Health Informatics Meets eHealth
G. Schreier et al. (Eds.)

doi:10.3233/978-1-61499-645-3-207

Evaluation of Game Engines for Cross-Platform Development of Mobile Serious Games for Health

Carina KLEINSCHMIDT[a,1] and Martin HAAG[a]
[a]Hochschule Heilbronn, GECKO Institut, Germany

Abstract. Studies have shown that serious games for health can improve patient compliance and help to increase the quality of medical education. Due to a growing availability of mobile devices, especially the development of cross-platform mobile apps is helpful for improving healthcare. As the development can be highly time-consuming and expensive, an alternative development process is needed. Game engines are expected to simplify this process. Therefore, this article examines the question whether using game engines for cross-platform serious games for health can simplify the development compared to the development of a plain HTML5 app. At first, a systematic review of the literature was conducted in different databases (MEDLINE, ACM and IEEE). Afterwards three different game engines were chosen, evaluated in different categories and compared to the development of a HTML5 app. This was realized by implementing a prototypical application in the different engines and conducting a utility analysis. The evaluation shows that the Marmalade engine is the best choice for development in this scenario. Furthermore, it is obvious that the game engines have great benefits against plain HTML5 development as they provide components for graphics, physics, sounds, etc. The authors recommend to use the Marmalade Engine for a cross-platform mobile Serious Game for Health.

Keywords. Mobile application, video games, education, medical.

1. Introduction

Studies have shown that Serious Games for Health can improve patient compliance and the chance of success of therapies [1,2]. Another application for Serious Games for Health is the field of medical education. As for example Boeker et al. (2009) [3] have shown with the game "Uro Island", it is possible to support the students' learning difficult medical subjects by wrapping the content in a Serious Game. Therefore, Serious Games for Health are a valuable contribution for improving healthcare. With the growing availability of mobile devices, especially Serious Games for Health as mobile apps are very useful.

The quality of the graphical UI plays an important role for the success of a Serious Game [4]. As many games for entertainment have highly realistic graphics, especially children and young adults have developed high demands in games graphics. Developing Serious Games with such good graphics can be very expensive and time-consuming, which is not acceptable especially in the healthcare sector. Game Engines are expected to solve this problem of development costs and time commitment.

[1] Corresponding Author: C. Kleinschmidt, Hochschule Heilbronn, Max-Planck-Str.39, 74081 Heilbronn, Germany. E-Mail: carina.kleinschmidt@hs-heilbronn.de.

Game Engines are data-driven architecture software pieces which contain certain codes needed for playing games and which can be reused for developing other games. Such codes consist for example of logic for collisions, physics and especially graphics [5]. For developing other games they supply frameworks, programming environments and tools. This article will evaluate if these Game Engines help to facilitate the development and to raise the quality of Serious Games for medical education and which Game Engine therefore provides the most benefit.

2. Methods

At first a systematic research for literature was conducted in different databases (MEDLINE, ACM, IEEE). Several publications about Serious Games for Health were found, but only a few mention the use of game engines [6] and only some of them what kind of game engine they used [7,8]. One publication aimed at a similar examination. Marks et al. (2007) [9] explored the use of game engines for simulated surgical training. However, this scenario is very specific and not applicable in general for Serious Games for Health.

Afterwards the scenario was determined for which the evaluation will be made. As Boeker et al. (2009) [3] found out that game-based e-learning in medical education is more effective than conventional methods, the scenario will contain the development of a Serious Game for medical education. To gain a larger target group of students who have many different mobile devices the game should be available for Android, iOS and different browsers. Therefore, the engines need to provide cross-platform development. To fully exploit the development process with a game engine a prototype will be implemented. For the graphics of this prototype, the images for the scene were designed with Inkscape [10]. It was intended to design an erythrocyte that swims in a blood vessel and has to avoid the collision with some “bad particles” that are represented by black squares. This idea derives from Barbosa and Silva (2011) [11]. They created a Serious Game called “OxyBlood” with the aim to teach young students the basic functioning of the circulatory system. A simplified version of this game without any medical knowledge covers the basic technical functions that should be evaluated with this prototype. It is desired that the erythrocyte can be controlled with the arrow keys on a PC and on a mobile device by touching the screen to point the direction, e.g. at the top of the screen for swimming up. This will be implemented as a sidescroller, which means the figure can only move left or right along the x-axis of a Cartesian coordinate system. The figure needs also to follow physical laws and a collision detection should be implemented for the “bad particles”.

The further evaluation beyond implementing will be examining various aspects ranging from costs, license and installation, to the aspects of development and deployment to different target platforms. Therefore, a utility analysis was conducted. This method belongs to decision theory and helps to quantify non-quantifiable aspects. [12]. Every aspect is furnished with an importance rating from 1-5, to consider major aspects more than others. Every engine receives points from 1-10 for every aspect. For deciding when to give the full points, a fulfillment norm is defined for every aspect. These will be multiplied with the importance rating and summed up, so that a clear result for every engine and the HTML5 development will be provided and compared. In **Table 1** the evaluation aspects and their importance rating can be seen. These aspects were partial derived from the process of cross-platform development and partial completed by

aspects from general software-tests of relevant magazines [13]. This is only an excerpt of the complete evaluation that can not be provided in this paper due to the lack of space. The complete evaluation including explanation of every aspect, the fulfilling norm and the definition of point's assignment can be found under this address [14].

Table 1: Excerpt of the evaluation: the aspects and their importance rating

Aspect	Cost	License	Expendable Functionality	Fulfills Scenario Needs	Problems with Installation
Rating	5	5	3	5	1
Aspect	Additional Tools for Inst. needed	Available OS	Clear Arrangement of the UI	First Orientation	Amount of Understanding needed
Rating	1	1	3	3	4
Aspect	Scripting Editor Provided	External graphic editor needed	Handling of the graphic editor	Support for Physic, collision detection and Audio available	Cross-Platform Testing
Rating	3	3	5	5	4
Aspect	Support for Installation of SDK's provided	Cross-Platform Deployment	Additional Software needed for deployment	Specialties of each Game Engine	
Rating	4	4	3	1	

As mentioned above, only a few publications reported whether or not they used a game engine and which one they used. For gaining a more complete overview about the available game engines, it was decided to start the selection process with a list of game engines from relevant resources from the Internet. For this article a list of game engines from the Wikipedia (2016) [15] was chosen, as this list contained 51 game engines at that time. This list was completed by two lists of relevant gaming magazines [16, 17]. Afterwards the list contained 79 game engines. As a next step, all engines that do not support cross-platform development for at least Android, iOS and common Browsers were excluded. Thus, the list was reduced to 14 game engines. Three engines were not available at this time (one will be released in spring 2016, the other two were stopped in development and are now no longer available). One game engine was removed from the list as the company does not provide maintenance anymore. With ten game engines left, a research in relevant gaming forums like for example the "unity-insider forum" [18] was conducted, to include experiences from users that are more skilled in this subject. Due to reasons of capacity it is not possible to evaluate all ten game engines left. As these ten game engines can be classified into three types of game engines, it was decided to examine one of each type to gain the best overview possible. One of these types are powerful 3D Game Engines that are mostly destined for developing 3D-First-Person-Shooter games to all kinds of target platforms. As these games require very realistic graphics, physics and sounds, the engines are very powerful. But it is also possible to develop a 2D mobile game with these engines. The second type that will be evaluated are game engines with focus on the development of 2D games. They do not provide the same amount of functionality like the first group of game engines. Instead they offer different functionalities to improve the comfort in developing. Additionally the third evaluated type are game engines without a graphical editor. These game engines provide a framework that contains functionality for supporting the developer in gameplay physics, collision detection, audio etc.. The usage of them is mostly provided via integration in other development technologies like for example in HTML5.

From these three types the game engines with the most recommendation from skilled users or magazines found are chosen for the evaluation. For the first type of powerful 3D game engines he engine of choice for this evaluation, will be "Unity" [18] as it is often recommended by experienced users to be the best choice for beginners and was mentioned in several publications [7,8] as the game engine in use. For the second type the "Marmalade" engine [19] is chosen, as it was especially recommended for developing mobile games. As an engine without a graphical editor the "Turbulenz HTML5 Game Engine" [20] was chosen. This game engine was recommended as it is fully open source and provides therefore high adaptability of the engine code.

3. Results

As mentioned in Chapter 2, the game engines Unity and Marmalade provide UI's for developing games. Whilst Unity comes with one UI that contains everything needed, like for example a graphic editor, a scripting editor and the possibility to simulate scenes, the Marmalade engine has several UI's for different purposes. The user starts with the so-called "hub". This management tool helps the user to dispense his projects, the dependencies, the different SDK's, the different simulators for testing, the iOS signing request and the IDE's. For developing in C++, Marmalade needs an IDE already installed, e.g. XCode for Mac. For developing in Lua, a powerful scripting language [21], Marmalade brings his own IDE. When starting a project, the hub opens the right IDE and uses the right SDK and simulators. For developing graphics Marmalade offers his own 2D graphics editor in a beta version. Scenes that were created with this editor can be imported to the Marmalade projects.

The Turbulenz HTML5 game engine is used in HTML5 projects by including some src-tags and then using the engine in the Javascript Code. Therefore, the engine provides no editor at all. The user decides which HTML5 editor he wants to use. The same goes with a plain HTML5 project.

When creating the scene for the prototype that was described in Chapter 2, first orientation was rather complicated in every engine. Unity and Marmalade provide several Documentations and Tutorials, as well as some test projects. Still it took time to understand different views and functions available, as they were not as self-explaining as expected. The Turbulenz engine provides a very detailed documentation on how to use the engine properly, but Tutorials were rare. It is expected, that the lack of Tutorials can be explained by the comparatively low distribution of the game engines without UI.

In Unity and Marmalade, the prepared images can simply be imported or dragged into the editor. They can as well be positioned by dragging them to the right position and alternatively by giving them numbers for the X and Y value. The objects are automatically declared as sprites. The creating of a scene in the Marmalade 2D Editor can be seen in Figure 1. This handling is very comfortable and saves a high amount of time versus the Turbulenz engine and the HTML5 project. When creating scenes, the images have to be positioned by tags. This can be very exhausting since after every change of the tags, the browser has to be reloaded in order to notice any changes.

As a next step, the scene had to be animated. Therefore, the erythrocyte is supposed to be able to move and the camera is supposed to focus on it when it moves. In Unity it is possible to use complete scripts that are provided by the engine. Marmalade provides functions for creating these animations. The usage of these both are quite easy to handle. The Turbulenz engine also provides functions and in HTML5 the users have to

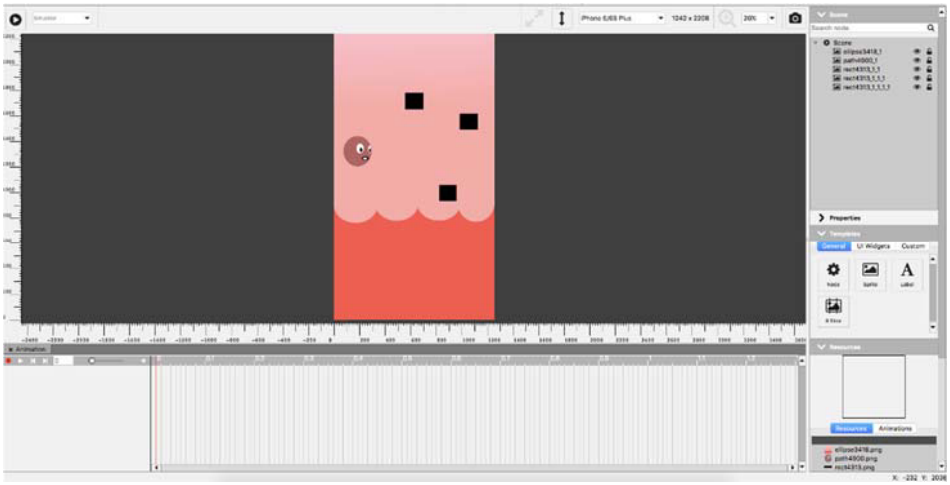

Figure 1: Creating a scene with the Marmalade 2D graphics editor

implement it all by themselves. Also in this point, the way of using them is not as comfortable as it is with the other two engines since for seeing any changes made in the code the browser has to be reloaded every time.

Apart from graphics, other important aspects of games that are not already implemented in the prototype, are physics and sounds. For realistic gameplay it is required that the objects follow physical laws like, for example, gravity, or that noise, e.g. rain, sounds realistic. All of these have to be implemented by the user, when creating a game in HTML5. Game engines normally provide these codes. In Unity this happens via characteristics of assets or via scripting code and is quite easy to handle. In Marmalade the objects can get the assignment of characteristics like gravity in the scripting code. The Turbulenz engine also provides the functionalities for physics and sounds. These have to be used in the scripting code.

When developing games cross-platform for mobile devices, a very important point is testing the game on different devices. This means not only the gameplay, but also the newly created UI, as every mobile device has a different format and the UI can appear different on each one of them. The Unity engine provides the simulation of the scene, but without considering different mobile devices. However, the user can adapt the camera view to a certain format e.g. the screen of an iPhone, but this will only adapt to one format and not to different formats of different devices. Testing on different devices can either be made with simulators or on real devices. Both of these require deployment to the target platform at first. This is also the way for testing games from the Turbulenz engine or HTML5. For testing in browsers, the Turbulenz engine comes with a testing environment that also contains a local server. For testing on mobile devices, a game has to be deployed to the target platform and then be tested either in a simulator or on a mobile device.

The Marmalade engine provides two kinds of simulations. In the 2D graphic editor, a simulator is provided that can simulate the appearance of the scene on common Android and iOS devices. In Figure 2, the current prototype scene is simulated on an iPhone 4/4S with the 2D editor. The other simulation is provided when scripting the gameplay. Here Marmalade offers the possibility not only to simulate the gameplay in the scene on the computer but also to use simulators for mobile devices. As mentioned

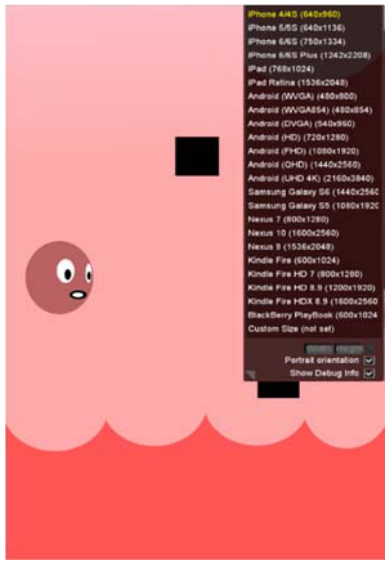

Figure 2: Simulating the scene with the Marmalade 2D graphics editor on iPhone 4/4S

before, in Marmalade the hub manages all projects. This includes also these simulators. If there are simulators installed on the PC the user can add them to the hub by pointing on their directory. Like this, the hub can use the simulator when running a project in the IDE. This is very practical and saves time as testing can be done before deploying and different formats are tested all by once.

In addition the deployment to different target platforms like iOS, Android and browsers can be very time-consuming. Marmalade and Unity support the user in this step, as the user can simply point the engines to the directory of the installed SDK's. The deployment afterwards is done by the engines. The Turbulenz engine does not provide any comfort in this step. Just as projects from plain HTML5, projects from the Turbulenz engine have to be deployed by the user with additional tools.

Finally, the results of the evaluation show a high benefit from game engines in general compared to the development in plain HTML5. Table 2 shows the overall points of each game engine and HTML5. The complete evaluation is provided under this address [14].

These results show that each game engine could reach more points than the development with HTML5. This is because HTML5 does not provide any support for the user for graphics, physics and sound components. Therefore, the game engines support the development process significantly. Additionally, the evaluation shows that game engines with graphical UI reached significantly more points than an engine without one. The graphic editor showed significant benefits while implementing, therefore the Marmalade and the Unity engine gained more points. When looking at the aspect of provides support for a great number of different mobile target devices and helps to manage the projects for them. It supports in deploying one project on different devices

Table 2: The results of the evaluation.

Game Engine	Marmalade	Unity	Turbulenz	HTML5
Total Points	585	550	402	329

cross-platform development, the engine of choice is the Marmalade engine. This engine and provides simulation of scene and gameplay on them. Therefore, as the aim of this evaluation was support in cross-platform development, the Marmalade engine got the most points in the evaluation and is the best choice for developing a cross-platform mobile Serious Game for Health.

4. Discussion

The evaluation shows that the development of games highly benefits from using game engines especially in the fields of graphics, physics and sounds. The game engines partial reached three times (Unity, Marmalade) the points of the development with HTML5. This can clearly be interpreted as major benefits from the game engines.

Also the cross-platform development can be highly supported by game engines. The evaluation aspects "cross-platform testing" and "cross-platform deployment" considered this mainly. The engine that provides the most support for this development is the Marmalade Engine that reached the most points in the complete evaluation as well (550). The engine supports simulation on different devices, management of projects, simulators, SDK's etc. This functionality, provided by the hub, can save a great amount of time, as testing and deploying do not have to be made for each device on its own. In addition, the simulation of the scene in the 2D-editor is very helpful to test the graphics on different screen sizes immediately.

The Unity engine also supports cross-platform development and reached the second most points (516). The game can be deployed to different target platforms, but as the simulation of the scene on a mobile device can only be done after deployment, the Marmalade engine is the better choice here. However, since the Unity engine is very powerful in 3D graphics, it may be a good choice when using 3D graphics.

The Turbulenz engine is not as comfortable to handle as the other engines and reached only the third most points (397). The main reason is the absence of a graphic editor, but as well to the lack of testing. The latter can be achieved in a provided testing environment with a local server, which is not as comfortable and effective as testing directly in the devices formats. The game has to be deployed as well first for being tested in the right simulator.

Therefore, the recommendation out of this evaluation is to use the Marmalade Engine for cross-platform development of Serious Games for health. As the market of game engines is highly dynamic it is definitely needed to repeat this evaluation in the future. If the capacity is provided it is also possible to evaluate other game engines that come into consideration.

The evaluation can also be expanded to more engines when the scenario of target platforms is changed. As there are many game engines available that can be used for developing for iOS and Android but not for browsers, the list of engines would be larger if browser would be taken off the scenario. However, this is not practical for a Serious Game for medical education, as it cannot be assumed for reasons of equalization that every student has access to mobile devices.

References

[1] Kato, P., Cole, S., Bradlyn, A., Pollock, B., A Video Game Improves Behavioral Outcomes in Adolescents and Young Adults with Cancer: A Randomized Trial, PEDIATRICS 122(2) (2008), 305-317.
[2] Bartolomé, N., Zorilla, A., Zapirain, B., Can Game-Based Therapies be trusted? Is Game-Based Education effective? A systematic review of the Serious Games for Health and Education, 16th International Conference on Computer Games (CGames) (2011), 275-282.
[3] Boeker, M., Andel, P., Seidl, M., Streicher, A., Schneevoigt, T., Dem, P., Frankenschmidt, A., Uro Islands I – Game-based E-learning in der Urologie, GMS Med Inform Biom und Epidemiol 5(1) (2009), Doc03.
[4] Ushaw, G., Davison, R., Eyre, J., Morgan, G., Adopting Best Pracitces from the Game Industry in Development of Serious Games for Health, Proceedings oft he 5th International Conference on Digital Health (2015), 1-8.
[5] Gregory, J, Game Engine Architecture, CRC Press, Boca Raton, 2014.
[6] Hassan, M., Hossain, M., Alamri, A., Hossain, M., A., Al-Qurishi, M., Aldukhayyil, Y., Ahmed, D., A cloud-based serious games framework for obesity, Proceedings oft he 1st ACM multimedia international workshop on Cloud-based multimedia applications and services for e-health(2012), 15-20
[7] Porcino, T., Strauss, E., Clua, E., Hugo against dengue: a serious game to educate people about dengue fever prevention, 2014 IEEE 3rd International Conference on Serious Games and Applications for Health (SeGAH) (2014), 1-5.
[8] Assigana, E., Chang, E., Cho, S., Kotecha, V., Liu B., Turner, H., Zhang, Y., Christel, M., Stevens, S., TF-CBT triangle of life: a game to help with cognitive behavioral therapy, Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play (2014), 9-16.
[9] Marks, S., Windsor, J., Wünsche, B., Evaluation of game engines for simulated surgical training, 5th International Conference on Computer Graphics and Interactive Technique (GRAPHITE) (2007), 273-282.
[10] Inkscape Community, Inkscape, https://inkscape.org/de/, last access: 19.02.2016.
[11] Barbosa, A.F.S., Silva, F.G.M. (2011): Serious Games – Design and Development of OxyBlood, Proceedings oft he 8th International Conference on Advances in Computer Enterteinment, Technologie (2011), 1-8.
[12] Keeney, R.L., Raiffa, H. (1976): Decisions with Multiple Objectives, Preferences and Value Tradeoffs, Camebridge, Camebridge University Press.
[13] Computer-Bild Spiele, http://www.computerbild.de/cbs/, last access:07.03.2016.
[14] Kleinschmidt, C., Haag, M. , Evaluation von Game Engines für die Cross-Plattform Entwicklung von Serious Games for Health: Detailübersicht, www.virtuellepatienten.de/Evaluation_Game_Engines.html, last access: 13.03.2016.
[15] Wikipedia, Liste von Spiele-Engines, https://de.wikipedia.org/wiki/Liste_von_Spiel-Engines, last access: 14.01.2016.
[16] Develop, The top 16 game engines for 2014, http://www.develop-online.net/tools-and-tech/the-top-16-game-engines-for-2014/0192302, last access: 11.01.2016.
[17] Appindex, The Big List of Mobile Game Development Tools, Engines, Guides and Resources, http://appindex.com/blog/big-list-mobile-game-development-tools-engines-guides-resources/, last access: 11.01.2016.
[18] Unity insider forum, http://forum.unity-community.de/, last access: 11.01.2016.
[19] Unity Technologies, Unity Engine, https://unity3d.com/, last access: 11.01.2016.
[20] Marmalade Technoligies Ltd., The Marmalade Engine, https://www.madewithmarmalade.com/, last access: 11.01.2016.
[21] Turbulenz Limited, The Turbulenz HTML5 Game Engine, http://biz.turbulenz.com/, last access: 11.01.2016.
[22] PUC RIO, About Lua, http://www.lua.org/about.html, last access: 18.01.2016.