**Mobile games II report**

Jan Kalinowski p17179637

//github link

[Game video](https://www.youtube.com/watch?v=WD_OUUZBGvs&feature=youtu.be)



**Content**

1. Overview
2. Game idea
3. Development process
4. Mobile considerations
5. Resources
6. Testing
7. Learning outcome
8. References
9. Appendices

**Overview**

In term 2 of academic year 2018/2019 we have been asked to develop a game for mobile devices. This assessment was a follow up to the module Mobile Games I, which introduced mobile devices aspects, concerns and considerations developers have to bear in mind while developing games for this kind of devices.

This report justifies my decisions breaks down the whole development process.

**Game idea**

The only requirement for the game was it had to be 3D, but in terms of genre, gameplay and other design decisions we had to make we have been given free choice. Therefore I decided to go for 3D hack’n’slash type of game. The reason for that was I wanted to challenge myself to make something I haven’t done before.

I was inspired by other successful games in this competitive market, such as *Bladebound* and soon to be released *Diablo Immortal*. This was the perfect opportunity to develop something similar and try to compete with them.

The game is kept simple, where players have to defeat waves of enemies in order to finish the level. To defeat enemies players may use classic sword attacks or spells.



**Development process**

As it is known, mobile devices are handy and fast, but the price of a pocket size computer is its power. There are also many types of mobile devices, such as phones or tablets. There is several major producers, such as Samsung or Apple, and each of their device is slightly different. All of them have many constraints, which developers have to bear in mind in order to make game run smoothly on as many mobile devices as possible. My main target is Android, as it currently has 75% of the worldwide market (see appendix 1).

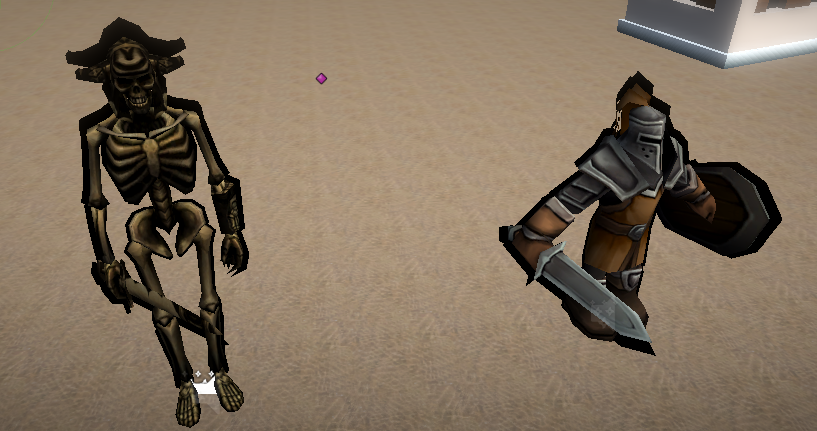
With that said, I can say, that while developing the game I was trying to keep mobile devices constraints in mind, thus I haven’t used any high resolution textures or 3D models, but I was trying to keep everything as ‘hardware cheap’ as possible, while keeping a fair amount of detail in my game.

** **

I have started with designing main goals of the game, as well as basic functionality, such as player movement and camera follow. I have thought about what content I want to include in the project and what do I want to achieve. I have also set myself deadlines, so that has allowed me finish the game right in time.

I am an author of almost all of the code, however, no graphics nor sounds have been made by me. To see where they have been downloaded from, please see file called ‘Resources’ attached in the main folder.

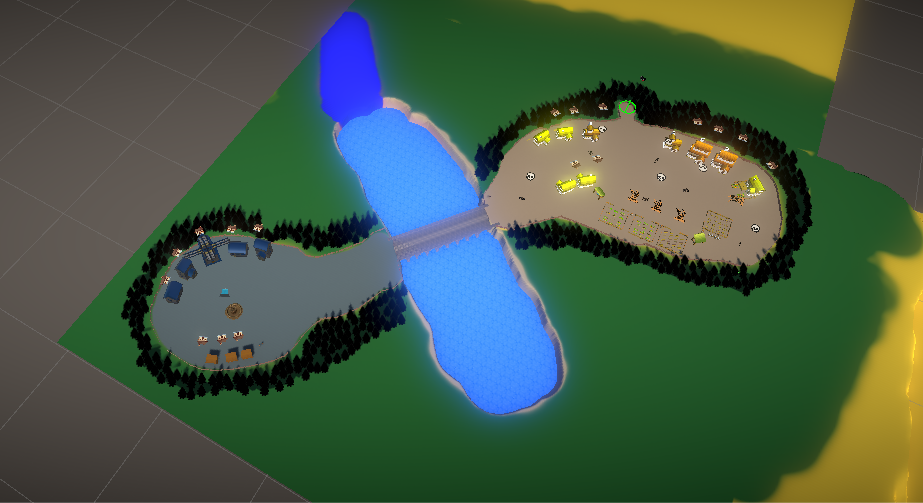
Over the entire development period I was trying to keep my code reusable and simple, yet using advanced techniques, such as coroutines (see appendix 3). Hence I have written my scripts once and was able to reuse them. I have reused my code in creating both of my enemies without altering a single line of code (see appendix 4).

****

The advanced techniques include, but are not limited to, use of coroutines or OnTriggerEnter/Exit functions (see appendix 5, so I could have kept my code efficient by calling the functions only when I need them to, rather than checking for events every frame in Update function.

Current flow of the level is as follows: players have to leave the starting area and defeat first wave of enemies in the Village. When all of them are defeated, the game unlocks second part of the map – the Town. Players need to go there across the bridge. In the town awaits second wave of enemies. When those enemies are defeated, the game loads main menu.

By dividing the level into two separate areas I was able to introduce a simple story, as the Town is initially blocked by an invisible wall, which is being disabled by clearing the Village.

Overall I have used many advanced Unity techniques, including, but not limited to, events handling and use of NavMeshAgents. I have also made my own outline shader and I made my scripts ‘talk’ to each other, so an event in one script can be triggered from another. This allowed me to introduce a simple, but complex level to my game, and a basis for future development of new mechanics.

**Mobile considerations**

As mentioned above, there is a huge range of mobile devices all over the world. They all may come with different specification, operating system version, power and size, so game developers have to bear all those aspects in mind.

Before developing my game I have done some research on smartphones specifications from 2013 (see appendix 6). As it is year 2019 I have assumed the processors are slightly better, however screen sizes have not changed much.

While developing *Paladin’s Tale* I have considered some of the mobile aspects, such as input handling and its power. Hence I have kept my UI to minimum, my models are ‘low poly’ in order to minimize light calculations and particle effects are the least hardware expensive as I could have found.



A good example of how have I been dealing with mobile features and how can I solve common design issues, is my target selection system – players can’t choose target manually, but it’s being done for them. The game chooses a target from enemies nearby the player. I have designed it this way due to constraint interactable area of the mobile phone and possible issues with manual choice of target.

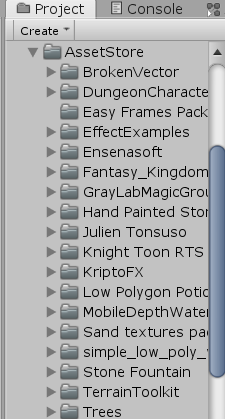


Due to the constraints I have also made 3 builds of the game – one with no postprocessing, one with half of desired postprocessing and the last one with full desired postprocessing (vignette, blur, enhanced light effects etc.) After some testing of how those affect game’s performance (tests were carried out on Samsung Galaxy S8) I have decided the official build is the one with no postprocessing attached. The game still looks good and runs smoothly, which is the opposite of two other builds, where I have noticed significant performance issues. However, the other two builds have been attached and are free to look at in order to compare performance.

**Resources**

Nowadays, to make games is super easy, as there is plenty of assets of all kinds available online. They can be both paid or free. Unity3D has even it’s own Asset Store. All developers may make use of that, even in published games, as their license allow to do that.

While developing *Paladin’s* *Tale* I have also used of Asset store and other online resources, such as mixamo.com (used to download animations and player character). All of my 3D models – buildings, potions etc., as well as music and sound effects – have come from the Asset store. I have also made terrain with very useful and easy to use Terrain Toolkit.



In terms of code the only thing I have downloaded from Asset store is joystick (which has come with ready to use sprites). I have used it because I wanted to test out how good can free assets from Asset store be and it appeared to work brilliantly. However, I have altered the code a little bit in order to match and work with my game.

The reason for downloading graphics and music is that my main interest is programming and due to the time constraints, rather than spend my time on drawing or creating music, I preferred to focus on creating a good code.

**Testing**

No game can be good without quality assurance, which checks the software on each step of development and ensures the final product will be fun to play and bug free. In terms of mobile devices there is also aspect of device diversity – developers have to ensure the device will be playable on as many devices as possible in order to hit target as big as they can.

Sadly, I do not own any Android or Mac device, so I could not build and test the game on my phone. However, I have done testing using Unity Remote feature – which, however, does not run on the phone, but is just streaming Unity’s output to the phone - and just by playing the game on my PC while developing.

Apart from that I have outsourced some tests, which were carried out on Xiaomi phone. Here are example results of such tests.

|  |  |  |
| --- | --- | --- |
| **Subject** | **Intended outcome** | **Actual outcome** |
| Movement | Player moves around | Player moves around |
| Attack | Player performs attack and deals damage to enemies | Player performs attack, but has issues with dealing damage – works fine 50% of time. |
| Spell casting | Correct spell is being cast upon button press | 3 correct spells have been cast upon pressing appropriate buttons |
| Fireballs | Fireball spawn around player | Fireballs spawn, but not in the circle around player |

I have also used Unity’s profiler, which is a great tool to see how efficient the game is. Looking at the output of one of the final stages of development build the game seems to be efficient, taking only around 313MB of memory. It seems to be working well on CPU too. For the profiler data please see appendix 7.

I have fixed all of the issues that I have come across during development process, so the game is fully playable.

However, there is one unfixed issue – the game seems to have issues with finding the target on mobile phone. This mechanic is quite important, as it indicates what enemy to attack (as mentioned above, players can’t choose target manually and it’s being done automatically). I have encountered this problem on Samsung Galaxy S8 and it seems that it is something deep down on the hardware and has to do with certain processor architecture. It has to be said, that when I have made a test build on a different device – a PC – I have not experienced this issue. However, this is something to take a closer look at and solve in future further development.

**Learning outcome**

I found this project highly educative. I have learned a lot about useful Unity3D features, such as NavMeshAgent, I discovered a lot of sources of free assets and I got to learn more about additional features – for example shader programming. I was also having fun while developing this game, as it is the genre I have a sentiment to and I found it really enjoyable. I am sure gained experience will allow me to build better structured code in the future and, overall, will help me be a better developer.

# **References**

[1] Newatlas.com, (n.d.), *2013 smartphone comparison guide* [online]; available at <https://newatlas.com/2013-smartphone-comparison-guide/29878/#gallery> [accessed 23.01.2019]

[2] Brewster, S., 2002. Overcoming the Lack of Screen Space on Mobile Computers. *Personal and Ubiquitous Computing,* May, 6(3), pp. 188-205.

[3] Chehimi, F., Coulton, P. & Edwards, R., 2005. *Evolution of 3D games on mobile phones.* Sydney, IEEE.

[4] Gong, J. & Tarasewich, P., 2004. *Guidelines for handheld mobile device interface design.* Boston, Proceedings of DSI 2004 Annual Meeting.

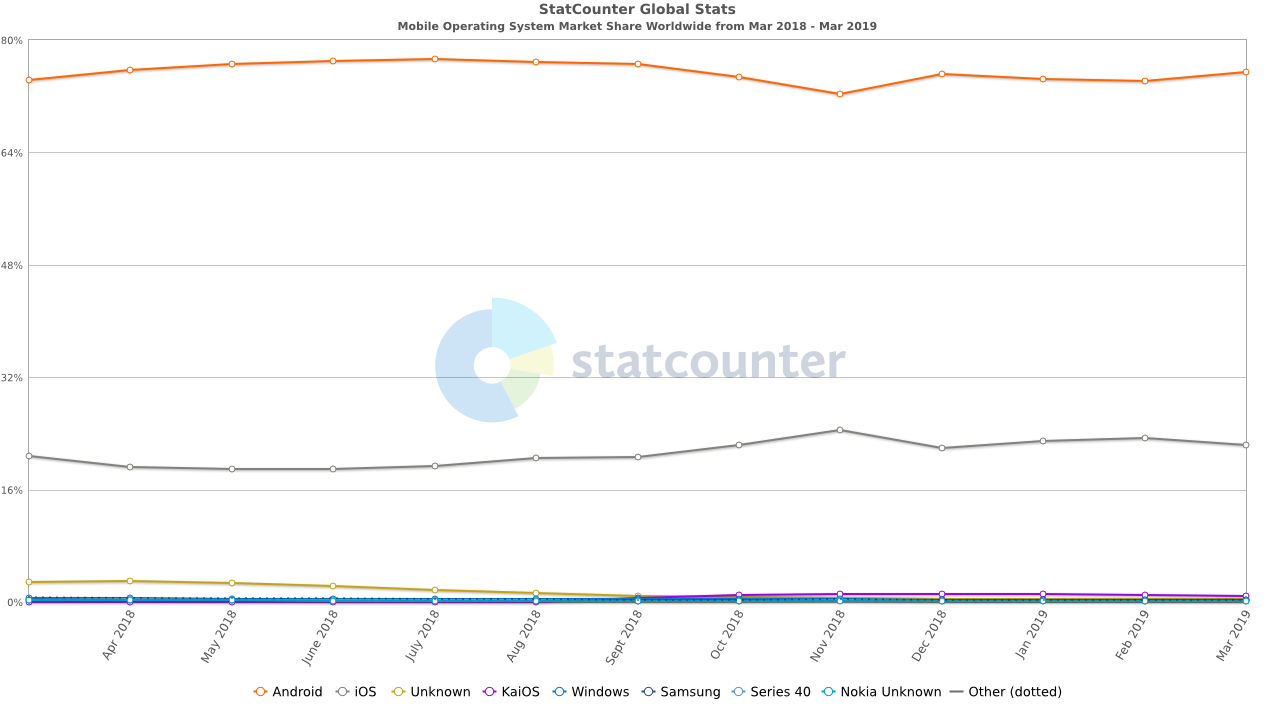
[5] Kimberly, C., Lim, Y. T., Khong, W. C. & Wong, Y. C., 2016. Usability and Player Experience of Input Device for Mobile Gaming. *International Journal of Mobile Human Computer Interaction,* July, 8(3), pp. 52-69.

[6] Koivisto, M. I. E., 2006. *Mobile Games 2010.* Perth, CyberGames '06 Proceedings of the 2006 international conference on Game research and development.

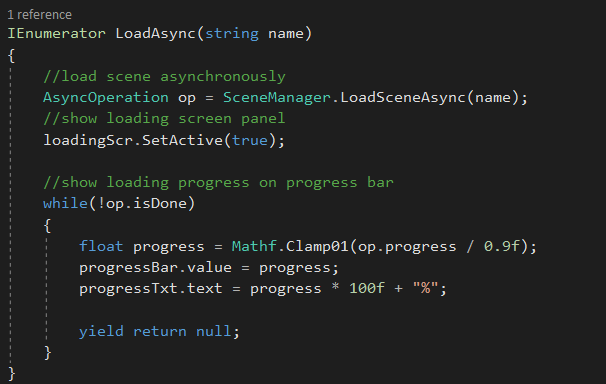
[7] Statcounter.com, (n.d.), *Mobile operating system market shared worldwide March 2018-March 2019* [online]; available at <http://gs.statcounter.com/os-market-share/mobile/worldwide?fbclid=IwAR2hz7BqHJB1z9GIj8R6k8mSSkB6brkd345ZmKKOG3aRObkwCImZJ6dtWpo> [accessed 01.04.2019]

**Appendices**

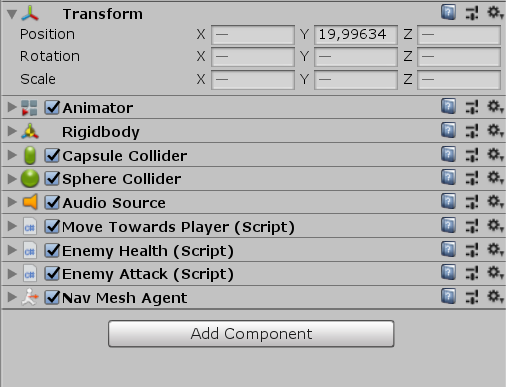
Appendix 1: Market statistics



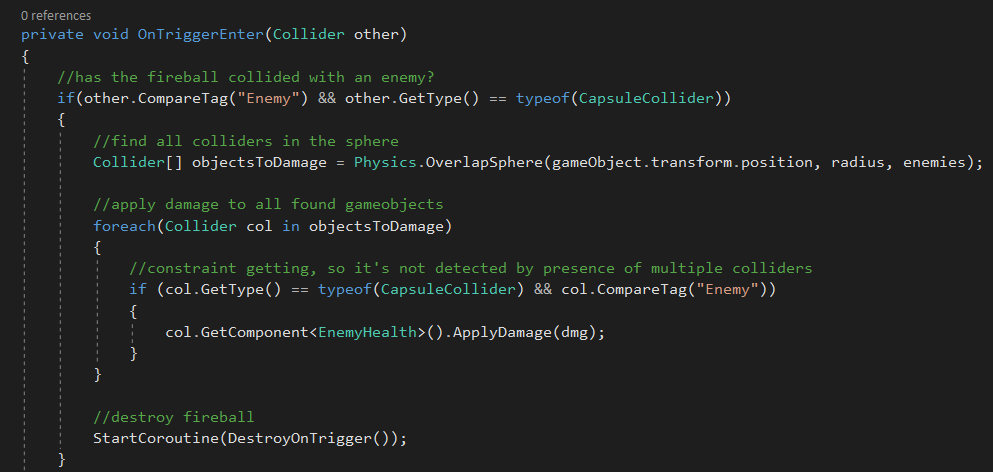
Appendix 2: Example coroutine



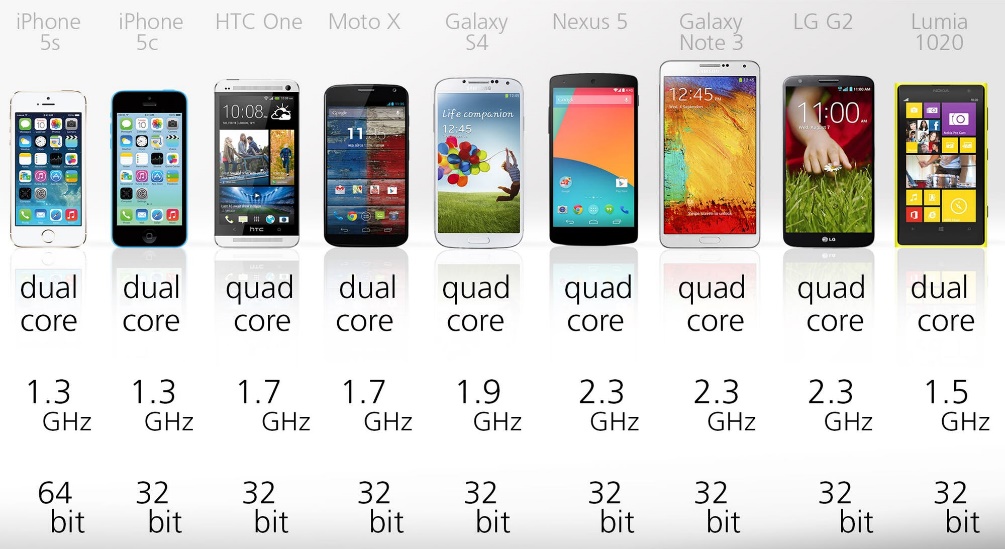
Appendix 3: Enemies components

****

Appendix 4: OnTriggerEnter() function



Appendix 5: Example mobile phones specifications





Appendix 6: Profiler

