent, Peggy Doerschuk, and Jiangjiang Liu. 2011. Work in progress – A survey of popular game creation platforms used for computing education. In *Frontiers in Education Conference (FIE)*. IEEE, F1H–1.

[15] Shannon Duvall. 2009. Creating a games class: a walkthrough. In *Proceedings of the 4th International Conference on Foundations of Digital Games*. ACM, 279–284.

[16] James Paul Gee. 2003. What video games have to teach us about learning and literacy. *Computers in Entertainment (CIE)* 1, 1 (2003), 20–20.

[17] Kristen Gillespie, Gabriel Goldstein, David Shane Smith, Ariana Riccio, Michael Kholodovsky, Cali Merendino, Stanislav Leskov, Rayan Arab, Hassan Elsherbini, Pavel Asanov, and Deborah Sturm. 2017. Connecting Through Kinect: Designing and Evaluating a Collaborative Game with and for Autistic Individuals. In *International Conference of Design, User Experience, and Usability*. Springer, 398–413.

[18] Evan Grandoit, Rose Bergdoll, Jenny Chan, Laura Rabin, Devorah Kletenik, Chelsea Chung, Wei Zhang, Ecem Olcum, Christopher Menedes, Ali Rishty, and Beliz Hazan. 2017. Comparison of learning efficiency with and without relevant extrinsic rewards in gamified psychological statistics classrooms. Northeast Conference for Teachers of Psychology (NECTOP) Poster Session.

[19] Lubomir Ivanov. 2015. 3D game development with unity in the computer science curriculum. *Journal of Computing Sciences in Colleges* 31, 1 (2015), 167–173.

[20] Bill Kapralos. 2012. A course on the design and development of serious games and virtual simulations. In *Games Innovation Conference (IGIC), IEEE Intl.* 1–4.

[21] Bill Kapralos, Stephanie Fisher, Jessica Clarkson, and Roland van Oostveen. 2015. A course on serious game design and development using an online problem-based learning approach. *Interactive Tech and Smart Education* 12, 2 (2015), 116–136.

[22] Stan Kurkovsky. 2009. Engaging students through mobile game development. *ACM SIGCSE Bulletin* 41, 1 (2009), 44–48.

[23] Scott Leutenegger and Jeffrey Edgington. 2007. A games first approach to teaching introductory programming. *ACM SIGCSE Bulletin* 39, 1 (2007), 115–118.

[24] David Lopatto. 2007. Undergraduate research experiences support science career decisions and active learning. *CBE-Life Sciences Education* 6, 4 (2007), 297–306.

[25] Manuel Palomo-Duarte, Juan Manuel Dodero, José Tomás Tocino, Antonio García-Domínguez, and Antonio Balderas. 2012. Competitive evaluation in a video game development course. In *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*. ACM, 321–326.

[26] Ian Parberry, Timothy Roden, and Max B Kazemzadeh. 2005. Experience with an industry-driven capstone course on game programming. In *ACM SIGCSE Bulletin*, Vol. 37. ACM, 91–95.

[27] Yolanda Rankin, Amy Gooch, and Bruce Gooch. 2008. The impact of game design on students' interest in CS. In *Proceedings of the 3rd international conference on Game development in computer science education*. ACM, 31–35.

[28] Donn Ritchie and Bernard Dodge. 1992. Integrating Technology Usage across the Curriculum through Educational Adventure Games. (1992).

[29] Timothy E Roden. 2014. Benefits of an introductory course in computer game development. In *Computer Science & Education (ICCSE), 2014 9th International Conference on*. IEEE, 316–321.

[30] Susan Russell, Mary Hancock, and James McCullough. 2007. Benefits of undergraduate research experiences. *Science(Washington)* 316, 5824 (2007), 548–549.

[31] Deborah Sturm and Roberta Klibaner. 2012. Early introduction of advanced CS topics to increase student satisfaction. *Journal of Computing Sciences in Colleges* 27, 6 (2012), 26–33.

[32] Deborah Sturm, Ed Peppe, and Bertram Ploog. 2016. eMot-iCan: Design of an assessment game for emotion recognition in players with Autism. In *Serious Games and Applications for Health (SeGAH), 2016 IEEE Intl Conference*. IEEE, 1–7.

[33] Kelvin Sung, Peter Shirley, and Becky Reed Rosenberg. 2007. Experiencing aspects of games programming in an introductory computer graphics class. In *ACM SIGCSE Bulletin*, Vol. 39. ACM, 249–253.

[34] Scott A Wallace, Robert McCartney, and Ingrid Russell. 2010. Games and machine learning: a powerful combination in an artificial intelligence course. *Computer Science Education* 20, 1 (2010), 17–36.