




Using Unity to Teach Game Development

Vladyslav S. Kuznetsov¹, Mykhailo V. Moiseienko¹ ^a, Natalia V. Moiseienko¹ ^b,
Bohdan A. Rostalny¹ and Arnold E. Kiv² ^c

¹Kryvyi Rih State Pedagogical University, 54 Gagarin Ave., Kryvyi Rih, 50086, Ukraine

²Ben-Gurion University of the Negev, P.O.B. 653, Beer Sheva, 8410501, Israel

Keywords: Computer Science, Education, Computer Game Development, Unity Engine, Software Engineering.

Abstract: The article gives an overview of issues arising in connection with the organization and conduct of the course “Computer game development” in the master’s program 014.09 Secondary education (Informatics). The study of the experience of similar courses in other educational institutions does not give an idea of what conditions are the best for conducting it, because “Computer game development” is mainly present in the curriculum of the Software Engineering specialty and is not limited to one course. The game development course is described in terms of content, software and teaching methods. This course, which was attended by 40 students in three years, was evaluated in the light of the approach proposed by A. D. Ritzhaupt and based on the students’ opinion. As a result of this research, it was concluded that a course in video game development could be based on the Unity Engine, as it has a small entry threshold, free for academic purposes, a crossplatform, real game engine, common in the gaming industry. A team strategy for this course is also effective.

1 INTRODUCTION

The software industry is a dynamic and market-oriented industry (Vakaliuk et al., 2020b). Along with the film industry, video games are one of the most interesting and popular applications of information technology (Haranin and Moiseienko, 2018; Katsko and Moiseienko, 2018).

Gamesindustry.biz journalists, along with analysts from Newzoo, UKIE, Sensor Tower, IHS Markit, ICO Partners and Fancensus, published a large infographic on the state of the game industry in 2020 (figure 1). It implies that computer games occupy a significant segment of the software market.

Educational institutions that train software engineers include game development as a set of relevant subjects in the curriculum. The reasons for their inclusion are diverse and include, inter alia, improving the effectiveness of the curricula (Barnes et al., 2007; Claypool and Claypool, 2005; Morrison and Preston, 2009; Roden and LeGrand, 2013; Sung, 2009), increasing the competitiveness of graduates in the modern labour market (Haranin et al., 2017; Vakaliuk

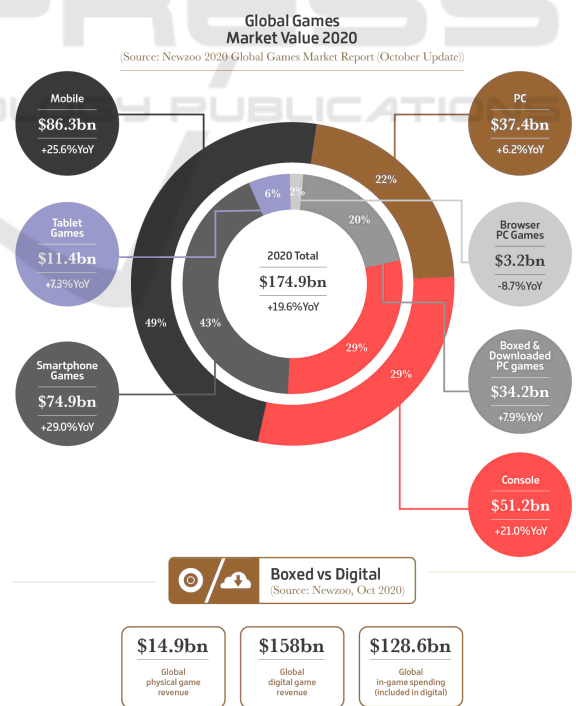





Figure 1: Global Games Market Value 2020 (Batchelor, 2020).

et al., 2020a), learning team work (Brown et al., 2009;

^a  <https://orcid.org/0000-0003-4401-0297>

^b  <https://orcid.org/0000-0003-0789-0272>

^c  <https://orcid.org/0000-0002-0991-2343>

Rankin et al., 2008) and project management (Barnes et al., 2007; Claypool and Claypool, 2005).

Despite this, many higher education institutions carefully include such courses for a variety of reasons, including the need to create games of some interdisciplinary skills, time constraints, lack of interest and experience among teachers in teaching games or the view that game development is not serious (Becker and Parker, 2007; Martin and Smith, 2002).

As teachers of Computer Science at Pedagogical University, we consider the inclusion of a variation course of computer game development in the curriculum of Masters 014.09 Secondary Education (Informatics) as a way to increase the level of motivation, engagement and professional pleasure for students.

The *purpose* of this article is to give an idea of the creation of a separate game development course, an overview of the methodology and tools used in its construction and the conclusions drawn so that teachers without experience, all of which have set such a course, have been able to avoid unnecessary waste of time. We describe our experience in creating an individual game development course using the Unity Game Engine (Unity Technologies, 2021). The aim of this course is to familiarize students with the development of games using the software available in industry. We focus on problem solving, project planning, SDK work, and teamwork, and see this course as a way to entertain and motivate students.

2 BACKGROUND

The first task of the game development course was to select an approach. Defining the content, goals and objectives of game development is an important step, especially in the light of limited material and time resources.

A review of publications on the subject shows that the implementation of training programmes on game development is quite diverse. It varies from individual courses (Jones, 2000; Parberry et al., 2005; Sweedyk and Keller, 2005) and the inclusion of relevant sections in the traditional computer science program (Coleman et al., 2005) before the course sequence (Clark et al., 2007; Fachada and Códices, 2020; Parberry et al., 2006; Rocco and Yoder, 2007; Prokhorov et al., 2021). Content of individual courses from the use of engines developed for training (GAMEMAKER (Claypool and Claypool, 2005), RPG Maker (Barnes et al., 2007), Alice (Werner et al., 2012)), development of own game engines (Labyrinth (Distasio and Way, 2007; Shultz, 2004), CAGE (Vanhatupa, 2011)), technical design (Parberry et al., 2006), Flash (Estey

et al., 2010) to a complete game development training course covering all aspects of the game (Jones, 2000; Martin and Smith, 2002).

The idea of developing a proprietary engine seems tempting at first, but, in experience, does not pay for itself by the time it takes, and eventually students will never see it again after the course (Dickson, 2015). The real game engine should simplify and speed up the development process and allow students to create interesting games in a short period of time. The problem of finding the most suitable game engine for this course is not very simple, and there are different opinions on this issue from the XNA Game Studio library to Unity and Unreal (Dickson, 2015; Dickson et al., 2017; Unreal Engine, 2021; Harris, 2011; Linhoff and Settle, 2008; Parberry et al., 2005; Peng, 2015). Dickson (Dickson, 2015) offers to use the Unity game engine (Unity Technologies, 2021) to teach game development. Given its widespread use in the industry (de Macedo and Rodrigues, 2011; Toftedahl and Engström, 2019) and even for teaching game development in the middle school (Comber et al., 2019), this seems logical.

There are also several important CS sections directly used in the development of computer games: the basics of physics, multimedia, network basics, computer graphics, and the basics of game artificial intelligence (Ahlquist and Novak, 2007; Millington, 2019; Yannakakis and Togelius, 2018).

Game design usually refers to the design of the game and focuses on story, mechanics, character modelling, environment, process content generation, etc., which is enough material to take a whole semester without going into too much detail. There are many textbooks covering these broad topics, such as (Adams, 2013; Ahlquist and Novak, 2007; Saulter, 2007; Bond, 2014). These areas are compulsory for the course.

3 SELECTING THE SOFTWARE

Once the approach to the gaming course was defined, the next question we faced was what tools to use to create games.

More recently, developers have made widely available many powerful game engines and development environments that provide functionality for video game development. An overview of some of the best known is presented below.

Godot Engine (Bradfield, 2018; Manzur and Marques, 2018)

Cost and Licensing: Completely free and open source under the permissive MIT license.

System Requirements (minimum): Memory: 4 GB, Graphics Card: NVIDIA GeForce 6200, CPU: Intel Core 2 Duo E8400, OS: Windows 7.

Platforms: Linux, Windows, OS X, Wii, Nintendo 3DS, PlayStation 3, PS Vita, Android, iOS, BBX, web-games with asm.js, NativeClient.

Overview and Features: Godot Engine is a feature-packed, cross-platform game engine to create 2D and 3D games from a unified interface. It provides a comprehensive set of common tools, so users can focus on making games without having to reinvent the wheel. Games can be exported in one click to a number of platforms, including the major desktop platforms (Linux, macOS, Windows) as well as mobile (Android, iOS) and web-based (HTML5) platforms.

Unity Engine (Unity Technologies, 2021)

Cost and Licensing: Personal Free version (your project revenue or funding cannot exceed \$100,000 a year), Unity Pro package \$125 per month (includes an impressive amount of services not included in the free version).

System Requirements (minimum): Graphics Card: DX10, DX11, and DX12-capable GPUs, CPU: X64 architecture with SSE2 instruction set support, Windows 7 (SP1+) and Windows 10, 64-bit versions only.

Platforms: Android, iOS, Windows Phone 8, BlackBerry, PS3, Xbox360, Wii U and web-browsers.

Overview: Unity is a cross-platform game engine. The engine can be used to create 2D/3D, virtual reality, and augmented reality games, as well as simulations and other experiences (Axon, 2016; Takahashi, 2018). The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering and construction.

Features: Creating and Destroying GameObjects, Access the Components, Events for GameObject, Dealing with Vector Variables and Timing Variables, Physics Oriented Events, Coroutine and Return Types.

Unreal Engine (Unreal Engine, 2021)

Cost and Licensing: Free (5% royalty on gross revenue more than \$1,000,000).

System Requirements (minimum): CPU: Quad-core Intel or AMD processor, 2.5 GHz or faster, Graphics Card: NVIDIA GeForce 470 GTX or AMD Radeon 6870 HD series card or higher, RAM: 8 GB Windows 7 64-bit or Mac OS X 10.9.2 or later.

Platforms: iOS, Android, Windows Phone 8, Xbox360, PS 3, PlayStation Vita, Wii U.

Overview and Features: Unreal Engine is a complete suite of development tools for anyone working with real-time technology. From design visualizations and cinematic experiences to high-quality games

across PC, console, mobile, VR, and AR, Unreal Engine gives you everything you need to start, ship, grow, and stand out from the crowd.

XNA Game Studio (Harris, 2011; Linhoff and Settle, 2008; Miles, 2011)

Cost and Licensing: Free download from Microsoft site.

System Requirements (minimum): Graphics Card Shader Model 1.1 support, DirectX 9.0 support, Operating System: Windows Vista SP2, Windows 7 (All editions except Starter).

Platforms: Windows, Xbox 360, Zune.

Overview and Features: XNA Game Studio 2.0 – application framework, integrated development environment. Features: Game component models, New framework library designed to support Microsoft Windows, XBOX 360, and Zune game development, Integration with XNA Framework Content Pipeline.

From an analysis of the capabilities of the video game development tools described, it can be concluded that they are all quite powerful. The choice of a specific tool is determined by the characteristics of the project being developed. Their use for educational purposes is almost equal, although the choice may be influenced by the size of the proposed course.

The second parameter to choose the instrument was its cost. All the tools described are free of charge for educational purposes and thus meet our needs.

The third, perhaps most essential, requirement is compliance with the minimum system requirements of the equipment and associated software. State educational institutions are at a disadvantage in this respect. Therefore, for the first version of the course “Computer game development” in our university was chosen Microsoft XNA Game Studio, which has a narrower range of possibilities.

We assumed that the experience of our students in C/C++ and C# programming would allow them to easily learn XNA. However, we were wrong. By the end of the course, many of them were halfway to the games. The greatest success was achieved by the group of students who developed the Tower Defence class game, but it was completed as part of the bachelor’s qualification work.

The problem with this approach is that in order for students to feel the process of developing games, they need an environment that they can easily use to create games. The focus of the course was to make the game good, not just work at all. We wanted our students to have experience working with a real engine, real skills if they decided to develop games.

The situation improved after the computers at our university were upgraded. We were able to work with a serious game engine. We decided to use the Unity

Table 1: Course part 1. Basics of work in Unity.

Topics	Duration, hours	Course materials	Deliverables
1. Introduction to Unity	3	Unity features. Examples of games created on Unity. Unity installation. The difference between 2d and 3d design. Overview of the main elements of the scene: Camera, GameObject, Direction Light. Moving the scene. Camera object. Location of objects on a 3d scene.	Laboratory work 1
2. Textures, materials and elements of the scene	3	Adding new textures to the project. Creation and use of materials. Shaders and their use. Work with aggregated characters and their components. Creating a Terrain. Terrain Landscape Editor. Trees, grass and surroundings. Placement of a player on Terrain.	Laboratory work 2
3. Scripts and object movement	3	Install Visual Studio Plug-in for Unity3d. Creating scripts. Apply a script to an object on the stage. The structure of the automatically generated script. Creating a character movement using a script.	Laboratory work 3
4. Player management	3	Using the Asset store. Download unitypackage. Use ready-made unitypackage. Creating unitypackage. The structure of projects created by other developers. Use of ready-made asset. Character Controller and its application. Move the object with the keyboard. Dynamic object creation.	Laboratory work 4
5. User interface	3	User interface and its application. Examples of basic controls. Bindings and orientation of controls relative to the working area of the screen. Creating elementary events. Customize Canvas to different screen resolution properties	Laboratory work 5
6. Animation	3	Using ready-made character animations. Create your own animation. Editing curves. Structure and main properties of the Animation component. Animator component	Laboratory work 6

Table 2: Course part 2. Game development based on Unity.

Topics	Duration, hours	Course materials	Deliverables
1. Game Development Basics	1	Game development life-cycle. Game terminology. Overview of game industry	Game Concept plan
2. Creating a character	3	Uploading models to the project. Features of creating game characters. Customize avatars for models that use humanoid animations. Working with the Animator component. Animator controller settings. Retargeting of humanoid animated clips.	Characters modelling and animation
3. Finding a way	3	Creating a game scene. Navigation grid settings. Add and adjust obstacles. Implementation of the movement of the character on the navigation grid.	Group projects element
4. Inverse kinematics	5	Animation settings. Attaching skeletal parts to objects. Creating a script to work with inverse kinematics. Fixation of skeleton points. LineRender component.	Group projects element
5. Characters not controlled by the player	6	Creating a slider and stylizing it. Move the coordinates of the slider to the position above the target. Creating goal health scripts. Using Raycast.	Group projects element
6. Construction of game levels	12	Creating a game level. Overlay post effects on the main camera. Set up bots to search for enemies. Game level layout. Creating multiple teams. Configuration and error correction. Possibility of application of scattering of bullets at shooting.	Final Game

Engine because it has a less steep learning curve than Unreal. It can be used to develop games for any platform, including the Web, for real games, not just training games for learning. Unity scripting can be

done in C# or JavaScript, with which our students have already had experience.

Table 3: Useful course element percentages, mean, and standard deviation.

Useful elements	1	2	3	4	5	M	SD
The way in which the material was approached	0	5	40	32,5	22,5	3,73	0,88
The pace at which we worked	2,5	10	45	30	12,5	3,4	0,93
Working with peers inside and outside of class	0	7,5	22,5	42,5	27,5	3,9	0,9
Viber discussion group	2,5	7,5	32,5	40	17,5	3,63	0,95
Teamwork in labs	0	2,5	15	57,5	25	4,05	0,71
The presentation of the final group project	0	10	20	45	25	3,85	0,92
The hands-on labs activities	0	5	25	32,5	37,5	4,03	0,92

Table 4: Student learning gains percentages, and mean.

Student gains from course	1	2	3	4	5	M	SD
Understanding the main concepts in game development	0	5	25	42,5	27,5	3,93	0,86
Understanding the game development process	2,5	2,5	10	60	25	4,03	0,83
Understanding Unity Engine using in game development	0	5	15	37,5	42,5	4,18	0,87
Ability to think through a problems in game development	0	1	10	20	9	3,93	0,76
Confidence in your ability to work in game development	0	5	25	47,5	22,5	3,88	0,82
Feeling comfortable with complex game development	0	2,5	35	45	17,5	3,78	0,77

4 ORGANIZATION OF THE COURSE

We wanted to build the course in such a way that students could learn the basics of Unity quickly enough and focus on creating the game for most of the semester.

After studying Paul E. Dickson's works (Dickson, 2015; Dickson et al., 2017), our first thought was to build a course based on a book with examples that could guide both us and our students, for example, Unity 3.x Game Development Essentials (Goldstone, 2009). One game is built throughout the book, each chapter introduces a new concept and aspect of the game. All examples of code are written in JavaScript and C#. This book quickly gives an idea of colliders, particle systems, etc. for anyone with no experience in game development. The work on the book provides enough information to study the basics of Unity.

One of the problems is the rapid development of Unity and the need to find relevant materials for work. Unity has an active online community that helps to find textbooks to cope with the new features and changes in Unity and could base the course on one of the online textbook series. However, since the duration of the course was only one semester, it was necessary to develop a manual sufficient to carry out the laboratory tasks in order to use the books only as an additional source of information.

Our goal in this course is to give students a sense of the game development process with a focus on project management, teamwork, and problem solving. The first part of the course focuses on teaching

students to use Unity, and the second part focuses on developing real play by groups of students. Classes were held for 3 hours per week: 1 hour of lectures and 2 hours of laboratory work. The basic structure of the course is shown in table 1, 2.

The method that we used in the first part of the course, to organize the study of Unity students, was to combine work on the assignments in the classroom with the performance of additional creative tasks by ourselves. In each work, students had to understand in detail what had been done in the classroom in order to determine how to complete the extra assignment. During the first part of the semester, students sought to learn how to solve various problems with Unity before they began working on their final game projects that required these skills. During this work, students built a basic game in which the player could control the movement and actions of the character in their environment.

5 RESULTS

It's hard to measure success when students are building different games. By calling the game playable, we mean that the students have created a mechanic for the game (possibly with minor errors), combined the art assets with the mechanics and made some introduction (history, list of game items) that enters into the game. In order to evaluate the results of our course "Computer game development" we used some parameters offered by Ritzhaupt (Ritzhaupt, 2009) to evaluate its such course.

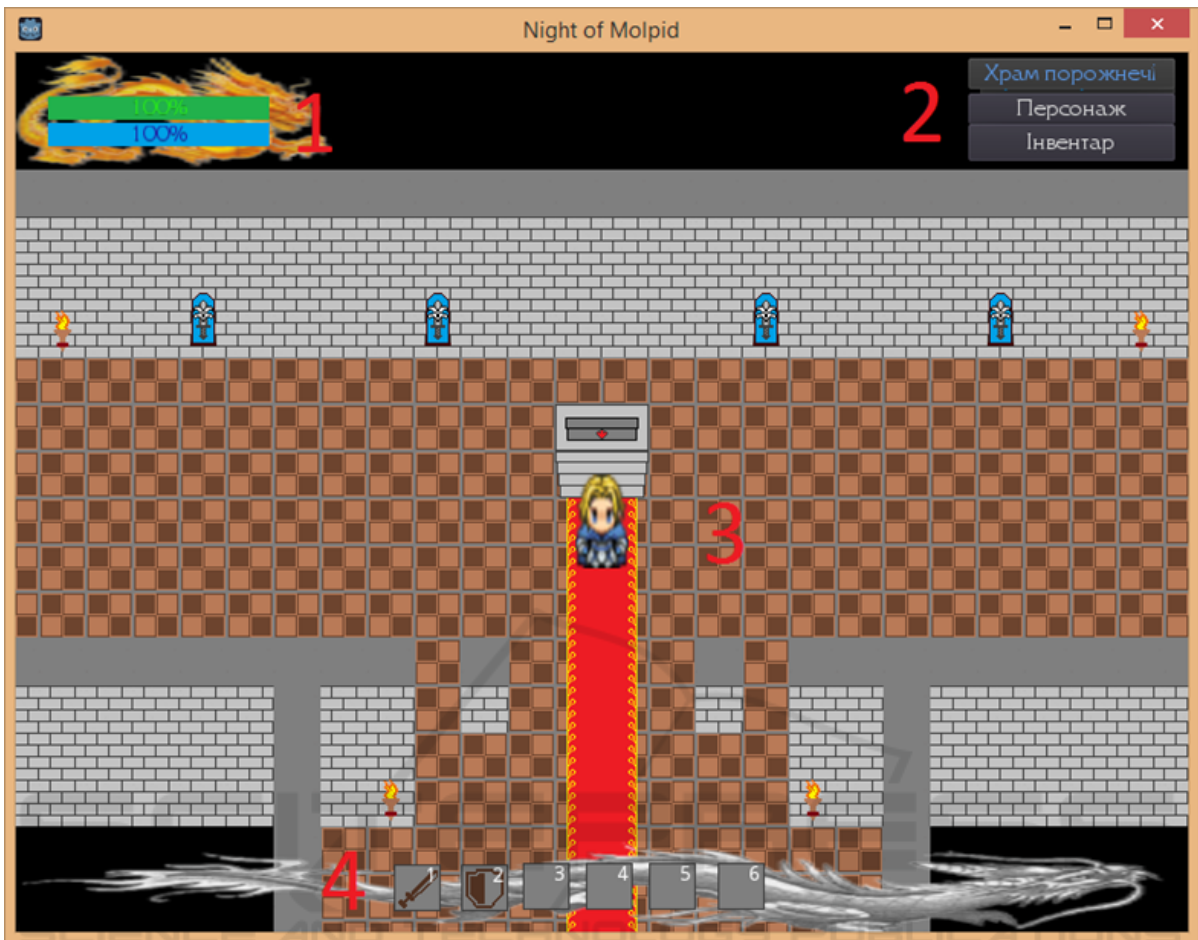


Figure 2: RPG game.



Figure 3: Quest game.

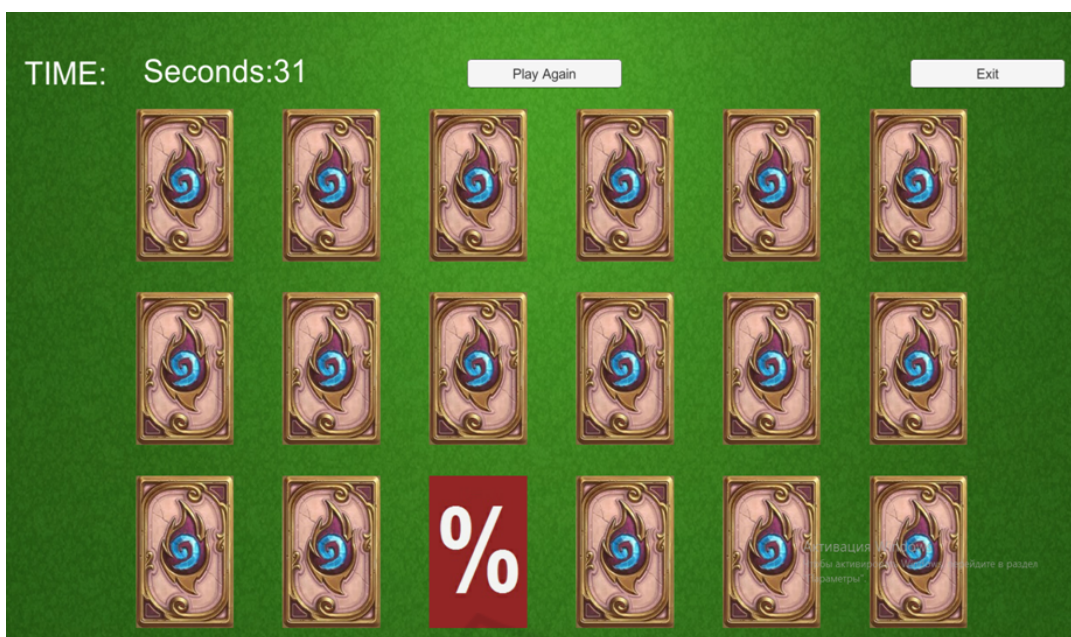


Figure 4: Logical game.

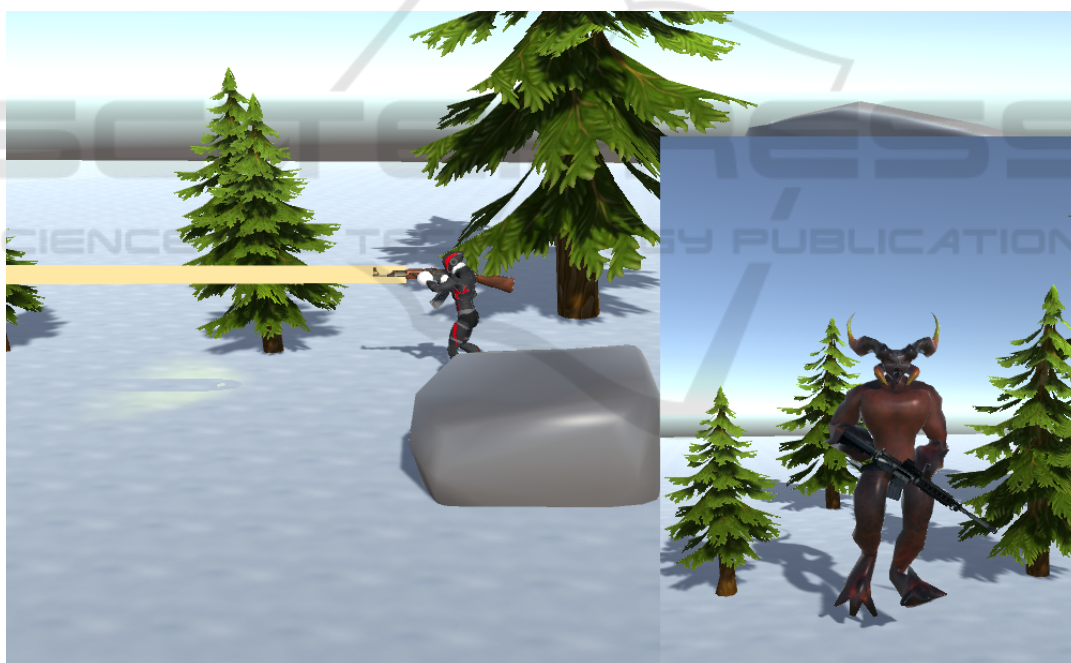


Figure 5: Action game.

5.1 Usefulness of Course Elements for Students

For studying the elements of the course that proved successful, we asked the students to indicate which elements of the course were useful for learning in the range from 1 – “not useful” to 5 – “very

useful” (table 3). Of particular interest are the highly rated elements: teamwork in labs ($M = 4.05$; $SD = 0.71$), working with peers inside and outside of class ($M = 3.9$; $SD = 0.9$), and the hands-on labs activities ($M = 4.03$; $SD = 0.92$). These results underline the importance of sufficient work in the computer laboratory and cooperative training in the game development course.

5.2 Student Assessment of Gains

Students were asked to evaluate their post-graduate achievements in a number of areas related to the development of games on a scale of 1 to 5 (table 4). The results showed that they made the most progress in understanding the game's development ($M = 4.03$; $SD = 0.83$) and the ability to use the Unity Engine ($M = 4.18$; $SD = 0.87$). In all other areas, progress has also been above average.

5.3 Final Project Game

In the second part of the course, students worked in groups (3–4) to create final game projects. We allow students to decide for themselves which games they want to develop and how to split into groups. Each group decided who would play what roles and what they would need to do to finish the game. Lectures on this part of the course covered a wide range of topics. Some specific aspects of game development that students are likely to need were discussed. All practical tasks for this part of the course are related to keeping students on their way to finishing the final project games. These include students presenting game ideas, project plans, vertical slices, usability tests, a final game, and weekly reports on who has achieved what.

Most of the groups were able to successfully build a playable game for the final project, which is significantly better than the previous version of the course. Students created RPG games (figure 2), quest games (figure 3), logical games (figure 4) and action games (figure 5). The variety of these games shows that students are free to create games of their choice instead of being limited to the genre and content given by the teacher.

6 CONCLUSIONS

This version of the course made more progress than the one based on the XNA Game Studio, even though students had to learn to use Unity in a short time. The fact that the game engine incorporates everything necessary to connect the player, the art and the surroundings greatly influenced what students could achieve.

Although, in order to keep up-to-date with video game development, it makes sense to focus on game development rather than learning the tool, implementation in the form of a set of courses.

The video game development course can be based on the Unity game engine, as it has a small entry threshold, free of charge for academic purposes, a cross-platform, real game engine common in the

game development industry. Based on our experience, Unity is a good option to teach a separate game development course. Since Unity is easy to learn, students can get it in less than half a semester, which leaves more than half a semester to focus on any topic you consider important (game design, project management, etc.).

Another result of this analysis is the effectiveness of cooperative learning and teamwork strategies, especially in computer laboratories.

One of the most important conclusions we have reached is that learning to design games requires a change of perspective from the teacher-centred environment in which the teacher dictates the learner's learning experience (e.g., leadership from outside (King, 1993)). A learning environment in which students have greater control over what they learn and teachers serve as facilitators in this process.

REFERENCES

- Adams, E. (2013). *Fundamentals of game design*. New Riders, 4rd edition.
- Ahlquist, J. B. and Novak, J. (2007). *Game Development Essentials: Game Artificial Intelligence*. Cengage Learning.
- Axon, S. (2016). Unity at 10: For better—or worse—game development has never been easier. <https://arstechnica.com/gaming/2016/09/unity-at-10-for-better-or-worse-game-development-has-never-been-easier/>.
- Barnes, T., Richter, H., Powell, E., Chaffin, A., and Godwin, A. (2007). Game2learn: Building cs1 learning games for retention. *ACM SIGCSE Bulletin*, 39(3):121–125.
- Batchelor, J. (2020). GamesIndustry.biz presents... The Year in Numbers 2020. <https://www.gamesindustry.biz/articles/2020-12-21-gamesindustry-biz-presents-the-year-in-numbers-2020>.
- Becker, K. and Parker, J. R. (2007). Serious Games + Computer Science = Serious CS. *Journal of Computing Sciences in Colleges*, 23(2):40–46.
- Bond, J. G. (2014). *Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C#*. Addison-Wesley Professional.
- Bradfield, C. (2018). *Godot Engine Game Development Projects: Build five cross-platform 2D and 3D games with Godot 3.0*. Packt Publishing, Birmingham.
- Brown, Q., Lee, F., and Alexandre, S. (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*, FDG '09, page 240–247, New York, NY, USA. Association for Computing Machinery.

- Clark, B., Rosenberg, J., Smith, T., Steiner, S., Wallace, S., and Orr, G. (2007). Game development courses in the computer science curriculum. *Journal of Computing Sciences in Colleges*, 23(2):65–66.
- Claypool, K. and Claypool, M. (2005). Teaching software engineering through game design. *ACM SIGCSE Bulletin*, 37(3):123–127.
- Coleman, R., Krembs, M., Labouseur, A., and Weir, J. (2005). Game design & programming concentration within the computer science curriculum. *ACM SIGCSE Bulletin*, 37(1):545–550.
- Comber, O., Motschnig, R., Mayer, H., and Haselberger, D. (2019). Engaging students in computer science education through game development with Unity. In *2019 IEEE Global Engineering Education Conference (EDUCON)*. IEEE.
- de Macedo, D. V. and Rodrigues, M. A. F. (2011). Experiences with rapid mobile game development using Unity engine. *Computers in Entertainment*, 9(3):1–12.
- Dickson, P. E. (2015). Using Unity to teach game development. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*. ACM.
- Dickson, P. E., Block, J. E., Echevarria, G. N., and Keenan, K. C. (2017). An experience-based comparison of Unity and Unreal for a stand-alone 3D game development course. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*. ACM.
- Distasio, J. and Way, T. (2007). Inclusive computer science education using a ready-made computer game framework. In *Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education - ITiCSE'07*. ACM Press.
- Estey, A., Long, J., Gooch, B., and Gooch, A. A. (2010). Investigating studio-based learning in a course on game design. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games - FDG'10*. ACM Press.
- Fachada, N. and Códices, N. (2020). Top-down design of a CS curriculum for a computer games BA. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. ACM.
- Goldstone, W. (2009). *Unity game development essentials*. Packt Publishing Ltd.
- Haranin, O. M., Katsko, O. O., and Moiseienko, N. V. (2017). Developer software tools in a course “Development of computer games”. *New computer technology*, 15:160–163.
- Haranin, O. M. and Moiseienko, N. V. (2018). Adaptive artificial intelligence in RPG-game on the Unity game engine. *CEUR Workshop Proceedings*, 2292:143–150.
- Harris, J. (2011). Teaching Game Programming Using XNA: What Works and What Doesn't. *Journal of Computing Sciences in Colleges*, 27(2):174–181.
- Jones, R. M. (2000). Design and implementation of computer games. *ACM SIGCSE Bulletin*, 32(1):260–264.
- Katsko, O. O. and Moiseienko, N. V. (2018). Development computer games on the Unity game engine for research of elements of the cognitive thinking in the playing process. *CEUR Workshop Proceedings*, 2292:151–155.
- King, A. (1993). From sage on the stage to guide on the side. *College Teaching*, 41(1):30–35.
- Linhoff, J. and Settle, A. (2008). Teaching game programming using XNA. In *Proceedings of the 13th annual conference on Innovation and technology in computer science education - ITiCSE'08*. ACM Press.
- Manzur, A. and Marques, G. (2018). *Godot Engine Game Development in 24 Hours*. Sams Publishing, Indianapolis.
- Martin, J. and Smith, C. (2002). A cross-curricular team based approach to game development. *Journal of Computing Sciences in Colleges*, 17(5):39–45.
- Miles, R. (2011). *Microsoft XNA Game Studio 4.0: Learn Programming Now!* Pearson Education.
- Millington, I. (2019). *AI for Games*. CRC Press.
- Morrison, B. B. and Preston, J. A. (2009). Engagement. *ACM SIGCSE Bulletin*, 41(1):342–346.
- Parberry, I., Kazemzadeh, M. B., and Roden, T. (2006). The art and science of game programming. In *Proceedings of the 37th SIGCSE technical symposium on Computer science education - SIGCSE'06*. ACM Press.
- Parberry, I., Roden, T., and Kazemzadeh, M. B. (2005). Experience with an industry-driven capstone course on game programming. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education - SIGCSE'05*. ACM Press.
- Peng, C. (2015). Introductory game development course: A mix of programming and art. In *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE.
- Prokhorov, O. V., Lisovichenko, V. O., Mazorchuk, M. S., and Kuzminska, O. H. (2021). Digital technology implementation for students' involvement base on 3D quest game for career guidance and estimating students' digital competences. *CEUR Workshop Proceedings*.
- Rankin, Y., Gooch, A., and Gooch, B. (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 - GDCSE'08*. ACM Press.
- Ritzhaupt, A. D. (2009). Creating a game development course with limited resources. *ACM Transactions on Computing Education*, 9(1):1–16.
- Rocco, D. and Yoder, D. (2007). Design of a media and gaming sequence for graduates in applied CS. *Journal of Computing Sciences in Colleges*, 22(5):131–137.
- Roden, T. E. and LeGrand, R. (2013). Growing a computer science program with a focus on game development. In *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE'13*. ACM Press.
- Saulter, J. (2007). *Introduction to video game design and development*. McGraw-Hill, New York.

- Shultz, G. A. (2004). The story engine concept in CS education. *Journal of Computing Sciences in Colleges*, 20(1):241–247.
- Sung, K. (2009). Computer games and traditional CS courses. *Communications of the ACM*, 52(12):74–78.
- Sweedyk, E. and Keller, R. M. (2005). Fun and games. *ACM SIGCSE Bulletin*, 37(3):138–142.
- Takahashi, D. (2018). John riccitiello q&a: How unity ceo views epic’s fortnite success. <https://venturebeat.com/2018/09/15/john-riccitiello-interview-how-unity-ceo-views-epics-fortnite-success/>.
- Toftedahl, M. and Engström, H. (2019). A taxonomy of game engines and the tools that drive the industry. In *DiGRA 2019, The 12th Digital Games Research Association Conference, Kyoto, Japan, August, 6-10, 2019*. Digital Games Research Association (DiGRA), DiGRA. http://www.digra.org/wp-content/uploads/digital-library/DiGRA_2019_paper_164.pdf.
- Unity Technologies (2021). Unity Real-Time Development Platform — 3D, 2D VR & AR Engine. <https://unity.com>.
- Unreal Engine (2021). The most powerful real-time 3D creation platform. <https://www.unrealengine.com>.
- Vakaliuk, T., Kontsedailo, V., Antoniuk, D., Korotun, O., Semerikov, S., and Mintii, I. (2020a). Using Game Dev Tycoon to develop professional soft competencies for future engineers-programmers. *CEUR Workshop Proceedings*, 2732:808–822.
- Vakaliuk, T. A., Kontsedailo, V. V., Antoniuk, D. S., Korotun, O. V., Mintii, I. S., and Pikilnyak, A. V. (2020b). Using game simulator Software Inc in the Software Engineering education. *CEUR Workshop Proceedings*, 2547:66–80.
- Vanhatupa, J.-M. (2011). Game engines in game programming education. In *Proceedings of the 11th Koli Calling International Conference on Computing Education Research - Koli Calling'11*. ACM Press.
- Werner, L., Campe, S., and Denner, J. (2012). Children learning computer science concepts via Alice game-programming. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE'12*. ACM Press.
- Yannakakis, G. N. and Togelius, J. (2018). *Artificial Intelligence and Games*. Springer International Publishing.