Bachelor Thesis for the University of Twente

Enhancing Video Game Design: Involving Users into the Design of Video Games

By Dennis Vinke.

Supervisor: Job Zwiers

Second Supervisor: Dennis Reidsma

Student number: 1390082
Date: 12-08-2016

Abstract

This bachelor thesis explores possible uses of participatory design into the early stages of game design. This was done by looking at existing solutions found in other types of software development and was tested on a prototype of a game that was made especially for this project.

Currently there is a severe lack of user participation in the most early stages of game design, Causing prototypes and game ideas to be neglected after the first playtest deems them not fun enough. These game concepts could be explored more with feedback from actual members of a target group to make a new prototype. For big companies this would be deemed a less efficient development method, but for smaller companies revising their earlier concepts or work with their potential customers from the start, this could prevent a loss of money and resources.

The research was conducted on a prototype of a game that was made alongside the early stages of this research. The game was designed based on other studies that tried to make the concept of fun quantifiable, to make sure the prototype has a certain level of fun upon which testers could improve. The game also contained several tools to measure the player's performance during gameplay. The testers got to play the prototype and got the opportunity to say what aspects they would improve. The feedback gained from the playtests and the data gathered by the game showed a lot of potential ways to improve the game. How well these ideas and improvements are in comparison with the prototype built for this thesis could not be tested yet because of time constraints and this should be done be done in a future project.

Acknowledgement

First and foremost I want to thank my supervisor Job Zwiers for allowing me half a year ago to let me undertake this venture. While it was a difficult and intensive project to do a research alongsides making a game engine and a playable prototype in the short time frame that was available for this bachelor thesis.

Next I want to thank fellow Creative Technology students Peter Verzijl and Daniel Pel for providing me with assets for the game's prototype while they were also busy with their own projects.

Lastly I want to thank Carlijn Oosterveen, Nathan Middelham, Thijs Dortmann and Robin van Emmerloot to be there to act as a rubber ducky when I needed to ramble about ideas, had to collect my thoughts, or needed any other help.

Table of Contents

Chapter 1 - Introduction

In game design, including users from a target group is are often included to find the faults and problems in an already existing concept. This concept came about by having a single person or a small team create and iterate over several concepts until the development team is satisfied with their prototype. This prototype will be shared with a small group of people, which will determine if it is fun. This is contrary to the popular belief that a game should be be played by a huge group of players as soon as possible[1,14]. The users could already be included in the first prototype iterations to create a game together with the actual target group.

For the bigger companies it is easier to dismiss a prototype or game idea when not enough people thought it was fun, instead of trying to find out why the concept did not work. An independent studio or developer has less leeway in throwing something away where they spend time and money on. For this group it could be interesting to design the next prototype together with their intended consumers. This so called participatory design is commonly used in other types of software development, so it could be interesting to see how this concept could be used in game design.

Earlier attempts to include gamers into the design were made in the past, like the canceled Mega Man Legends 3[1], which opened a forum where players could help decide which designs for characters would be included in the game. Occasionally developers also give their fans the opportunities to include their own designs into the game, like it was the case for Dark Souls II's design a shield contest[2]. All of this is purely cosmetic and even through means of Steam's early access[3], beta tests or playtests the core of the game has almost never changed and user inclusion in the iteration process of the game is as good as non existent.

[1]http://www.capcom-unity.com/devroom
[2]http://www.darksoulsii.com/uk/news/forge-your-own-original-shield-design-and-secure-your-legend-beyond-grave-dark-souls-ii.html
[3]http://store.steampowered.com/earlyaccessfaq/

The tools to include users into the design of a software project should be already available with studies conducted on participatory design, game design and of how quantitative data could be included in the game. To discover which tools and how these tools could be utilized in a participatory design based game design process, user inclusion solutions in the other types of software development will be examined. A prototype of a game will be made with the same constraints an independant developer would have to test the theory on. This newly made prototype should not only have a solid base on which could be constructed upon further, but should also be at a certain level of fun.

The necessary steps taken to fill in the knowledge gaps are found in the Background, Ideation, Realisation and Evaluation chapters. In the Background chapter, all the relevant studies and needed materials found during this research are discussed. This also includes information on which characteristics of games make a game fun. This can then serve as a base for the prototype.

The ideation phase is split into two sections. One section focuses on how to incorporate users into game design and comes up with concepts that makes it possible, while the other section is dedicated to the prototype and how it came to be. In the Realisation chapter it is explained how the prototype is engineered and how incorporating users into game design and the prototype comes together. The evaluation phase discusses the results that the prototype achieved after testing.

Chapter 2 - Background

To get an insight in how game design and participatory design are conducted, a literature study was done. To design a game, it was also needed to do a small literature study on what makes a game fun. The most important parts of the Literature research can be found in the section Literature research, which can be found below. The complete Literature Research can be found in Appendix A. Furthermore a small inscription of noteworthy projects that help with participatory design or game design were gathered in the state of the art section.

Chapter 2.1 Literature research

Playtesting

To get a good understanding about the likes and dislikes of a potential game idea, tests on so called focus groups are conducted. "It refers to sessions where potential players are interviewed about their likes and dislikes, often in an attempt to determine whether they like a game idea that a company is considering."[1] The most known game test, quality assurance has nothing to do with how enjoyable a game is [1].

Something that does test the enjoyability of a game is playtesting. Playtesting is a good way to force a developer to face the problems that have been put off [1]. Playtesting finds problems early enough to fix them and it also solidifies if the game is made for the right audience[1]. Playtesting is also essential to make a good game![1] Playtesting is also one of the most scariest test a developer or designer can conduct. When making a game with a lot of effort, the team behind the game will fear to be criticized[1]. This fear can make it so that certain feedback is dismissed, sweet talked or the tester will be persuaded to change its opinion.

According to Jesse Schell the four groups these test are conducted on are developers, friends, expert gamers and tissue testers[1]. Every group has its own pros and cons on what kind of feedback they will deliver. The test can take place in the game studio, playtesting labs, public venue, a playtester's home or on the internet [1].

Playsessions are done by observing a tester play the game. It is also possible to collect data automatically. “The more data one collects the more useful the data will be to you” [1]. Data mining is a subtle art that gives new opportunities to understand player behaviour [1].

Disturbing players while they play the game is a double edge sword. On one hand asking the right question may give an insight which would not be gained otherwise[1]. On the other hand is it a possibility to disturb a player and interfere with their natural play patterns[1]. The “think-aloud protocol” could be seen as a middle road where a player says what he or she is thinking without needing any interference of the person conducting the test [1].

To end a playtest, interviews and questionnaires could be used. For questionnaires Jesse Schell recommends to use an online survey to save a lot of time. He also states that a likert-scale containing Terrible, Pretty bad, So-so, Good and Excellent is a lot better that using a scale from 1 to 10. A survey should be given right after they played and there should be someone to ask questions to clarify answers [1]. Another nice datapoint which should be considered when making a survey is to take the age and gender of each playtester surveyed [1]. The last important remark Jesse Schell makes about questionnaires is that the data should not be taken as gospel.

For interviews one should have a script of questions ready, The interview should be conducted in private[1]. Also take in mind that playtesters will avoid to hurt the feelings of the person that conducts the interview and that the interviewer’s ego should be put aside [1]. An interesting statement Schell makes is that one should not expect playtesters to be game designers. This is contradicting the goal of the research where the playtesters will have input on the next iteration of the game. This would mean that only people with at least a rudimentary understanding of game design will be useful for the input of the research.

Game interface

Playtesting is not the only way to find the faults in a game. Nowadays Nielsen Heuristics[12] are a commonly used instrument to find flaws in the design of user interfaces for computer software. Federhoff [5] makes it clear that having a poorly constructed interface can keep a player from enjoying a game, so it is important to actually make it so that the interface does not confuse the player. Federhoff [5] divided the elements of a game that can be measured through the ten user heuristics.

Interaction

The interaction between a player and a game can be divided in three levels, which are Game interface (what the player sees), Game mechanics (the rules of a game) and Game play (how the player interacts with the rules)[3]. Where Clanton also states that most games have interfaces that Human Computer Interaction (HCI) designers would consider obvious flaws. Some examples for the problem he states are hard to navigate menu's, unclear instructions and games that do not feel nice to control. This are some problems which will be encountered when testing an early prototype. How strong people will react to the HCI faults described by Clanton will be encountered during the testings itself.

While a flaw in the user interface does not have to be obtrusive, a flaw in the game's mechanics is [3]. Clanton continues by saying that game play is about overcoming problems, so problems in the game interface and mechanics can be masked by the fun of solving problems in the game. The reason for this according to Clanton is the fact that people enjoy solving problems and love to overcome meaningful obstacles. These problems can become the goal of a game.

According to Lazzaro [9], the four components of entertainment are hard fun (fun out of challenge and completing goals with skill), easy fun (fun by exploration and roleplaying), altered states (fun by clearing one's mind or to avoid boredom) and socialization (Fun by interacting or watching others). Yannakakis et al. [10] noticed that three of the four types of fun Lazzaro [9] found have a relation with the intrinsic qualitative factors of Malone [6] stated earlier. Yannakakis [10] also acknowledged another definition of fun, which is composed out of endurability, engagement and expectations. The four components as defined by Lazzaro can all be brought back to what kind of goal a game was intended with.

Goal

While Crawford [4] believes games should not be solvable, but contain an illusion of being winnable. Other researches believe that a game is played to achieve a goal, and because of that should be achievable. Malone [6] on the other hand states that the attainment of a goal should be uncertain. While the containability of a game is not clear, the goal should least be clear and interesting to the user. Jegers [8] even states that concentration on a game is possible if the tasks in a game is clear.

Chuck Clanton [3] states a goal should be set early in the game, but can be changed during the duration of the game. Changing the goal during the progression of the game can keep the game fresh. Another way a goal can induce fun is by reflecting on the player's progression in the form of score-keeping[6]. Federhoff [5] complements this statement by stating that a score is form of positive feedback to encourage players to enhance their mastery of a game. Jegers [8] adds to this, that a player should always know their status or score. Platinum games[4] made good use of this knowledge by letting their players know for what they attained a score, so the player knows what should be improved. A goal can also be made fun by setting conditions[6]. These conditions introduces pressure to the goal, which can be fun[3].

Curiosity

Curiosity is also a factor that keeps a player hooked to a game. Curiosity is the motivation for a player to learn, that in itself is not directly related to a goal, or fantasy-fulfillment [6]. Curiosity can also be used to enhance the fantasy or challenge aspect directly[6]. Yannakakis [10] agrees with the fact that curiosity is part of what makes a game fun. Getting involved and motivated in combination with clear goals and challenges could lead to getting the person aroused and influence the focused attention [11]. Since curiosity seems to motivate a player, it should also be a goal for a designer to create a game that makes a player want to explore the game or its mechanics.

Game mechanics

Where the story and setting is influenced mostly by the designers, the second level - game mechanics - is at the mercy of the game play. This does not mean that everything is allowed when creating the mechanics because Federhoff [5] identified that "mechanics should feel natural and have correct weight and momentum." She also states feedback should be given immediately to display user control and get the player involved quickly and easily. Yannakakis et al. [10] state that gameplay also "affect the player's cognitive process." **Even if the mechanics are complex, it should not influence enjoyment if the game is fun.** When a player has fun he or she is willing to learn the more in depth mechanics of a game

[4] https://www.platinumgames.com/

[5]. Therefore, it is important for a game's mechanics to be easy to learn and hard to master, but there should also be a balance of skill and challenge in the game [5].

Chapter 2.2 State of the art

User testing, usability testing and user experience testing is nothing new in game design. That is way this section takes a closer look at the tools already in use to get the data needed to improve the overall quality of a game. There is done some research to solutions found by other types of software development. In particular the website evaluation tools that conduct test with the help of the internet because this branch of software development dwells a lot on user feedback.

Zynga[5]

This company makes mobile games entirely based on software metrics. A gameplay element is not even considered if Zynga does not have a metric analysis that proves it will have a positive effect on the game.

UserTesting[6]

UserTesting is a platform that has a large user base to usability tests based on the prototypes, apps and websites uploaded by customers. UserTesting also offers a solution to test mobile games. The features they promise consists of videos and audio results from the testers and a recording of the taps, swipes and gestures those tests have done during the test.

Alpha and Beta testing

Public beta testing, a test where developers put on online version of a game in production, can get millions of testers who give a lot of feedback, just by making the beta accessible to the general public [2]. These tests can give a good insight in a player's do's and don'ts, which can be used during a game's development to make the product better. The problem with a public beta for a game is the fact that the game is relatively close to its release window, so the data collected will not bring any major changes to the core gameplay. The

[5] http://zyngagames.com/

[6] https://www.usertesting.com/who/mobile-game-developers

recently timed alpha for Nioh[7] shows that developers are considering to release far from finished games to the public and ask input from their player base.

Eye tracking

Nowadays eye tracking during playtesting is becoming more and more common. Eye tracking is used because it provides direct and objective insight into player behavior.[8] Eye tracking helps with observing what on the screen gets noticed first, what objects are ignored, how hard the path finding for the player is, how players search for clues to progress the game and to identify what was looked upon last before a player become confused or frustrated.

PlaytestCloud[9]

PlaytestCloud is quite similar to UserTesting, as the site offers usability experts that help set up a test, tester to play the game and offers a video of the playtest in return. PlaytestCloud is only usable for mobile and browser games.

Simple usability behavioural research consultancy[10]

Simple usability is a company that mostly focus on researching user experiences of websites, products and advertising. The company also conducts Gaming UX research on all possible gaming platforms. They research the user experience with the help of eye tracking, in-depth interviews, video recording, EEG's and claims to let the testers be recorded in a natural environment.

Player Research[11]

Player Research is a company that specialises in user experience research for games. What makes this company special when comparing to similar companies is that Player Research also includes a competitor analysis. The company also does playtesting at home and at

[7] http://teamninja-studio.com/nioh/
[8] http://www.tobiipro.com/fields-of-use/user-experience-interaction/game-usability/
[9] https://www.playtestcloud.com/
[10]http://www.simpleusability.com/our-services/games-testing/
[11]http://www.playerresearch.com/services

players’ homes. They also offer a expert evaluation and also do an evaluation of the game post-launch.

PLAYTEST[12]

The bigger game publishers have their own user research lab. Ubisof for example named their user research lab PLAYTEST, where they recruit testers from age three and older. The reason they do it inhouse and are not outsourcing the tests has to do with keeping their Intellectual Property from disclosure. The publisher keeps a list of interested participants, which they invite for a test if their profile matched the pre-requirements for a game that needs to be tested.

Recording through Twitch.tv[13]

Twitch.tv is the world largest social video platform and communitie for gamers. It would be nice to incorporate the streaming site in a user test to see what the user experience of a certain game is. That is probably what Kacey Misskelly thought when she set up a test with the help of Twitch. The idea behind this concept is that players play a game while they think out loud. This way the observer can observe a player over the internet and while the site uploads the recorded video on Twitch. The video can also be uploaded to YouTube, after which it becomes capable to download the video to a hard drive. Interaction with the streamer could be achieved by doing a VOiP call, while the streamer is actually conducting the test. This way an observer can ask questions if something interesting happens. A problem with using Twitch.tv as the source of your observing material arises with setting up a recording session. A tester needs to have knowledge of both streaming software and Twitch.

“By running tests on games in the same genre as the one they’re developing, studios are able to see how their game is going to stack up against the competition once it’s released onto the market.” [Misskelley]

[12]https://playtest-secure.ubi.com/index.php?language=EN
[13]https://www.usertesting.com/blog/2015/08/11/desktop-game-testing/

Concept Feedback[14]

Concept Feedback is a site where a person or company can upload a website design on which experts from the site's community will offer feedback in the form of suggestions, recommendations and input about the design that was uploaded. The uploader can rate each individual piece of feedback so the reviewer gets a track record. Other users can use the track record to determine what the quality of the feedback is based on the scores the reviewer gets.

Chalkmark[15]

Chalkmark is a site that creates heatmaps from user's first clicks on a website. This done by uploading a picture or framework of your website and a list with tasks the tester should accomplish. The site than registers every first click made by a new user. It aggregates this data to into a heatmap so the user gets an impression of how a user starts with fulfilling the task. With the help of the data acquired from the heatmap, you can optimize the first click to make the site more usable and intuitive.
An example of how the site works on a Coca-Cola advertisement can be found here[16]: https://www.optimalworkshop.com/blog/the-remote-testing-tool-that-keeps-on-giving-when-chalkmark-met-coca-cola/

ClickHeat[17]

Like Chalkmark, ClickHeat creates a heatmap based on the places visitors of a site clicks. ClickHeat does this for the whole site, instead of just the first click as is the case with Chalkmark. The reason this one gets a special mention is the fact that ClickHeat offers the option to incorporate the function to your own site without the need of using HeatMap's own website.

Treejack[18]

Cut the visuals to make the structure more clear.

[14]http://blog.conceptfeedback.com/
[15]https://www.optimalworkshop.com/chalkmark
[16]https://www.optimalworkshop.com/blog/the-remote-testing-tool-that-keeps-on-giving-when-chalkmark-met-coca-cola/
[17]http://www.labsmedia.com/clickheat
[18]https://www.optimalworkshop.com/treejack

OptimalSort

Let's the users give their own implementation of a certain order. This concept could be interesting to incorporate in the participatory game design by explaining testers the concept, and ask if the players could tell how the game would work in their understanding.

ClickTale[19]

ClickTale is a tool that a gives users the ability to record and replay sessions of visitors visiting their site. The tool also offers the input given by all the visitors by generating a mouse movement, mouse click and attention (how much of the page was actually visible on the screen during one visit) heatmaps on every page of your site. Next to that it also places all website traffic analytics in different categories on the same page to get a nice overview of those analytics. ClickTale also keeps analytics of how the website visitors interact with the webforms on the page. In short the tool combines A/B testing with web analytics and user behaviour tracking. The nice part of this tool is that it points out were the errors on a site are so the user can look at the recording to gain insight on what went wrong.

Five second test[20]

Five second test is a site where people can upload home page, landing pages, logos, brochures and marketing material. Once uploaded, the site will show the upload to other site users for 5 seconds. Once the five seconds are over they have to note down how much they could remember from the site. Once done, the site asks the visitor what he or she liked and disliked from the site. The site collects all the words filled in and generates a word cloud out of it.

Loop11[21]

Loop11 is a site that gives testers the option to create a list of tasks and questions online and link it to the site the tasks should be completed on. The tester can invite tester through its own site or through one of the clients of Loop11 to fulfill the predefined tasks. Loop11 gathers the success, fail and abandon rate of every task and also keeps track of the clickstream of the testers and generates visualisations out of it in the form of heatmaps.

[19]https://www.clicktale.com/
[20]http://fivesecondtest.com/
[21]http://www.loop11.com/

Morae[22]

Most tools record only what happens on screen. Morae is a tool that actually tries to record the user that is testing a site in his or her own environment. The tools does this by recording the tester's voice, facial expressions and eye movement while also recording

mouse movements, clicks, keystrokes, the actual screen and how long a person stays on a page. Multiple observers can observe the recordings at the same time at different locations because the tool also offers a chat program to communicate with fellow observers and the ability to assign them to certain tasks.

[22]https://www.techsmith.com/morae.html

Chapter 3 - Ideation User Inclusion

Introduction

The idea to incorporate user feedback into a game's prototype sounds like a nice concept to attempt, but the hurdle to get useful criticism and comments will be hard. This chapter will discuss what possible solutions get the needed answers were thought of during the duration of this project. This section is divided in the subquestions: How to find the correct testers?, Which existing techniques could be used to acquire user feedback? and how to combine the two questions in a viable product.

Chapter 3.1 Ideation User Inclusion

While searching for solutions to include users, certain requirements were in place. First of all the kind of game makers that would benefit the most of this research would be the independant small developers. So the solutions should keep that target group in mind.

Chapter 3.1.1 Finding suitable testers

How to gather testers?

Before you can actually test a game with testers, these people need to have access to the game. A small developer has not a lot of money as it is, so they want to have access to a lot of testers without paying much. For this reason alone it is not viable to pay testers to test a game.

The easiest way to acquire testers would be by asking people in and around the place you can do place a test setup for the game. By having the testers in personal, you have the opportunity to interact with a test player and ask them questions like: “Why did you choose to jump here” and “Did you know you could dodge by pressing …” to get a better insight why players did certain actions. This can also become a big negative if the developer is very defencive of the game and tries to rectify all the criticisms of testers or the testers does not dare to hurt the developer and sweat talks or smoothens their quirks with the game. A nice side effect of being able to create your own test setup would be the capability to use more elaborate testing equipment and for example record a player, or observe the player from another room.

The problem with asking persons in the close vicinity of where the tests take place, is most certainly that the testers a person can test with is limited. It is most likely the players playing the game are not that well engulfed in the genre of a game, or gaming at all. Another big problem with having a limited tester pool is that it will be hard to get different groups racial,age or gender groups. For a test group where people would prefer people from one of the focus groups the game is aiming to attract or getting a wide variety, a large tester pool would be a must.

Enlarging the pool of testers by placing advertisements online or in cities relatively far from the game's testing spot would also be less than ideal, because those testers would more likely than not find the distance to travel to the testing spot cumbersome. Even more so if the only compensation they get is playing a game in development and a small refreshment. Though one can say that the persons one may attract this way is a dedicated and suitable tester, it is not very likely this way of attracting people will offer much participants.

Another way to attract a lot of different players is by spreading the game online. With the help of demo's, beta's, alpha's on one hand and the ability to buy unfinished games on Steam's early access[23] and the Humble Store[24] on the other, people are already getting used to playing games that are still in production. With Steam early access it is also easy for players to give feedback to the games that are part of the early access program. Also without the use of a storefront like Steam's, it is possible to get a lot of player feedback by using a dedicated forum, social media or something similar. The feedback delivered will be more honest, simply because the comments made are

The biggest problem when attracting testers online with this approach would be the loss to have interaction between the tester and the one conducting the test. This would be one of the issues that certainly have to be taken into account and should preferably be solved when going for this approach, because this inconvenience is far from unsolvable. Not having a way to interact directly with the testers prefends the one that conducts the test from defending his or her work, and the anonymous tester that can say what it wants without feeling the barriers confronting the tester directly. This fact alone will make it so that the feedback given will be more honest and in line with what a game actually offers.

[23]http://store.steampowered.com/earlyaccessfaq/
[24]https://www.humblebundle.com/store

That last point could also be seen as a very valuable aspect gained in losing the direct interaction between a player. Another big negative for this approach would be the inability to keep the game a secret, which can have an impact on the marketing side as well as other developers that can steal and incorporate the ideas shown in the playable version of the game.

The biggest reason to go for fetching players online is because this approach offers the broadest range of players and player types. This means not only that it is possible to find the playtester that fits in your target group and has an affinity with the type of game you are currently making, but it will also attract players which you did not take into account earlier. All the feedback received could give a tester the aspects of the game that is also loved or hated by the people outside the target group, which could be used to enlarge the target group or make at the least, make it more accessible to other players.

Chapter 3.1.2 Acquire feedback

What test could be used to acquire the needed data?

To get a good understanding of what the possible testing procedures in current software testing are, a mind map was created. In this mind map the most used software tests that could potentially help out with game design are written down. The mind map can be found in Appendix B. During the search different software tools were encountered that utilize one or more of the test included in the mind map. These tools can be found in chapter 2.2 State of the art above. The most interesting techniques are discussed in more details down below.

Playtesting

Playtesting is maybe the most easiest and obvious choice to test a videogame with. The idea behind this test is to find a group of people that fit your target group. Let them play the game and observe while they are doing so. While playing the game they are expected to write down or state the problems they encountered. Once the session is done, the testers will be asked some questions about their experience with the game and the comments they made during the game. This way the persons conducting the test get a good impression of what parts of the game needs some more work, are plain boring or are genuinely fun to play. It helps improve the overall quality of the game. Having a direct interaction with the players, it is also an easy way to gain the ideas a tester may have on improving the game.

Game level testing
Another aspect the testers should be able to have a lot of impact on is the level design and game pacing. A way to do this would be by letting the testers play parts of the game or levels. The way this test would be conducted, should be similar to play testing but with the emphasis on the level and game parts.

Artificial intelligence testing
The last important part of a game on which the input of a tester could be useful, is on how the enemies behave. This can differ from making some parts easier or harder to suggesting new movement patterns for a enemy. Like with the game level testing, this input would be gained during playtesting. It would help to have testers that are familiar enough with the game in such a way that they understand and are capable of using the more difficult parts of the mechanics. This means that those testers need to participate in multiple playtests, or should have access to playing the game in their own time to master the game play.

Usability testing
Usability testing is one of the more important tests one can conduct on a game. Usability problems are one of the more likely reason that the user experience is influenced negatively. If a player is annoyed by the interface while doing a repeating monotonous task like item management in an inventory or restarting a level, the player can drop the game easily even though the game itself can be fun. The problems in the way a player interacts with a game should be noticed and fixed as soon as possible, so this would be a test that a game could benefit from in an early stage of its development.

Interface testing
As noted above, it is important that different components of a game should not influence the user experience. Like with the usability testing, it is important that a player understands the interface of the game, or else it will make the player spend unnecessary time away from the actual meat and fun of a game.

Compliance testing

To actually release a game on the consoles of Sony, Microsoft of Nintendo the game should pass a certification process for every of the three companies. This certification process is there to make sure the game does not crash or hurt the hardware it is running on in any way. While compliance testing is an important testing asset in normal testing, it does not have any relevance in this project because the scope of this project is limited to games and prototypes that are in early development. That means that so much can still change in the game's lifecycle that it would not be wise to focus on the points the results of these test would produce.

Integration testing

While integration testing would normally comply on different components of a system coming together into one system, the concept could still be used for a game. Instead of testing the system, it could be used to see how well the different tests that are included with a prototype are working together and which of the tests became redundant because another test provided the same information and more.How to actually define how usefull testing the compatability of test are would also be a challenge on its own.

Feedback through hardware

Using hardware to measure how certain body parts like eyes or the heart reacts while a tester plays a game is an interesting method to incorporate. This is another method that can provide valuable player feedback without the player actually having to say a thing.

Blackbox testing

Because a tester playing a game will most likely know next to nothing of how the inner workings of the game functions, blackbox test are the most commonly used methods to test a game. It is no surprise most of the mentioned game tests fall under the blackbox banner.

Performance testing

Performance testing would be interesting for a game that is well on its way in its life cycle. For a prototype to be tested, the game should run well on the device it will be playtested on or on a whole range of systems if online distribution is the chosen method to gain playtesters. In each case performance tests would be performed before the game's prototype is played by the testers. It could be an option /to track the hardware of individual players if the game will be distributed over the internet to get a general impression of the hardware that is used to play the game for a later stage in development.

Use case testing/ heuristic testing

Use cases could be used to guide a user through the playtest by giving them certain assignments. This way, a tester could make sure that certain aspects of the game are experienced, while also getting a general impression what every influence certain aspect have on the game's usability and the user experience. This could also works as a small tutorial to the game.

Exploratory testing

This form of testing could help in gaining an insight in how easy it is for a player to get accustomed to the game and its functions. This method will probably be more useful when one can observe the tester, so this could be an interesting way to test testers. This would not work if the prototype will be distributed online. The reason for this is that players will miss necessary game data, which can get people stuck during the test. This can be prevented to make a help system inside the game, or guide the player through the demo with the help of the aforementioned use case testing.

Ad hoc testing

While ad hoc testing and exploratory testing are roughly the same, the main difference between the two tests is the knowledge that gets dispatched to a tester before they start the test. In ad hoc testing all the necessary data is given before the test starts, which makes it usable for both offline and online distribution. How ad hoc could be used in a game, is by giving the player a room where all the game mechanics are included. It can give interesting results based on the options the player decides to do. Will the player engage enemies, explore the room or do something that was not expected by the designers. It could possibly be a way to find out which mechanics are not clear or fun to use, based on the amount of interaction between the game elements and the testers.

Whitebox testing

In software testing whitebox testing is a solution often used to see how well the software is performing and if there are bugs in a piece of code. The testers know how the system works and more often than not also know the codebase of the system. It is not probable that a game tester knows anything of how the game works, so using test classified as a whitebox test would not make a lot of sense. While whitebox testing methods are not really viable in game testing there are certain methods that could be usable.

Software metrics

Software metrics is one of the more interesting types of testing that can be done as a whitebox test, without the player or tester needing the knowledge of the inner workings of the game. With this test the game keeps statistics of the player that is playing the game. The game can track the positions of where the player is, how often he or she does a certain action like jumping to more elaborate concepts like the accuracy of a certain weapon on an enemy in the air, while the player is riding an elephant. This data could be useful to see what type a player is or how a certain player reacts to different types of content. This method could help the game's concept and give generate player input through actions without the tester giving any criticism.

A/B testing

A/B tests offer testers with two or more options and decides which option is the best by selecting the option that received the most clicks. This is a commonly used tool in website testing, but it can also be used as a tool to learn user behaviour [25] [26] . This can also be used in a game by giving the player options between certain game elements or equipment to see what is prefered by players. In later stages this system could also be used as a balancing tool to see why i.e. a player prefers weapon A above weapon B. Even more so if this test is mixed with the possibility to change attributes of the options a tester can choose from on the fly. This test can also be an option to determine if a certain incorporated gameplay suggestion made by a tester is better than the first version by letting another tester test both versions of the game and let them point out which of the two versions is more fun.

Expert evaluation

Expert evaluation is a practice often used to identify problems on a certain aspect. An example of this is an usability expert evaluation (or heuristic evaluation) which points out the flaws in a user interface design or a UX expert evaluation which does the same for the

[25]http://nerds.airbnb.com/experiments-at-airbnb/
[26]http://nerds.airbnb.com/experiments-airbnb/

user experience of a software application. For the project it would be nice if "experts", which are game developers and people well versed in the genre of the game being tested, could have access to the game and point out what they think is wrong with the gameplay or general design of the game. This advice should be separated from the feedback "normal" players deliver, if the developer wishes so. That way a developer or development team can decide if they want to emphasize on the feedback received from the so called experts, players or take both groups' feedback on equal weight.

Chapter 3.3 Possible idea's

Idea 1: Prototype distribution platform

Based on the thoughts above the original idea was to construct an online platform where small developers could upload a playable version of their game. Players and experts would then be able to play the game and leave feedback when they are done. At the same time, the game should also keep track of certain metrics, which it would send to the platform or developer. After a small period the developer could dig through all the gathered data and based on the results improve the game. This cycle would continue till the developer would be content with the latest iteration of the playable project.

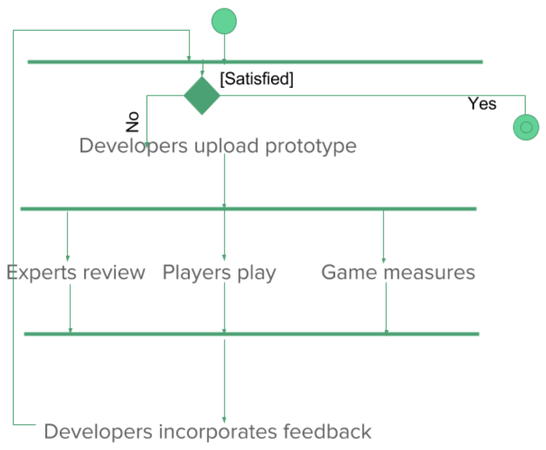


Image 1: Possible activity diagram of the distribution platform

While the idea in itself sounds feasible, it would not be possible to gain enough traction in six to eight weeks to create a user base big enough to make a nice sample size for the actual tests. Even though there is the possibility to use existing sources with a reasonable user base to direct them to the new site or download, it is not realistic to gain traction with the amount of people.

Like mentioned earlier it is not possible to interact directly with a person that is testing the game in his or her own environment. This means it is harder to actually get the feedback a tester is looking for. Using the camera equipment of the testers PC or laptop is always an option as long as the tester agrees. A problem with this is that one does not know where the camera is recording. It could also record things you do not want to see or produce recording so that no additional software like face-tracking is usable so all the recordings would have to be tracked manually.

Not all people feel obliged to fill in a survey or give feedback after they play a demo. When a person is there to supervise them, it is more likely they will actually fill in the survey than when there is not a person to ask them to fill in a survey. It is also harder to get back to people and inquire about the answers they gave.

Idea 2: Twitch streaming game

To come up with a solution to face tracking on distance, the idea to incorporate Twitch (a video streaming site for games) into the game was thought of. Because those streamers more often than not include a face when streaming, it is a good way to see how the person playing the game feels about the game. The chat could influence the game by determining what the player is capable of doing. The determining what the streamer can and cannot do would be changed per level. For example, the chat watching the stream could determine that the streamer can only attack enemies by shooting a gun and move. So in short the chat becomes the gamebuilder.

A big problem with this is to get in contact with the streamers and make the game interesting enough to have the streamers show the game. This idea would also require the most polished version of the game to get streamers to even consider putting the game on their channel.

Idea 3: Prototype cabinet

To test a game, it would be best if a lot of different players have access to it. The best way to do this is by having your game playable in a room where there are a lot of different people available. This idea has the idea to build an arcade cabinet which would showcase the game in a public room. The problems with this concept are roughly the same as the online distribution platform. The realisation pluspoints for this idea are the lack of needed internet presence and the ability to see players play the game.

Idea 4: Self learning game

Another idea that was thought of was an idea based on the game adjusting to the player. What if you play a simple game like Super Mario Bros. that will become harder or easier based on your performance and text input by changing the level, enemies or even game mechanics. The change could even be made so that a completely different game could be made. The idea behind this is to compare the different generated games and see how well tailored they are based on the feedback and performance given by the players.

This idea would be almost impossible to create in the time available to this project. Even if more time was available to build this idea, the level of the assignment itself would be very high and realistically not feasible to achieve in less than multiple years.

3.4 Final choice

From the four ideas mentioned above, the online distribution platform was chosen. Not long after the choice was made, it was deemed too hard to actually realise a reliable platform with the intended users in the allotted time. Instead of creating a distribution platform, the focus was on seeing if it is even possible to get the required feedback from participants. This is done by building a prototype of a game and tests incorporated into the game and see if it is possible to create a better prototype that is better than the one started with together with the feedback gathered from the players. The tests that will be incorporated into the game are based on the features that were deemed worthy in the state of the art section in chapter 2.

Chapter 4 - Ideation Game

To come up with a concept for the game, it was also needed to go through the design progress for the game that should validate the tests. This game could have been a game that already existed, but the choice was made to build a game from scratch so that the design process and life cycle of the game could be experienced while setting up the tests. It also helps to have a prototype of a game to incorporate the feedback of the players directly. This decision will hopefully make valid proof of the fact that it is possible to include users in the prototype phase of a game.

To decide what game should be made, it was eventually opted to go with an idea established before doing the test. The game will be an action-adventure game, where the emphasis will be on exploring different landscapes and attack combinations. The world will feel connected by showing parts of one level in the background of another level. The attacks the player can do will also be based on the enemies a player encounters. The goal of the game is to fulfil small missions that introduce the player to the world and the game play.

2D vs 3D

Because of time constraints it was deemed best to make a 2D game. The reasoning behind this is the fact that making a 2D game is much faster to prototype and building a game engine for a 2D game requires less complex mathematics and a easier to program rendering pipe. Another big pro for going with 2D over 3D is the way objects are animated. In 3D you work with 3D models which have to be coded so they can be loaded in next to the use of 2D graphics a 3D game also needs. With that taken into account, it will be more time efficient to animate characters per frame out of a sprite sheet.

Another big positive for making a game in 2D instead of 3D is the fact that a 2D game will run on most modern systems without the need of a lot of optimisation and a 2D game will also need a lot less resource management, which should offer tremendous less development time. Working with sprites and other 2D images will also grant the benefit of keeping the file size much lower than a game using 3D models, which will result in people downloading the game because the game is small.

Game ideas

During this stage of the project, a lot of different game concept were considered. These ideas can be found in appendix C, D and E. In the end it was decided to go with the concept that was worked out the best, and its gameplay concept could help make players return to the game. The game idea is explained in the beginning of this chapter and in the next session.

Recurring players

It would be ideal for the testing the methods to have users come back to the game more often, so more test data is gathered and it would also be a nice indicator of how well a gimmick is liked by the test players.

To make people come back to the game, several components could be utilized. The most common tools to make a player come back to a game can be found in the list below. These results are required after asking gamers what keeps them coming back to games.

With the help of these components the card system was thought of. In the game, players can find cards, which also function as attacks the player can use to attack enemies. By having a lot of different cards a player is given the tools to test a lot of different playstyles. Having a lot of cards also creates a new goal and challenge by introducing the need to collect all the cards[15].[27] Testing has to find out how much cards or attacks would be needed.

Another component of the card system mechanic is the fact that it allows to combine certain cards to create new attacks. This way there will be more emphasis on the deck building component mentioned before, because certain cards can influence each other so it would be beneficial to find and equip the correct cards in the same deck.

Aspects that returns a player to a game:

- Competition - beat friends or other players
- Achievements/working towards goals
- Social interaction

[27]S. J. Johansson, "What Makes Online Collectible Card Games Fun to Play?," Digital Games Research Association, 2009.

- Completion - loot
- New experience while replaying (non static experience)
- replayability
- Depth - discovering new things (even when already played a lot)
    - How many different ways are there to achieve the same goal?
    - What have I not tried yet? Is it a good choice? Is it a better choice?
    - What secrets did I miss? Is it worth exploring?
    - What new things can I create with the things I have in front of me?
- Lose yourself in the world (fantasy)
- Huge amount of content - Have a lot to do
- Different story branches - See the other side of the story
- No end point
- Find optimal routes in a game
- Investment

Chapter 5 - Realisation

In the previous chapters all the ideas were discussed. In this chapter everything that was actually made will be explained. During the duration of this research a game engine was made that incorporates several user tests.

Chapter 5.1 - Game engine

During the development of the game it was decided to also create the game engine from scratch. To idea behind this decision was the possibility to include certain tests more accurately. How the game engine was constructed, can be read in this part of the chapter.

CPP

C++ is a well known and commonly used programming language for creating games, which is great for low-level memory manipulation. Having easy access to the program's memory is also the main reason why C++ was chosen as the programming language to build the engine in, because the language makes it easy to the measuring functions without creating too much overheat. Furthermore C++ has a great array of libraries to create a game out of.

While other programming languages like Java or using pre-existing engines would be a valid option, the ability to make a snapshot and recording of the memory was a feature that should be included. This way a perfect record could be made of enemies and players. Unity in itself does not support this feature and would require a lot of a lot of extra programming works and hacks to make it possible[28] [29] [30]. This way of implementing the future would most certainly also influence the performance of the game, which would lower the user experience.

C++ is chosen over other languages like Java for its ability to manage the memory and performance. The available examples and tutorials for OpenGL in C++ in comparison with the same content for Java was also a reason to go with C++ over Java.

[28]http://www.gamasutra.com/blogs/AustonMontville/20141105/229437/Implementing_a_replay_system_in_Unity_and_how_Id_do_it_differently_next_time.php
[29]http://www.gamasutra.com/view/feature/3057/instant_replay_building_a_game_.php?print=1
[30]http://gamedev.stackexchange.com/questions/6080/how-to-design-a-replay-system

Chapter 5.1.1 - Libraries

The engine was made from scratch with as little premade libraries used as possible. The reason behind this was to keep the engine as simple as possible.

Open Graphics Library[31]

While it is possible do software based rendering, it would be easier to use an existing library. Open Graphics Library (OpenGL) is the most used application programming interface (API) for 2D and 3D graphics applications. One of the biggest advantages OpenGL has, is that the library is compatible with the most used operating software out there.

Glad: Multi-Language GL/GLES/EGL/GLX/WGL Loader-Generator[32]

To actually use the functions of OpenGL, a OpenGL Loading Library[33] is needed to handle the actual loading of OpenGL functions. This extra library makes it possible for OpenGL to be cross-platform. In the application glad fulfils the need of loading the OpenGL functions at runtime.

GLFW[34]

OpenGL also needs a window to project the graphics to a so called window context. GLFW is a multi-platform library for OpenGL, that is compatible with Windows, OS X and other Unix based systems like Linux. The main functions this library can handle is creating a windows context (a window in the operating system), poll or do callbacks for user input over keyboard, mouse and gamepads.

GLM[35]

OpenGL Mathematics (GLM) is a small C++ library that covers most if not all of the needed basic mathematical functions needed when dealing with programming a game. These functions contain but are not limited to dealing with matrix, transformations, quaternions, half-based types and random numbers.

[31]https://www.opengl.org/
[32]https://github.com/Dav1dde/glad
[33]https://www.opengl.org/wiki/OpenGL_Loading_Library
[34]http://www.glfw.org/
[35]http://glm.g-truc.net/0.9.7/index.html

Freetype2[36]

Freetype is a small and often used library for displaying font based text. This library is also used for Android, iOS, OSX and modern PlayStation devices.

JsonCPP[37]

During the design stages, one of the features that would be nice for the engine was the ability to update data from the game without recompiling the whole game. To incorporate this component into the engine, the engine needed to have a way to read and edit a configuration file. The JavaScript Object Notation (JSON) data type was chosen over other data formats e.g. XML or .INI, because the format supports nested data and is also lightweight. To manipulate the data format, the JsonCPP library was used instead of writing an own JSON interpreter.

SOIL[38]

To load images and textures into OpenGL textures Simple OpenGL Image Loader or SOIL in short is used.

5.1.3 Incorporated tests

In this part of the realisation chapter the incorporated tests are included. These data gathering techniques were chosen for their usefulness and potential they could have. Some functions like the changing data during run time were added to make the tweaking of the game easier, these functions could also be used during tests.

Metrics

The current engine is able to keep track of different user data and at the end of each play session the game saves the data gathered into a file. The data that is recorded varies from the input the player gave on the controller or keyboard to how accurate a player was with a certain attack or how long the player was on the ground. All these statistics are so called metrics. The metrics offer valuable information and insights on why a player did certain actions during a play session. The metrics are not translated to a visualization, so the raw data it generates right now, still needs some improvements.

[36]https://www.freetype.org/
[37]https://github.com/open-source-parsers/jsoncpp
[38]http://www.lonesock.net/soil.html

Replay play sessions

As mentioned above, the game tracks all the buttons pressed by a player. The recorded button presses can be used to make a playfile of a play session. In other words, it is possible to replay the a whole play session of the input. This can be done because the game keeps track of every frame and works in such a way that the game keeps track of how long it took to generate two frames and adjust the logic in such a way that the changes over second would be consistent even though the time for each frame could not be. To make sure the recorded play session is even more accurate, the framerate is locked to 60 frames per second. With this, it is possible to gain data from previous tests if for example extra metrics are added to the game.

Ad Hoc testing

At the start of the game, a player can play around with his or her character and do all the moves the character is capable of doing. The metrics are not recorded in this part of the game, so the player can experiment with the game and its mechanics before starting the actual play session. Saying that none of the metrics is being recorded is a lie because the input in this stage is being recorded. The reason for this being that the input can be replayed in a one-to-one version of the original test, so if in the future it is deemed necessary to track a certain attribute during this Ad Hoc testing phase it is possible to do so.

Changing game data during runtime

While the game is running, it is possible to change the level layout and the attack data of the game. These functions make it possible to do special tests in the future, where the level layout or attack data could be tweaked or generated to adjust to a certain player.

Chapter 5.2 Game

During the duration of this project, a game was build on top of the engine which is explained in detail above. The game was supposed to be an action platformer, but due to time constraints that idea had to be scaled down. What has become of the game and its implementation can be read in this part of the chapter. A screenshot of the game made in parallel with this research can be seen below.

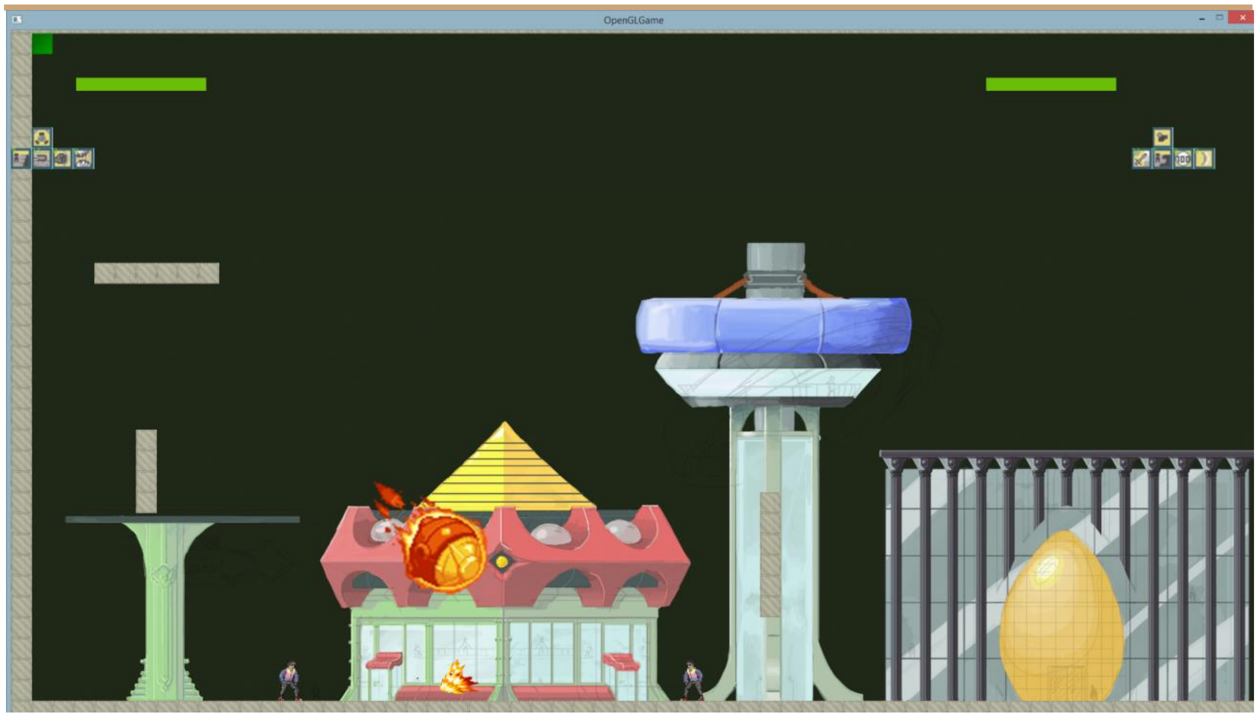


Image 2: Screenshot of a playable state of the game

Goal

Every game should have a goal, and the game made during the course of this project is no exception to this. While the original goal was to have a player reach the end of a level to progress with a small story arc, it hat to be changed to beating the other player before your own player bites the dust. Every player has three lives, which can be lowered by depleting the health bar of the opponent's character.

Grid based movement

While it was known that the game would be a 2D game, the way a 2D game works could be realised in a lot of different ways. The solution chosen for this game was to make it a so called tile-based video game. This means that all most of the games logic is based on a grid and where every square has impact on the behaviour of all the game objects inside its surface. While this technique is commonly used for video game since the nineties and noteable engines like Construct 2[39] and GameMaker[40], it is also a quick and easy to implement. While the game's logic is based on the squares, GameObjects like the playable character are not restricted to a square and can move freely over it. Static level objects like

[39]https://www.scirra.com/

[40]http://www.yoyogames.com/gamemaker

platforms are, so it is a must to design levels and their objects in such a way that they fit the size of one or multiple squares.

Genre

The game itself also changed from being a platform-adventure game to something that would classify itself better as a multiplayer brawler game. The reason for this big change was the fact that making the engine took much more time than anticipated, which resulted in less time to actually work on the design of the game itself. Making a brawler out of the game was a simple and effective way to create simple fun in the form of competition. It also lessened the workload by removing the need for additional enemies, and elaborate level and game design.

Chapter 5.2.1 - Gameplay

One aspect of what makes a game a game is the ability to interact with the game world. How players can manipulate the game world and its rules is explained in this part of the chapter.

Movement

The game's origins as a platform game can still be found in the movement and the controls of the playable character. The character has access to all the movements one would expect from a character in a platformer like running with an acceleration, jumping, double jumping and wall jumping. The character can also change in size to for example duck or have a transformation state. In the game's current form these mechanics are not being used.

As addition to the normal movement options, the playable character can also has dashing and dodging movements. For the brawler expect these movements are a way to give the gameplay more depth. In the tests with the game, these movement are not used because the combat system in itself was deemed too complex for the players already. Adding even more moves for a player to utilize would only end up in making the game even more confusing than it already is at this stage.

Combat

The combat system is the main gimmick of the game. The way a player has a broad range of different options to his or her disposal is something not done that often in games. If it is done, it is never combined in a game where the action is not paused. Giving the player access to so much different options in itself is also a problem in itself because it generates a high entry level to understand the game. In the following section the combat system is explained.

Cards

In the game, a player can perform attacks to damage and kill enemies. An attack is illustrated as a card. Every character has its own deck of cards. The deck of cards can be composed entirely by a player, so a player can determine which attacks and in what order they are stacked in the deck.

Once the player is actually playing the game itself, the player can access all the cards of the deck by cycling through the deck. At all times only one card of the deck is selected. This card can be used to be assigned to one of the four attack buttons. Once the card is assigned, the player can actually use that card with the button it is assigned too to attack. A player can “equip” four cards at the same time.

Once an attack is activated it will continue till the attack is done, hits a wall or collides with another attack. What attack will be done depends on the card that is used. A card is categorized in a certain attack group that will determine what behaviour it has. This means that a card placed in a sword group, the card will do a short distance attack, while a card in the ranged class will do a ranged attack. The games has a wide diversity of groups and combined with the categoristics of a card, it gives access to a lot of different combinations to use. Every attack also has an element and can afflict a certain status ailment to the player.

Another gimmick this card system has, is the ability to combine cards. If a card is assigned to an attack button and the player assigns another card to the same card, there is the possibility the two cards form a new card. This card is made if the two cards meet a certain requirement specific for that card.

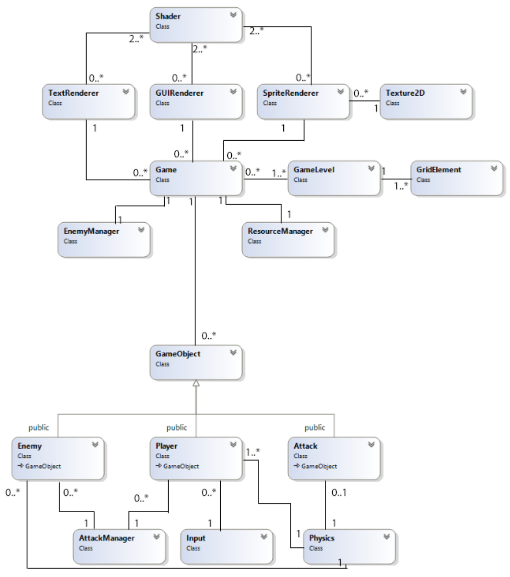
Chapter 5.2.3 Story and setting

For the overall style of the game, it was decided that it should be retrofuturistic. The era chosen as a base for this artstyle is the eighties. A moodboard to get a general impression of the style can be found in Appendix F and G. Naturally the retrofuturistic can be found

back in the games assets. How the style has been reworked into the assets can be read below. The character is based on how people dressed themselves in the 80's The style of the character can be found back in one of the pictures of the moodboard seen in Appendix G.

Chapter 5.3 Design program

As can be seen in the figure below, the code consists of a renderer part and a game part. The renderer makes sure that all the parts that should appear in a screen, like text, images and basic shapes are actually placed inside a window. The game part governs all the game logic and sends all the necessary data to the renderer, so the renderer knows which objects has to be drawn onto the screen. The two components will be discussed later in this chapter in more detail.

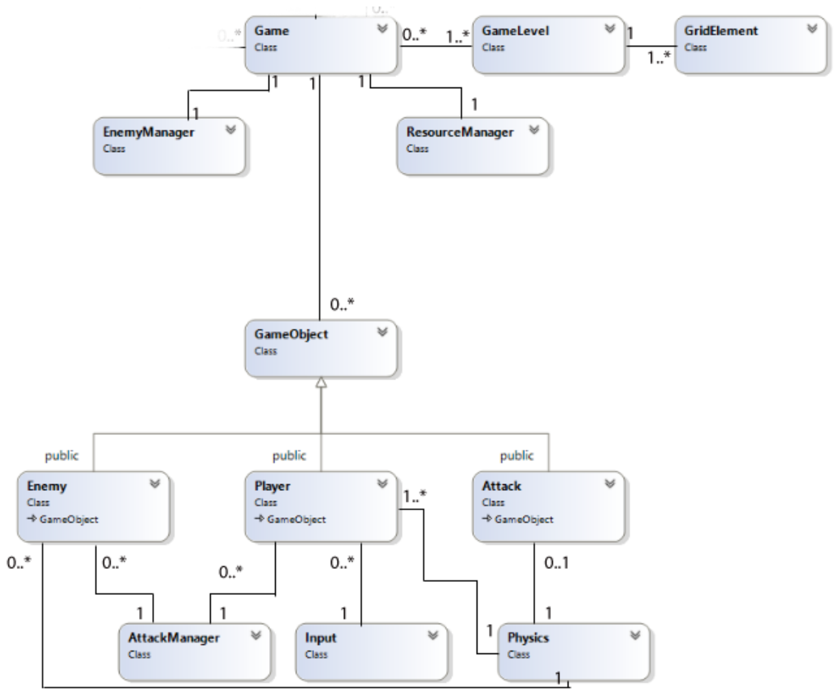


Simplified class diagram

5.3.1 Game

Manager

As already mentioned earlier the game part of the code governs all the logic for the game and communicates the relevant changes to the renderer. The game does this with so called managers. All relevant objects of a certain type (i.e. a GameObject) is stored inside a map or arraylist inside one manager object. All the objects inside the manager can only be manipulated through the manager in question. The GameLevel class can also be considered a manager class, because the GridElement on which the levels are based are stored in this class. The AttackManager is also worth mentioning because the GameObjects that can actually attack, do so by sending a number to the AttackManger, in which the AttackManager will create a new Attack and take care of it because the AttackManager also has the capability to load in and store a file that contains all the data for the possible attacks of the game. The ResourceManager contains all the shaders and textures, while the GameObjectManagers are stored in

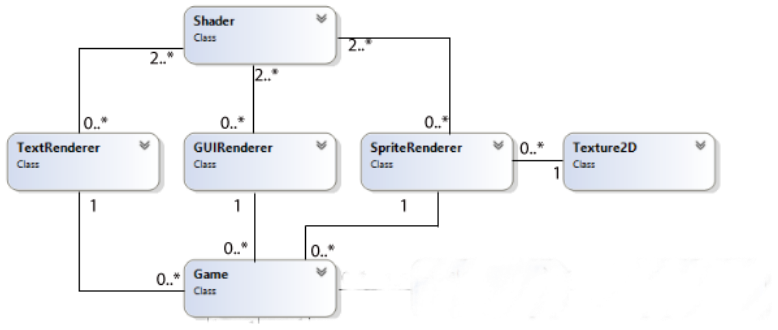


GameObject

A GameObject inside this context consists of all the objects in the game that can interact with something else and should be rendered on screen. In the game those objects are the playable character, enemies, attacks and platforms a player can interact with. These GameObjects can also contain other optional components. Examples of these components are the input component, which like the name suggests, makes it possible to move a GameObject while playing the game, or add a physics component to make collisions between objects possible.

Chapter 6.3.2 Renderer

When the game is done with calculating the new game state for the upcoming frame, the game will send the data to the renderer, which will call one of the three renderers (Text, GUI or Sprite) to make a graphical representation to send to the screen.



SpriteRenderer

The SpriteRenderer is the core renderer of the program. This renderer is responsible of drawing all the sprites in the game. This is done by rendering a single quad (two triangles placed in such a way that they form one square) with a part of the texture (which is UV-mapped on the two triangles. Animations are done by changing the UV coordinates of a quad so another part of the spritesheet is shown.

GuiRenderer

The GUIRender is capable of rendering basic shapes. The name GUIRenderer should be renamed, but this renderer was used to make simple GUI elements like the healthbar.

TextRenderer

The TextRenderer is solely based on the Freetype2 library to make it possible to load in fonts. The rendering part itself is mostly the same as sprites are rendered, but keep the relative height into consideration when placing the character on the quad, to take the different sizes of all the letters into account.

Shader

Shader is a class which governs all the shaders files used within the program. It functions similar to that of the manager classes explained above. The Shader class can also compile and bind (make the shader active) the shaders so it can actually be used in the rest of the program.

Chapter 6 - Evaluation

The tests were conducted on 13 testers ranging from age 18 to 24. The average age was 20.8 years old. The ratio between male and female was 7:1, where 1 female and 2 male gamers admitted they played games daily. The most common device the game were played on was the PC, with 12 testers stating they played on the device. The group of testers are also well versed with consoles that are 10 years or older. The most played game genre the testers played are Adventure games. Role-playing, Action and Strategy games are also very common types of games played by the participants. This means the tester group fits the intended target group which aims at players that are familiar with older adventure games.

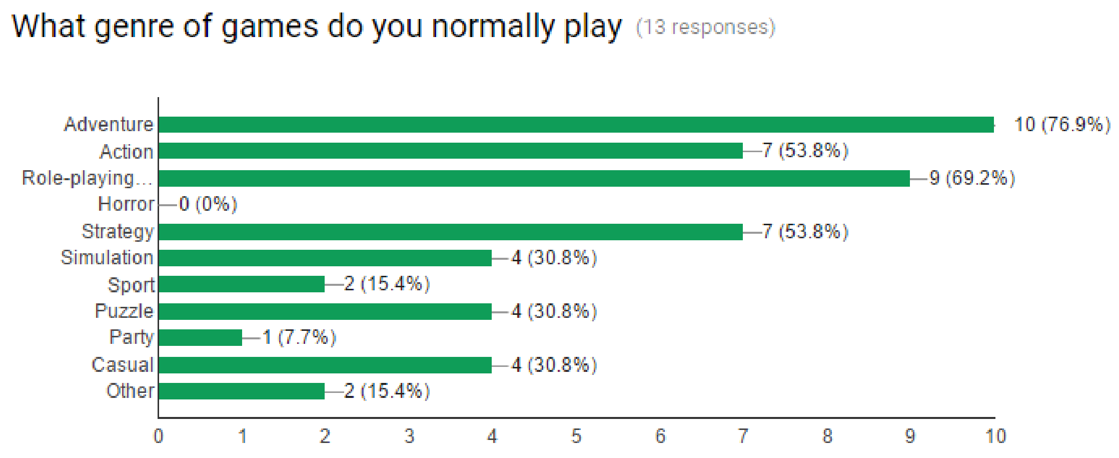


Image 3: A brief overview of the game genres played by the testers.

The test was conducted by asking a tester to fill in a questionnaire. Once a tester was done with the questionnaire, they were asked to take place behind a computer that showed the game. They got a brief explanation about the goal of the research, the game and on how the controls worked. At this point the tester could play around with the game till they stated they were ready to take on another player. Once the player says they are ready, the game will reset the player and the game will start. At this point two players have to kill each other and the first player to gain three kills will win the game. Once the game was done, the tester was asked to fill in another questionnaire and once it was done the tester was thanked for participating.

Chapter 6.1 Ad hoc

For the way the test was conducted, the ad hoc method made a good tutorial substitute. The players got a brief explanation of how the game worked, while they could also try it themselves at the same time. The explanation the participants got was limited to the goal of the game and how the controls work. The more in depth mechanics had to be found out by the testers while playing the game. The testers could say when they felt accustomed enough with the controls to start playing the actual game.

Most testers used the ad hoc test to find out what all the cards and attacks did. The amount of time spent was varying a lot between the testers. So did one person start the actual game as soon as he had a feeling for the movement controls, while others planned out the order of the attacks so they at least knew what attack they would have when. Seeing how people “memorized” the attack sequence instead of relying on the user interface was surprising, but not unexpected because the interface of the game is unclear. It were these kind of testers that found out about mixing cards to create new attacks and tried to play more skillful and with thought behind their actions when playing the game. People that did not took long in the ad hoc testing phase tried their luck with hitting as many attack buttons as possible in the hope that they could win the game by having a lot of attacks on the screen on the same time.

The ad hoc tests was also a great way to interact with the tester. While they were fiddling around to get a grasp of how the game works, they did not mind questions like why did you try that to get a limited feel of how players would act if they tried the game for the first time. Also frustration points and points of confusion came to light when players started making sounds like “Aha”, or ask why something did not work as intended when they tried to do something several times. Because of the interaction it was hard to not defend certain design choices made for the game. This became harder when questions like; “Why did you not implement the cards by making them spawn over a certain period of time” or “Why did you not use that button for the controls”.

Chapter 6.2 Metrics

Collecting data within the game during the playtest in itself provided a lot of raw data. As an entity of its own the raw data does not offer interesting data, so the large amount of information had to be translated into something more readable. To understand the large stream of data it was chosen to make data visualisation out of the data. For data like the positional data of each character a more visual description (like seen in image 4) was created, while other data was deemed more useful to be represented in a table with other data. An example of such a table can be found in table 1 which contains a collection of all the attack statistics of one player.

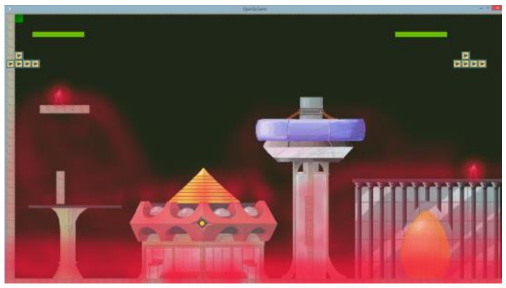
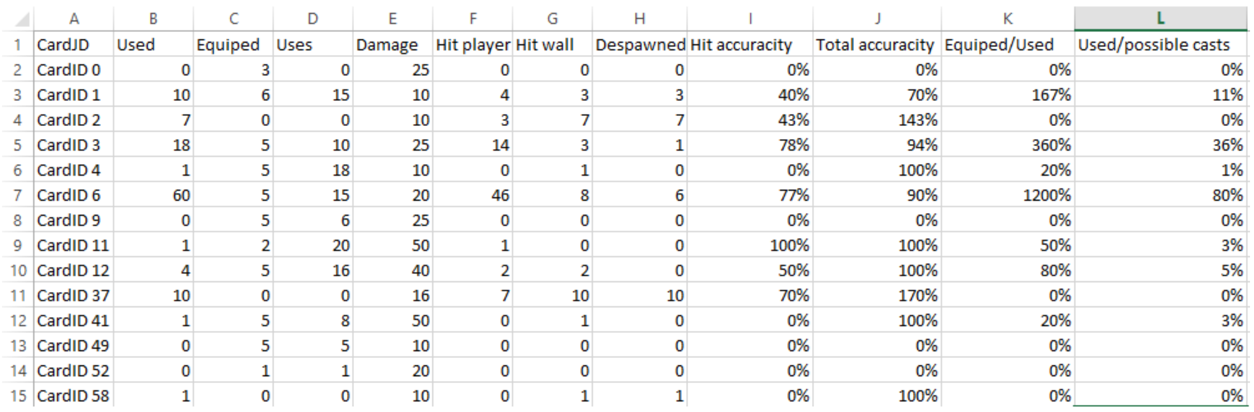


Image 4: Heatmap of player positions

In table 1 one may notice that some attacks can be used without being equiped. This has to do with the combo system explained in chapter 5.2.1 - Gameplay (Cards), which can activate a new attack after an attack is finished. So one card ID can activate itself and another ID while only executing one attack.

While the data the metrics offer can be used to lead to new ideas, its usefulness lies in uncovering problems in the game. An example of this is how in the game used for the tests how players could not reach the upper platforms anymore after a change made to the prototype before starting the actual tests. This change was a small balancing fix that made the playable characters a bit slower so they would be easier to hit. The lost in maximum velocity also meant it become hard to get on top of all the platforms. Because of the familiarity with the game, this issue was not noticed till after the tests were done. After the heatmap (red is where a player was, the intensity of the spot indicates how often the players was on that particular spot) was made, it was clear that the majority of the time the characters were on the lower level of the map. When watching some replays of the game tests, it became clear that players could not reach the top level of the stage anymore.

Like mentioned above, it is possible to generate new ideas for the gameplay-aspect of the prototype. Metrics can be used to see which elements of the game are liked or not by seeing how often a player interacts with a game element. During the test the game gathered the attack behaviour of every player by counting how often an attack was used and equiped. Together with the data of how accurate a tester was with the attack, an indication can be collected of what types of attacks the players liked. Based on other facts the developer knows about the game, like how strong or fast that certain attack is, the metrics could serve as a base of why certain attacks work and why others don't. This knowledge can be used to come up with new game concepts or situations where the prefered elements could shine more. The gained insights can also fix other attacks by incorporating the factors the developer thinks makes the other more fun or useful to use. 2



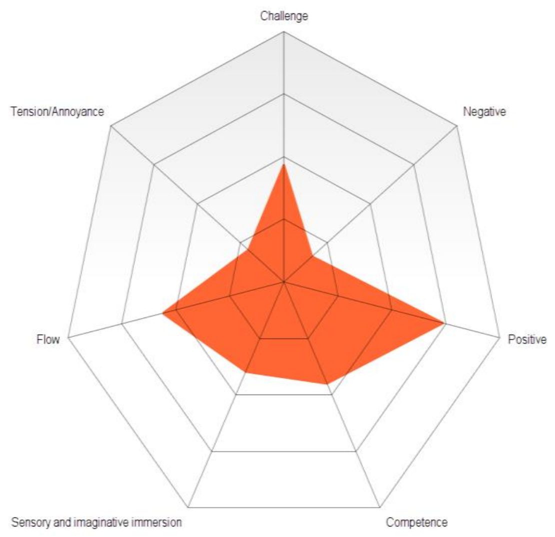
	A	B	C	D	E	F	G	H	I	J	K	L
1	CardJD	Used	Equiped	Uses	Damage	Hit player	Hit wall	Despawned	Hit accuracity	Total accuracity	Equiped/Used	Used/possible casts
2	CardID 0	0	3	0	25	0	0	0	0%	0%	0%	0%
3	CardID 1	10	6	15	10	4	3	3	40%	70%	167%	11%
4	CardID 2	7	0	0	10	3	7	7	43%	143%	0%	0%
5	CardID 3	18	5	10	25	14	3	1	78%	94%	360%	36%
6	CardID 4	1	5	18	10	0	1	0	0%	100%	20%	1%
7	CardID 6	60	5	15	20	46	8	6	77%	90%	1200%	80%
8	CardID 9	0	5	6	25	0	0	0	0%	0%	0%	0%
9	CardID 11	1	2	20	50	1	0	0	100%	100%	50%	3%
10	CardID 12	4	5	16	40	2	2	0	50%	100%	80%	5%
11	CardID 37	10	0	0	16	7	10	10	70%	170%	0%	0%
12	CardID 41	1	5	8	50	0	1	0	0%	100%	20%	3%
13	CardID 49	0	5	5	10	0	0	0	0%	0%	0%	0%
14	CardID 52	0	1	1	20	0	0	0	0%	0%	0%	0%
15	CardID 58	1	0	0	10	0	1	1	0%	100%	0%	0%

Table 1 Metrics data of the attack pattern of one tester

An example of how to get an insight how developers can get an understanding of how the player interacted with the card system in the prototype can be understood with the help of table 1. In table 1 it can be seen that the particular player used the attack with CardID 6 a lot (80% of the possible uses was done). When looking at the other data, this attack does relatively a lot of damage while also being accurate and having a lot of uses per equiped. This player's other most used card is also a card that has a lot of uses, but the attack speed (which is not shown in the table but could be found in a general attack table used by the game itself) of the attack with ID 6 was four times faster. Based on this it can be assumed this player did search for the cards that did a lot of damage and have a lot of uses per card. The amount of equiped cards with no or almost no uses also seems to support this assumption. A lot of additional data, like the accidental use of the combo system could be gathered from this simple table in the same way.

An important issue found when using metrics is that the data should be saved efficiently. The game collected all the data and pumped it into one file per play session. The way the data was collected in the prototype was far from ideal, so a lot of time went into cleaning and separating the data in different categories, before the data could be visualized. In hindsight it would have been better if the data per play session was also saved in different files.

Chapter 6.3 Questionnaire



To conduct interviews the Game Experience Questionnaire is used. The questionnaire contains three modules; the core questionnaire, the Social Presence Module and the post-game Module. For this research only the core questionnaire is used. The core questionnaire assesses the game experience as scores on seven components: Immersion, Flow, Competence, Positive and Negative Affect, Annoyance, and Challenge. The questionnaire has to be filled in immediately after a game session has ended to have its fullest effect.

The average scores for the seven components produced through the Game Experience Questionnaire can be found in the radar chart next to this text. The questions of the Game Experience Questionnaire and the answers given by the testers can be found in Appendix H. The average data can be found in Appendix I.

Competence

The average scores for the seven components produced through the Game Experience Questionnaire can be found in the radar chart next to this text. Based on the average score, the questionnaire indicates that the people that played the game did not feel competent when playing it. When looking at the individual scores, there is a clear split between people that scored pretty high (above a 2) and pretty low (1 or lower).

Immersion

With the exception of two testers the players thought the game had a low sense of sensory and imaginative immersion. That is not a big surprise if one takes into account that the prototype does not contain any story elements and the only sense of immersion could be achieved by a combination of the art of the game and one's own imagination. It is interesting to note that most players could create some immerse themselves with only having the implication of a story.

Flow

With the exception of one player, every player could come into a state of flow.

Tension/Annoyance

During the tests, there was only one tester that got really got annoyed by playing the game. This could have multiple reasons like the lack of story, not understanding the controls. While there were some more players that were mildly annoyed, the overall consensus was that the game score relatively well on this aspect.

Challenge

This part of the game received mixed scores. The people that liked to attack as fast as possible did score the game a lot lower on the challenge part than people that used a more calculated playstyle. Out of this small sample, it does not seem that winning or losing did factor into how challenging the player experienced the game. Overall the game scores a bit under average for challenge.

Positive and Negative Affect

With an average score of 2.95, the tester did experience the game as rather positive experience. There is one participant that really did not like the experience.

Chapter 6.4 Interview

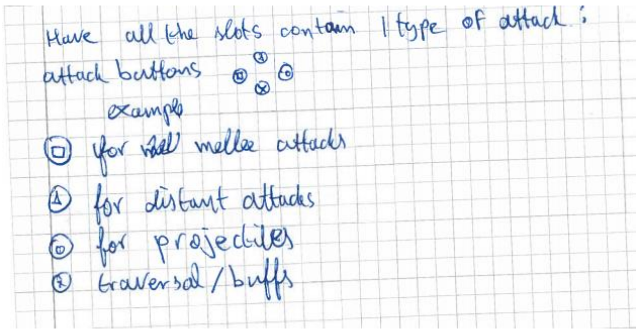
To end every test session a small interview was conducted. Every participant had to answer the following questions; What did you think of the game?, What would you like to see changed to the game? and How would you change the game?. Below are the most valuable insights and tips gained through the interview. A transcription of the interviews can be found in appendix J.

The most prominent reaction the interviews pointed out was that the Graphical User Interface (GUI) was not clear. According to the testers it made the game harder to play and understand. It was also the reason why most of the players started to click the buttons as fast as possible to do attacks instead of choosing the attacks more appropriately. There people that took more time in the ad hoc testing phase did try to find the attacks they liked most during that test to play the game, but they also agreed that it was hard to find that attack during a match through the GUI.

The overall consensus of the game was that it was fun to play. Because the play session was short the people testing the game could not say if they would still like the concept if they would play it for a longer period (for more than 30 minutes or several hours). The concept to have a lot of different attacks was thought of being fun. A problem with the huge amount of attacks is, according to the interviews that it also make the game a lot more confusing. It also did not help that the controls were not designed to fit a keyboard, but the interviewees that are familiar to playing games on consoles or with a game controller could understand how the scheme would fit a controller. Most suggestions for fixing the controls controls when playing on a keyboard involved a layout that resembles those of a first person shooter's weapon equip scheme.

One person also missed the story and would have liked to see it incorporated to explain the controls, the reason why they needed to fulfill the objects of the techdemo and to get a feel for the game's world in general.
Like expected the interviews led to ideas that were based on the original concept. So did one of the participants suggest to change the game so that it mimics a shooter and only give to option to use one weapon at the same time, instead of having four attack buttons like it is the case right now. This would also mean that the card game gameplay needs a rehaul. Combining cards would be much harder because one cannot have another attack option, while waiting for the correct card that is combinable. This in itself could also give

some interesting game play situations however, because the player can be forced to go in a defensive playstyle.



Have all the slots contain 1 type of attack :
attack buttons
example
□ for ~~all~~ mellee attacks
△ for distant attacks
○ for projectiles
✕ traversal / buffs

Another tester came to the conclusion that the amount of option was to overwhelming and that it should be limited. The way the tester visualized the idea was to reduce the four attack buttons by one, and limit the button to one attack categorie, so attack one would be close distance attacks, attack 2 could be for mid-range attacks, while the third button could be for used for projectiles. The fourth button could be used to map traversal influencing effects.

To further flesh out this idea, the tester complimented his suggestion by replacing the other attacks with different attack attributes or properties. So if a player would have a stone that is thrown with an arc on attack button 1, the attack could change to a grenade by adding a card on that button that has an explosion attribute. This way the game would more focus on the combining aspect of the card system, which was something that by various was missed by some testers in their play session. The concept of the idea is something that is well liked, but obviously as with the other suggestions gotten through this method, the suggestion need to be rebalanced to be playable. This idea however can and should be used in a second iteration of the game. A/B testing between the old and new idea would be a possible way to determine which of the two systems is preferred by the testers.

A different person also came up with a solution to the abundance of choice the current game play integration gives. This person had the idea to limit the access to the deck by pausing the game every minute. The player would only have access to 5 cards from the deck at a time. This would mean that every card can only be equiped once. When the game pauses, the deck would refill to have access to 5 cards again. Effectively this means that a player can have access to the four cards assigned to a button and the 5 cards received from the deck. Of course the pause could also be removed and the cards could be added automatically.

Chapter 7 - Conclusion

Gameplay idea's:
3 fix'd cards:
have deck of cards that can develop your start attack
Has a high lev
etc.

The purpose of this research was to see if it would be possible to include participatory design into game design. During the duration of this research it became clear that incorporating users from one's target group can become a valuable source to get an evolving iteration design with the feedback provided from the users. Even though players that played the game were no designers themselves, they did had enough knowledge to point out the fault of the game and they could make themselves clear when discussing new concepts while interviewed.

Within this research it turned out that, while limited in time and test subjects, the correct interaction with the users could be established. It is indeed possible to gain the feedback in the prototype phase needed to make a new sensible iteration on the prototype, without needing much additional input from the designer themselves. The ad hoc test highlighted the problems that arose during the first encounters of the game and which elements of the game were not clear enough.

The metrics held a similar function and helped to support the problems found in the game with actual data. Because the metrics are a large dump of information, the data should be cleaned and preferably use to be used in a visualisation to be most useful. The data in itself does not offer any new gameplay ideas in itself, but it can give insight to which elements of the game are liked most to make sure the design could be emphasised on this aspects. That in itself could be seen as a big influence of the overall gameplay.

Incorporating the Game Experience Questionnaire was used to get a base impression of how well the initial prototype was experienced. The scores obtained through the questionnaire proved the base gameplay was liked but still had room for a lot of improvement. Based on the last fact it means that the game itself was good enough to conduct the test on.

The interviews were a great way to let the testers give their own interpretation of the game. With one simple question it was possible to give rise to a lot of interesting takes on the main concept. These new ideas were not incorporated into a next iteration as was intended because the realisation of the game engine took too much time which left no time to do additional iterations, so it could not be tested if the gameplay ideas gotten through this method have an actual value, and if they are able to improve the overall quality of a game.

The original goal of making an online distribution platform to make a gain the feedback needed, was not achieved. The question remains if the four methods to gather the required feedback mentioned above can be achieved the same way as is done in the offline versions. While the ad hoc session was very useful because the testers were on the spot, it is not clear if the method could offer the same result if the tester is observed through other means. The interview method worked well, because the participants could be guided to concrete their ideas with the help of pen and paper or gestures. It is not clear what happens to the general quality of the ideas the participants come up with if this option is not available.

All by all it is possible to gain the data necessary during the prototype phase to make a new iteration. If the suggestions of a player makes a better prototype is still a question and should be done researched in a later stage.

Chapter 8 Discussion

During the duration of the project too much time was spend on realising the game engine. This means that the overall quality of the research was diminished and the actual goal of this research was not met. This all would not have been a problem if the reason why the decision to make an own engine (to create a perfect snapshot of a playtest) was actually realised. The way it is incorporated into the engine now, is not something that could not have been done in an existing game engine like Unity3D.

Another problem with the research was the lack of knowing exactly what to do at the start of the research. This means a lot of time was spent on research that was not the main focus of the actual bachelor thesis.

Chapter 9 Future Work

The most important thing that should be done in the future is to carry on this research and see determine if the game ideas given by the players really has added value based on the base concept. The tests could be conducted with an A/B test as explained in the current report, or by using the Game Experience Questionnaire to compare the scores of different prototypes.

While the Game Experience Questionnaire was a useful source in getting concrete scores of a tester's user experience, the questionnaire was not liked by the testers. It was too long. Also the questionnaire should be placed inside the game to make testing over the internet possible. A problem with this is that people will not fill in the survey if they do not need to do it. If the questionnaire becomes mandatory by keeping further process of the game behind the questions of the questionnaire, it could disturb the game pace or have the effect that players stop playing the game. That last one could also be an interesting metric in itself because the players not playing your game for something so trivial would also mean that they did not enjoy the game enough. All by all, the effects of having the questionnaire in the game itself should be tested in a future study.

It would also be interesting if more tests can be found that would work in coherence with the tests used in the current prototype. At the moment the way the data is tracked, most of the data if not all is unique to different aspects of the game. If other tests are included, some of the tests could be looking at the same aspects which means that one of the tests is being redundant. It could also be by adding a certain test another one works better because the two tests complement each other. In a future study more combination of tests should be tried and the effects of those has to be recorded.

Another thing that has to be done in future research, is finding other ways to gather game ideas. In the current prototype the most successful way to gain new game concepts was through the interview. Even though the same three questions was asked to every participant, it was the ability for the tester to draw or explain with gestures the ideas that it was so successful. These options are not available to a tester if the interview would be switched with three static questions online, so there also has to be found a fix for that problem.

Appendix A - Literature Research

What aspect of a game makes a game fun? The answer to this question is something that keeps developers and designers behind a game occupied during a game's lifecycle. While fun is something subjective, a general consensus on if a game is fun can be reached by doing extensive user tests. How these tests focus on the interaction with the test participants, can be found in detail in [14]. Bigger companies also use much grander approaches, which do not shy away from using methods that include eye and heart monitoring [2].

Public beta testing, a test where developers put on online version of a game in production, can get millions of testers who give a lot of feedback, just by making the beta accessible to the general public [2]. These tests can give a good insight in a player's do's and don'ts, which can be used during a game's development to make the product better. The problem with a public beta for a game is the fact that the game is relatively close to its release window, so the data collected will not bring any major changes to the core gameplay. The recently timed alpha for Nioh[41] shows that developers are considering to release far from finished games to the public and ask input from their player base.

For independent developers and big companies it could be beneficial to have a user test that would combine the usefulness of the extensive users tests and the online playtesters that are eager to play the game. Even more so for the smaller studio's which do not have the money nor time to conduct extensive user tests. To design a test, which can be used during the early stages of a game's development cycle, guidelines need to be compiled.

To validate the test resulting this bachelor project, a game will be build up simultaneously. This game will be tailored to the needs and wants of the test to try and see if the designed test will indeed show points for improvements in the game. The main goal of the bachelor project is to find an answer to the research question: "Which guidelines need to be taken into account when designing an online quantitative playtest which focuses on video games?".

[41] http://teamninja-studio.com/nioh/

Before any guidelines can be constructed for a playtest, the understanding of which elements of a game makes a game fun has to be gained. This literature review will try to find the characteristics of a game that makes the medium so fun to play,so the real purpose of this review is to answer the sub research question “Which characteristics of a game makes this form of multimedia fun?”. This starts by finding out how a player interacts with a game and how the interaction makes the game fun. These findings will than be translated to be the cornerstones

What makes the interaction between a player and video game enjoyable?

The interaction between a player and a game can be divided in multiple layers. In Clanton’s [3] understanding there are three levels, which are Game interface (what the player sees), Game mechanics (the rules of a game) and Game play (how the player interacts with the rules). Where Clanton also states that most games have interfaces that CHI designers would consider obvious flaws. Some examples for the problem he states are hard to navigate menu’s, unclear instructions and games that do not feel nice to control. Nowadays these flaws are less prevalent but can still be encountered in modern video games. While a flaw in the user interface could not be that obtrusive, a flaw in the game’s mechanics is [3]. Clanton continues by saying that game play is about overcoming problems, so problems in the game interface and mechanics can be masked by the fun of solving problems in the game. The reason for this according to Clanton is the fact that people enjoy solving problems and love to overcome meaningful obstacles. In short the interaction of a player with a game contains three levels, whereas game play is the most important level.

Goal

For game play to be enjoyable, a game should have a goal. According to Crawford [4] games should not be solvable, but there should be an illusion of being winnable. Federhoff [5] also agrees in a way to a certain degree by stating that a player plays a game to achieve a goal points out that a game’s goal should offer challenge. On the other hand, in the aforementioned paper a statement of a director that participated in Federhoff’s study was mentioned, that a game should ultimately be solvable. Malone [6] also agrees in a way by stating that in order for a game to be challenging; “it must provide a goal whose attainment is uncertain.” Pagulayan et al. [7] not only mention that a game should have clear goals, but that these goals should also be interesting to the user. All sources agree however that the concept of being winnable (albeit being an illusion) is important to a player. Chuck Clanton [3] states a goal should be set early in the game, but can be changed during the duration of the game. Desurvire et al. [13] agree with the statement Clanton made [13]. Jegers [8] even states that concentration on a game is possible if the tasks in a game is clear. So it seems that having a challenging goal introduced early is quite important for a player to get into the game.

How to create a fun goal?

To have a goal that is fun for the player different methods could be used. According to Malone, score-keeping is a good way to have a goal that reflects the player's progression. Federhoff [5] complements this statement by stating that a score is form of positive feedback to encourage players to enhance their mastery of a game. Jegers [8] adds to this, that a player should always know their status or score. Platinum games[42] made good use of this knowledge by letting their players know for what they attained a score, so the player knows what should be improved. It seems that adding a score is a good addition to make a goal fun, which also add direct feedback to the player.

Another way to include fun to a goal is by creating a challenge. Challenge can be introduced to a goal by setting a condition that something has to be done as fast as possible [6]. Clanton [3] agrees with the previous statement by saying that pressure can be fun, and mentions adding a time limit to the mechanics of the game as one of the means to introduce the pressure. The last statement is powered by the following quote: "Note that in the extreme case, an unintelligent opponent may generate an exciting game just because of the unrealistically fast reaction time required of the player." [10]. This all seem to agree with the observations of Federhoff [5] to include a challenging goal. This establishes that another method to make a goal challenging or fun, is to add a condition or time limit.

Curiosity

Curiosity is also a factor that keeps a player hooked to a game. Curiosity is the motivation for a player to learn, that in itself is not directly related to a goal, or fantasy-fulfillment [6]. Curiosity can also be used to enhance the fantasy or challenge aspect directly[6]. Yannakakis [10] agrees with the fact that curiosity is part of what makes a game fun. Getting involved and motivated in combination with clear goals and challenges could lead to getting the person aroused and influence the focused attention [11]. Since curiosity

[42]https://www.platinumgames.com/

seems to motivate a player, it should also be a goal for a designer to create a game that makes a player want to explore the game or its mechanics.

Lazzaro’s four components of fun

According to Lazzaro [9], the four components of entertainment are hard fun (fun out of challenge and completing goals with skill), easy fun (fun by exploration and roleplaying), altered states (fun by clearing one's mind or to avoid boredom) and socialization (Fun by interacting or watching others). Yannakakis et al. [10] noticed that three of the four types of fun Lazzaro [9] found have a relation with the intrinsic qualitative factors of Malone [6] stated earlier. Yannakakis [10] also acknowledged another definition of fun, which is composed out of endurability, engagement and expectations. These different definitions allow for a better understanding of why players like a game.

As mentioned earlier, it is not always necessary to have a story or setting to motivate a player. This statement contradicts the third factor of what makes a good game according to Malone, namely fantasy directly. According to Malone [6] there is a difference between intrinsic and extrinsic fantasies. Federhoff [5] suggests that fantasy is not always an important factor to enjoy games. Lazzaro’s [9] Hard Fun is, as stated before a good example of this phenomenon. Easy fun on the other hand relies heavily on fantasy [9]. The importance of a story or a setting is dependent on the kind of fun a designer tries to create.

Game mechanics

Where the story and setting is influenced mostly by the designers, the second level - game mechanics - is at the mercy of the game play. This does not mean that everything is allowed when creating the mechanics because Federhoff [5] identified that “mechanics should feel natural and have correct weight and momentum.” She also states feedback should be given immediately to display user control and get the player involved quickly and easily. Yannakakis et al. [10] state that gameplay also “affect the player’s cognitive process.” Even if the mechanics are complex, it should not influence enjoyment if the game is fun. When a player has fun he or she is willing to learn the more in depth mechanics of a game [5]. Therefore, it is important for a game’s mechanics to be easy to learn and hard to master, but there should also be a balance of skill and challenge in the game [5]. In the end the game mechanics should follow some guidelines before supplementing and being dictated by the game play.

Game interface

The last level to discuss is the game interface. Federhoff [5] make it clear that having a poorly constructed interface can keep a player from enjoying a game, so it is important to actually make it so that the interface does not confuse the player. Nowadays Nielsen Heuristics[12] are a commonly used instrument to find flaws in the design of user interfaces for computer software. Federhoff [5] divided the elements of a game that can be measured through the ten user heuristics. The most important parts that came from these divisions can be found in the table 1. The data found in the table could help define a way to test the game interface and to make sure that the interface can not negatively impact the user enjoyment.

Visibility of system status	➔ Score and/or level design. ➔ Score is a form of positive feedback which can offer encouragement to master games. ➔ Audio feedback can also offer precious feedback.
Match between system and the real world	➔ Games do not have to be reality based, ➔ Analogies to the real world can be useful
User control and freedom	➔ If a user feels restricted, they will most likely become frustrated and this can result in disinterest in the game. ➔ Players not only need to feel in control of the character, but also the manner in how he or she explores the environment. ➔ By giving players the option to reassign their control scheme, can help in creating control. ➔ Give the ability to save games at different states.
Consistency and standards	➔ Use controller standards.
Error prevention	➔ Make sure the player gives a confirmation by confirmation questions

Recognition rather than recall	➔ Make instructions available within the game. ➔ Could be unnecessary by nature of games to learn player how to play early.
Flexibility and efficiency of use	➔ Games should be played by players of different skill levels. Flexibility can be provided by using difficulty levels.
Aesthetic and minimalist design	➔ Game controls and on-screen interface should be simple and non-intrusive to provide easy access to the game environment.
Help users recognize, diagnose and recover from errors	➔ Error messages are not necessary, but are useful in multiplayer. ➔ In contrast to game play issues, this heuristic is relevant to the game's user interface which has the ability to assist the user in the recovery or prevention of errors.
Help and documentation	➔ Game should be understandable without any extra means after the tutorial. Small tips may help

Table 2 How Nielsen Heuristics can help making a game better according to Federhoff [5]

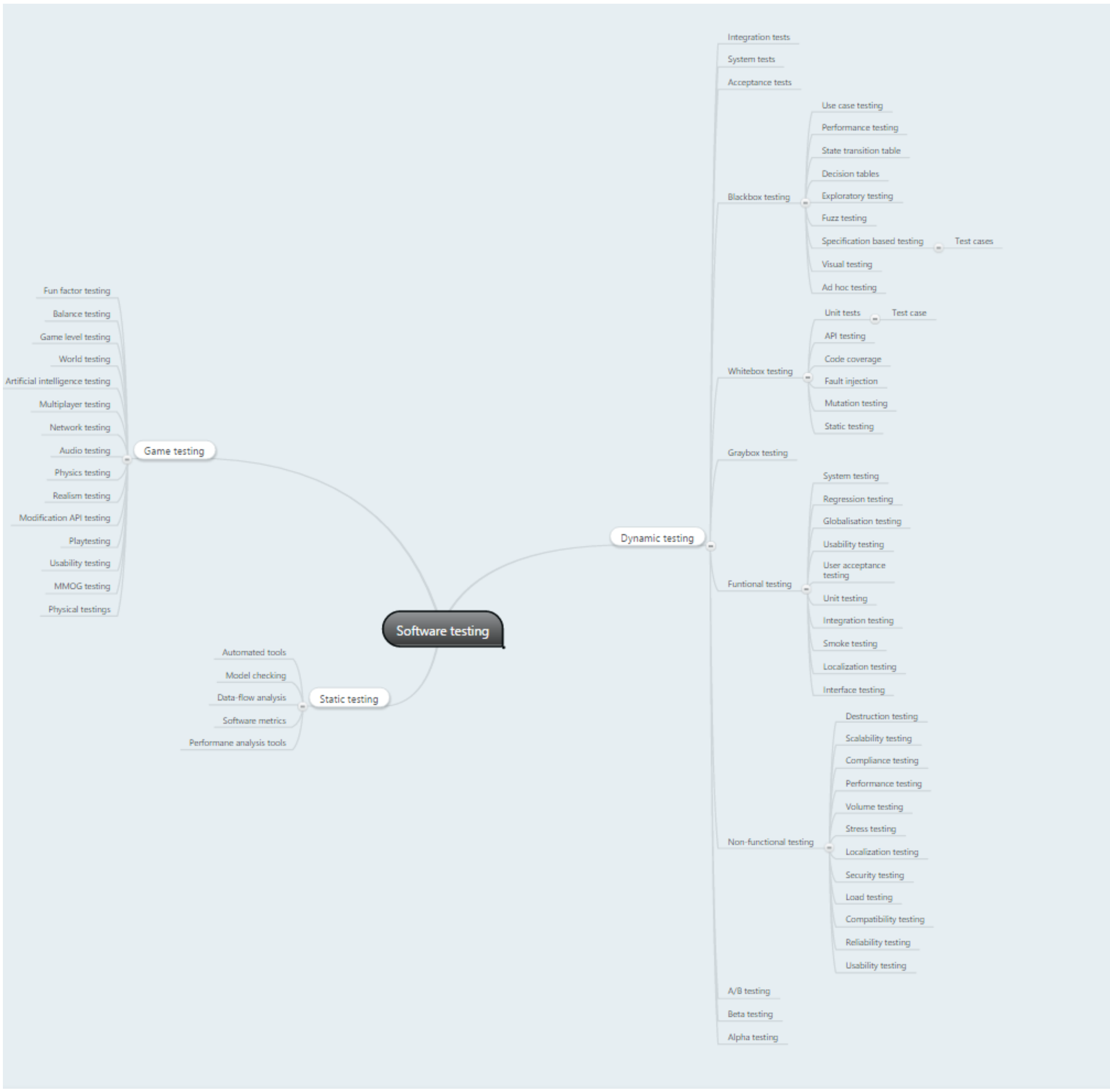
Conclusion

Which guidelines need to be taken into account when designing an online quantitative playtest which focuses on video games? The answer to the question cannot be given yet. During the creation of this literature review it became clear that when a player interacts with a game, the player comes in contact with the game play, interface and mechanics of a game. While all of these factors can influence the enjoyment of a player, if the gameplay is fun it can mask problems in the other components. To make a game good, a game needs to be challenging and make the players curious. Depending on the game, a fantasy element can also help the game in being conceived good, but this is not a necessary need as games can be good without a story or any setting at all. To make a game challenging, setting conditions or use a score and a clear player feedback should be used.

While game play is important, it does not mean that no focus should be given to the game interface or game mechanics. A poor interface can keep a player from enjoying a game. Mechanics should feel natural and have a correct weight and momentum. The feedback the controls and interface give can get a player to be involved quickly and easily. A game should be easy to learn and hard to master. A player should be eased into the game first so a player can have fun, and when the player is having fun, he or she is willing to learn the more in depth mechanics of a games

While these findings on what elements of a game,makes a game fun is a good start to figure out what is needed for a game to be fun, it does not offer any insight in the way to translate their characteristics into a quantitative playtest. The next step to come to an answer for the question mentioned above, is to find out how the elements can be measured in a quantitative way. What the findings of the review do offer, are good data points which the game that will accompany the play test should focus on.

Appendix B - Mind map possible software tests



Appendix C - Game idea platform game

80’s SciFi (fantasy)
Card based action combat. (Fast paced, customizable game play)
Cards behave like guns are normal weapons seen in other games (similar concept but different presentation)
Realistic/heavy movement vs precise platforming
Gridtile based graphics. (Easiest method to implement datahooks)
Characters can be larger but I prefer a hitbox that is a multiple of gridsize. (Preference)
Card attacks based on the attack monsters do. I.e. monster spits fireball -> monster card also includes a fireball. (Make a consistent world)

Player replay/media player like system. (For seeing how players interact with the game)

Character:

- Scifi: jaren 80 futuristiche
- Iets op de arm om units in te steken(Dia’s Cards, floppy discs of wathever
- Geen metalen suits want clunky sound design
- Voor de rest ben je vrij om zelf iets te ontwerpen

Sound:

- Bladerunner-esque
  - Layers!
    - Layers gebaseerd op het scherm zichtbaar is

Goal:

For this module the slice is to create at least one level style (for example one spritesheet for all the city based levels) and have a working tech demo with most if not all of the gameplay features incorporated.

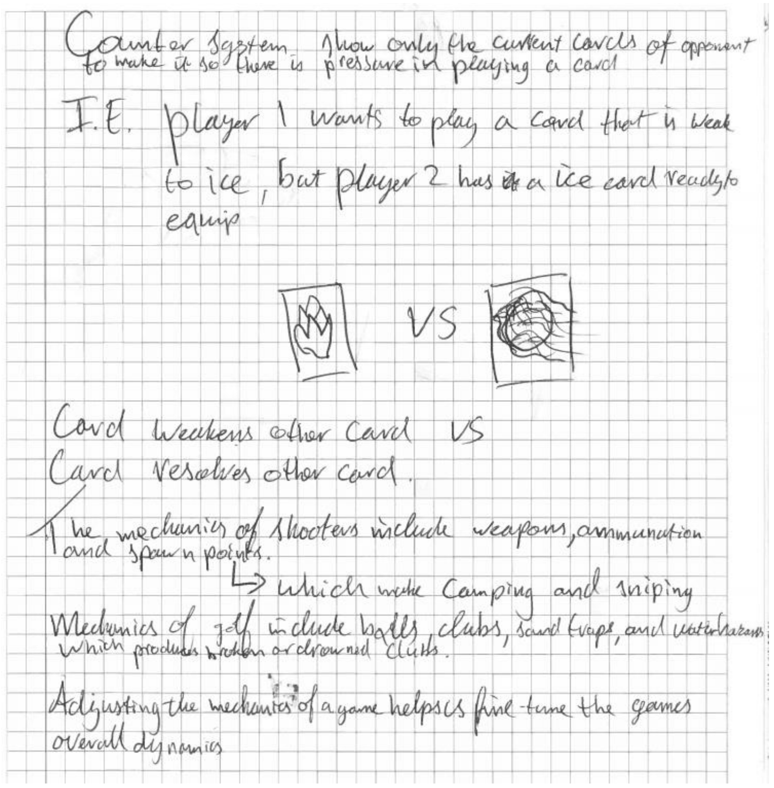
Aesthetics

- 2D
    - Tilebased (gridbased) for platforms and interactables
        - Other layers do not have to be placed on a grid.
    - Fluid
        -
    - Animation
        - Stiff vs fluid
        - Will have effect on the gameplay
    - Parallax scrolling
    - Multiple layers to work with
    - Lighting can be achieved through shaders
        - Lamp resource
- Consistency
    - Everything in the game come from something that already exists
        - I.E. Player attacks are copied from monsters
    - Buildings are set pieces do not fall out of place
    - Story to justify everything
- 80's SciFi
    - Setting ideas
    - City based
        - bladerunner esque.
        - Downtown New York
    - Style to be determined. A brainstorm session between us would help
- Sounds
    -

Mechanics

- Platform action game
    - Movement

        - very smooth and Super Meat Boy like
        - Realistic and very heavy movement (with a lot of weight)
- Cards
    - ;
    - An equiped card can be used a predefined amount of times
        - When a card is used up, the card will be unequipped.
            - This card will be restored based on...
                - a long timer?
                - After a fight?
                - Interaction with specific objects?
                - Items?
    - Stamina/Mana/MP meter to prevent spamming?
        - Only used when people button mash?



Counter System show only the current cards of opponent to make it so there is pressure in playing a card

I.E. player 1 wants to play a card that is weak to ice, but player 2 has ~~it~~ a ice card ready to equip

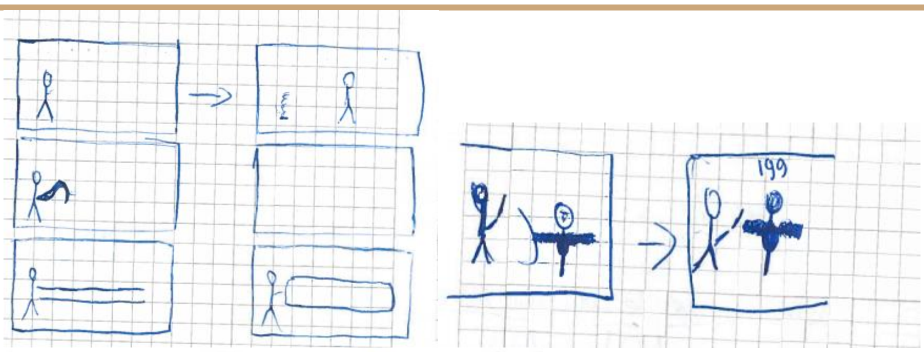
VS

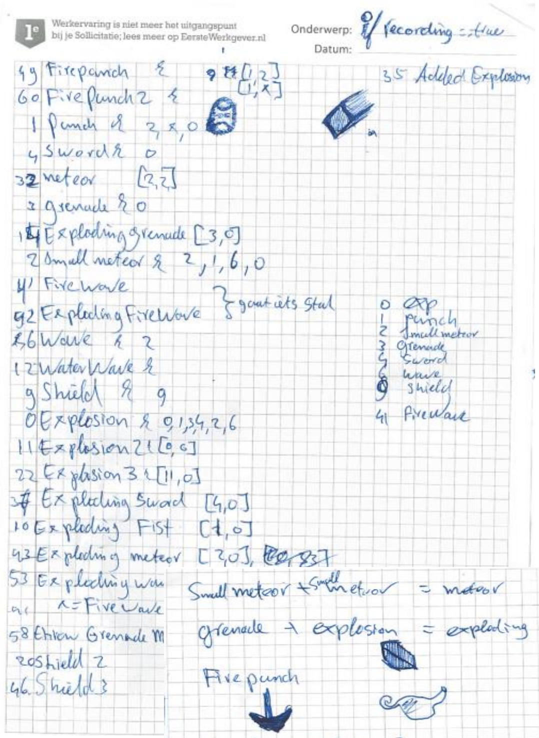
Card weakens other card VS
Card resolves other card.

The mechanics of shooters include weapons, ammunation and spawn points.
↳ which make camping and sniping

Mechanics of golf include balls, clubs, sand traps, and waterhazards which produces broken or drowned clubs.

Adjusting the mechanics of a game helpscs fine-tune the games overall dynamics





Werkervaring is niet meer het uitgangspunt bij je Sollicitatie; lees meer op EersteWerkgever.nl

Onderwerp: ~~0~~/ recording =true

Datum:

49 Firepunch & 9 ~~17~~ [1,2] [1,x]

35 Added Explosion

60 FirePunch2 &

1 Punch & 2,x,0

4 Sword& 0

32 meteor [2,2]

2 grenade & 0

1~~4~~ Exploding grenade [3,0]

2 Small meteor & 2,1,6,0

41 Firewave

92 Exploding FireWave } gaat iets stuk

26 Wave & 2

12 WaterWave &

9 Shield & 9

0 Explosion & 0,1,3,4,2,6

11 Explosion2 & [0,0]

22 Explosion3 & [11,0]

3~~7~~ Exploding Sword [4,0]

10 Exploding Fist [1,0]

43 Exploding meteor [2,0], ~~[0,83]~~

53 Exploding wa

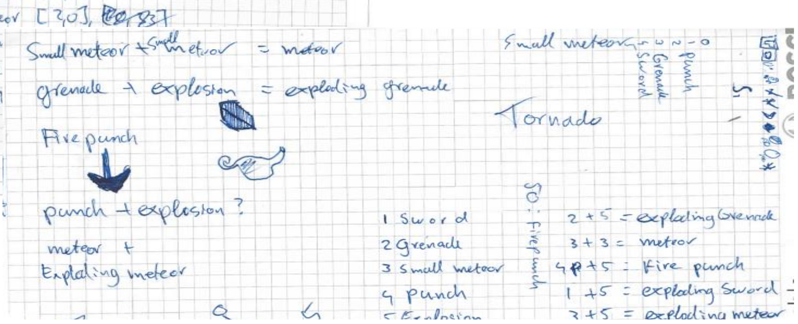
x=FireWave

58 throw Grenade M

20 Shield 2

46. Shield 3

0 exp
1 punch
2 Small meteor
3 Grenade
4 Sword
6 wave
9 shield
41 FireWave



Small meteor + Small meteor = meteor

grenade + explosion = exploding grenade

Fire punch

punch + explosion ?

meteor +

Exploding meteor

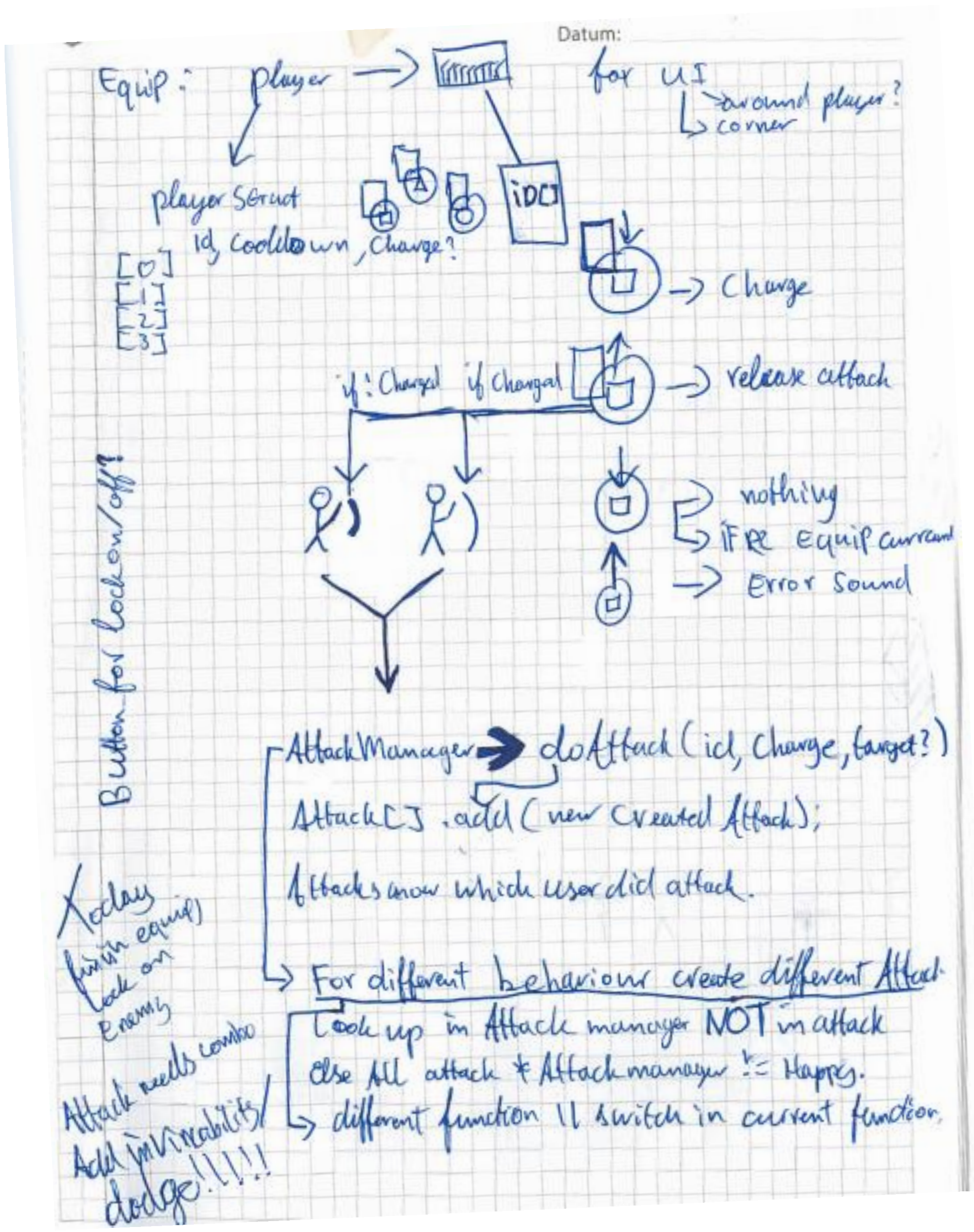
Small meteor 3 Sword 2 Grenade 1 Punch 0

Tornado

50: Firepunch

1 Sword
2 Grenade
3 Small meteor
4 punch
5 Explosion

2 + 5 = exploding Grenade
3 + 3 = meteor
4 ~~p~~ + 5 = Fire punch
1 + 5 = exploding Sword
3 + 5 = exploding meteor



AttackManager → doAttack(id, Charge, target?)

Attack[] .add(new Created Attack);

Attacks know which user did attack.

→ For different behaviour create different Attack

Look up in Attack manager NOT in attack

else ALL attack + Attack manager != Happy.

→ different function || switch in current function.

~~Today~~ finish equip) lock on Enemy

Attack needs combo

Add invincibility/ dodge!!!!!

Appendix D - Game idea Rogue-like rpg

- Game design idea
    - Rogue-like rpg
    - Random generated dungeons
        - With at least one:
            - Upgrade room
            - Shop
            - Save Crystal
            - Boss
    - From city -> grasslands -> caves -> dungeons -> final castle
    - Random generated quests
        - Generic rpg fetch quests
            - Rewards party member, XP, items and/or equipment
            - Not needed to go to next level
    - Story
        - Random generated for every new area visited
            - Story inspired by existing tropes[43]
    - Combat
        - Turnbased
            - Active time battle
                - Meter that will fill up before you can do your next action
                - Time meter needs to fill up is based on player stats
            - Order based
                - Order will be based on a player attribute resembling speed
                - Characters can act multiple times before other character can act
                    - I.e. Attack order P1,P1,P1,P1,Boss,P2,P1,P3,P1,Boss
        - RealTime
            - Control one player
                - Character will be able to handle different weapons
            - Control multiple players

[43]http://tvtropes.org/pmwiki/pmwiki.php/Website/TheRPGClichesGame

Every character will only be able to use a certain set of weapons or abilities

Equip attacks

- Character can do certain attacks if they are equiped
  - Simple controls but condition on which attack will be done
    - One or two attack buttons
      - Which attack done is based on the target and location of the player.
      - E.g if distance between the player and target is 10 meters button 1 does a slash. If distance between player and target is 25 meters button 1 does a fireball attack.

Appendix E - Game idea Endless RPG

Endless rpg

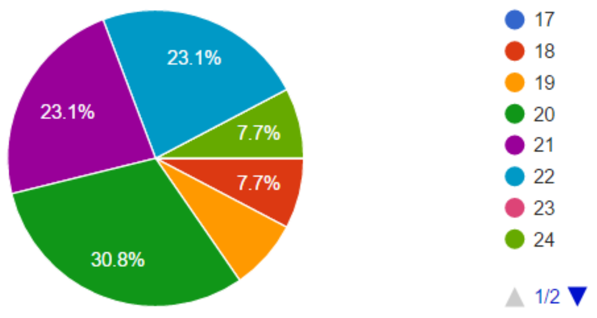
Classic turn based role playing battle game where a player must try to defeat as much waves of enemies as possible. The enemies will get stronger depending on how much waves have been defeated. Every monster defeated will give the player experience points, with which you can make the character stronger. Like an ordinary RPG, monsters will also drop loot and money. The loot can be equipment, or items that can make equipment more powerful. Once the player loses to a wave of enemies, the player would lose all progress on that character (perma death). Based on how well the player acted, the player will be rewarded with a special kind of points which he or she can invest in special spells or abilities so a sense of progress is still achieved.

Appendix F - Moodboard city and atmosphere

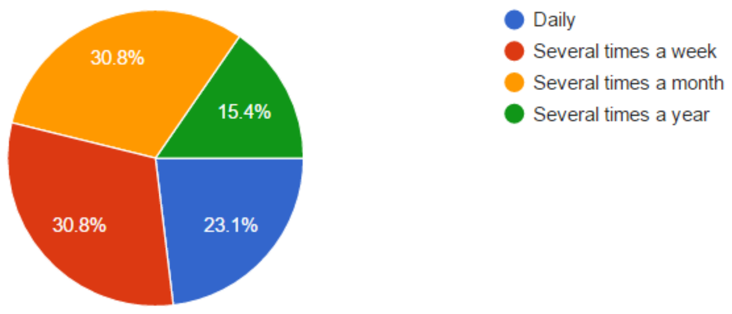
Appendix G - Moodboard outfit and objects

Appendix H - Questionnaire and results

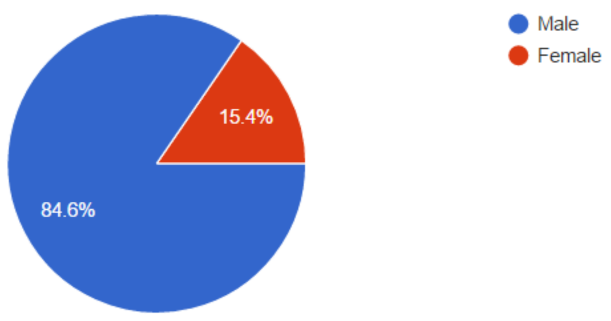
What is your age? (13 responses)



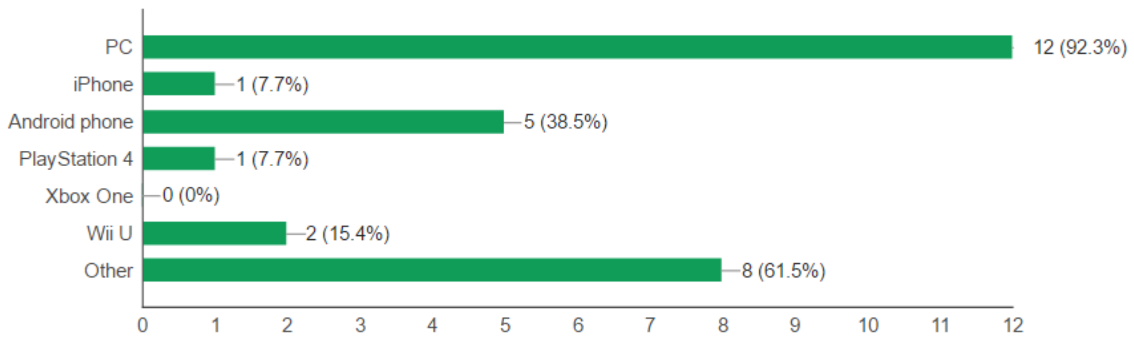
How often do you play video games? (13 responses)



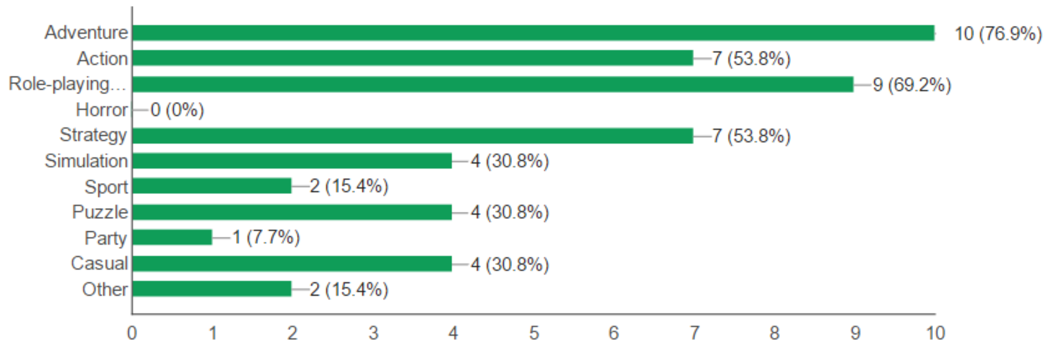
What is your gender? (13 responses)



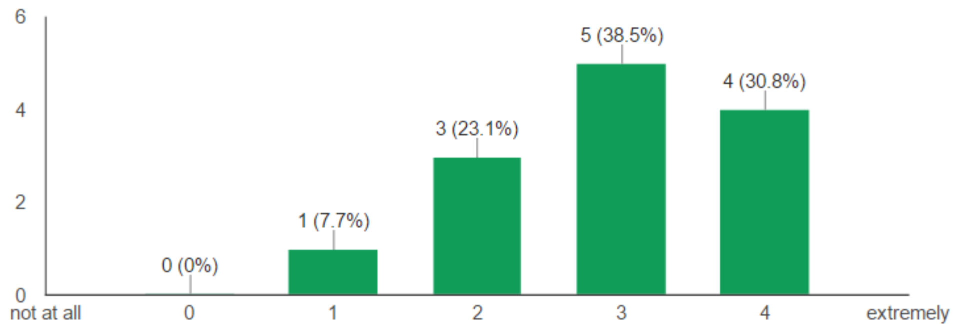
On which devices do you play video games (13 responses)



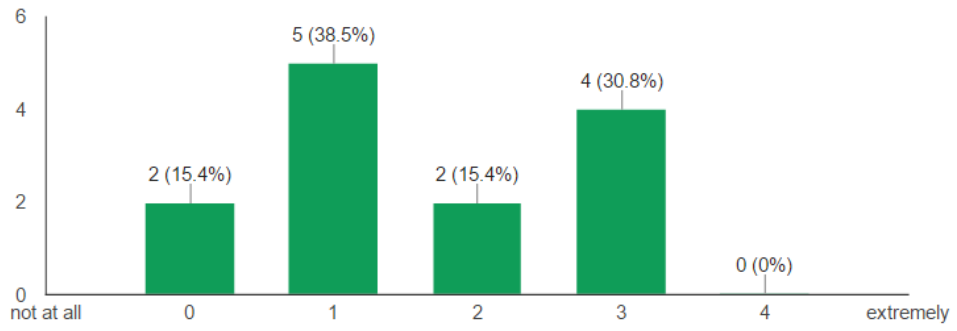
What genre of games do you normally play (13 responses)



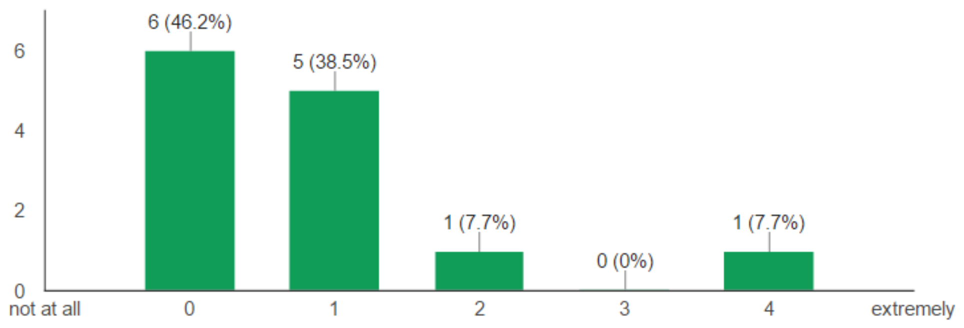
I felt content (13 responses)



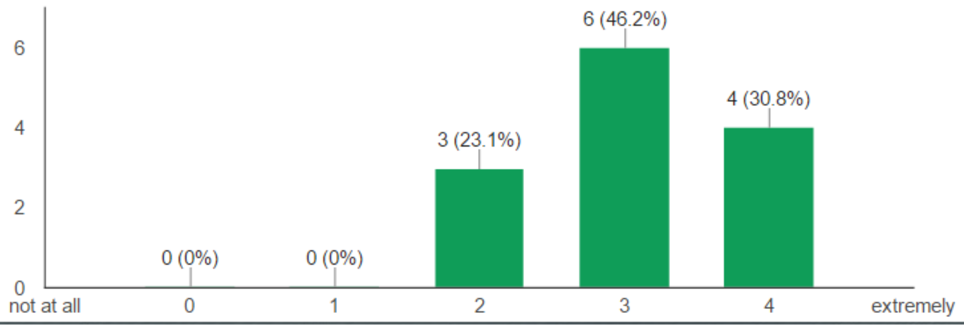
I felt skilful (13 responses)



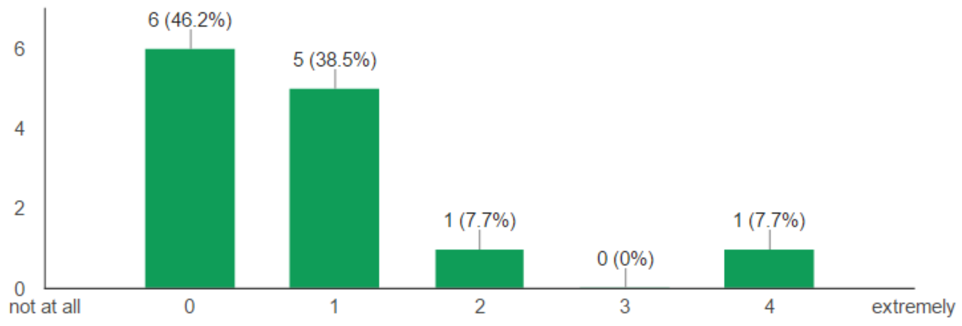
I was interested in the game's story (13 responses)



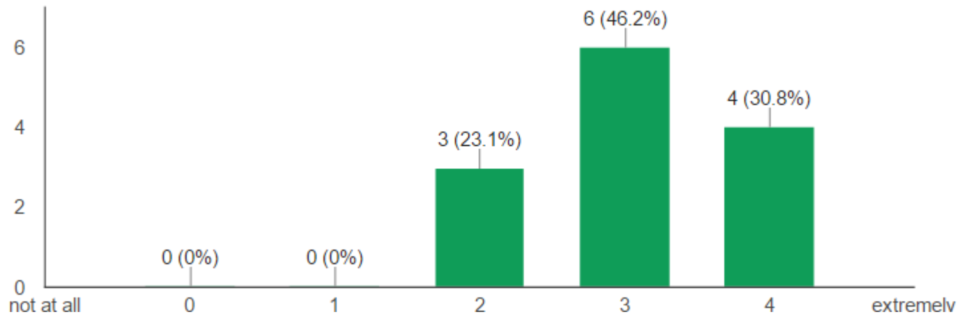
I thought it was fun (13 responses)



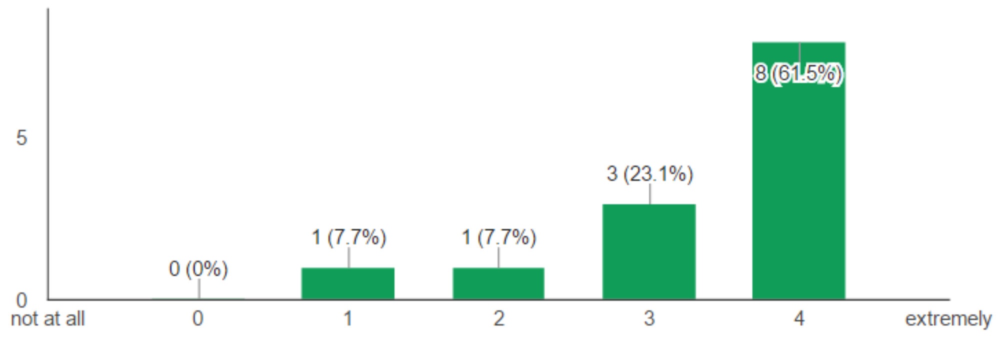
I was interested in the game's story (13 responses)



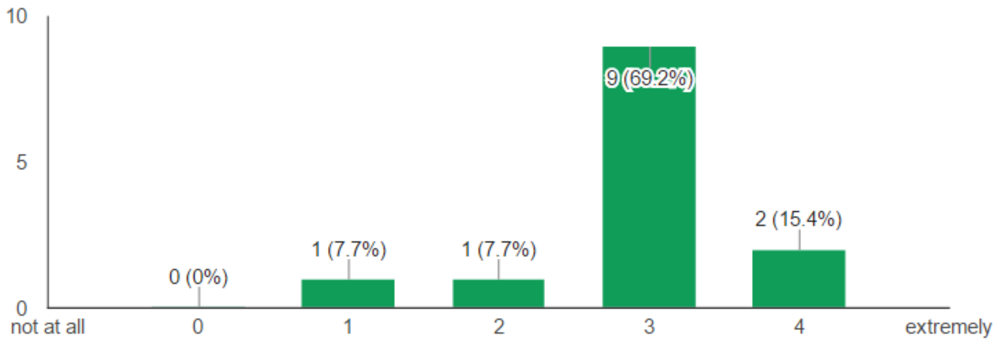
I thought it was fun (13 responses)



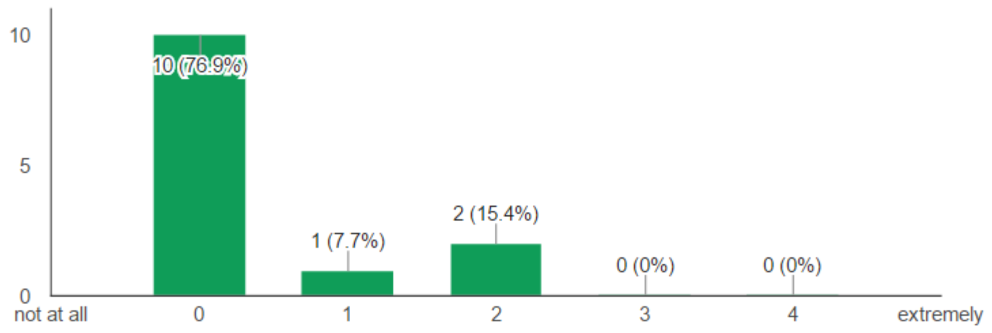
I was fully occupied with the game (13 responses)



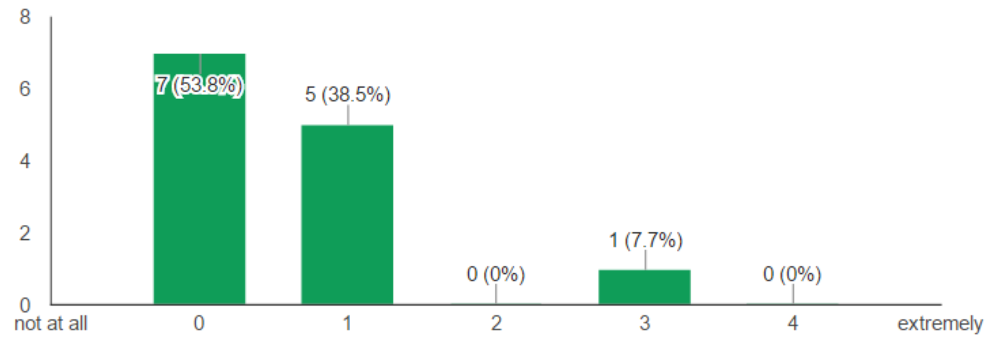
I felt happy (13 responses)



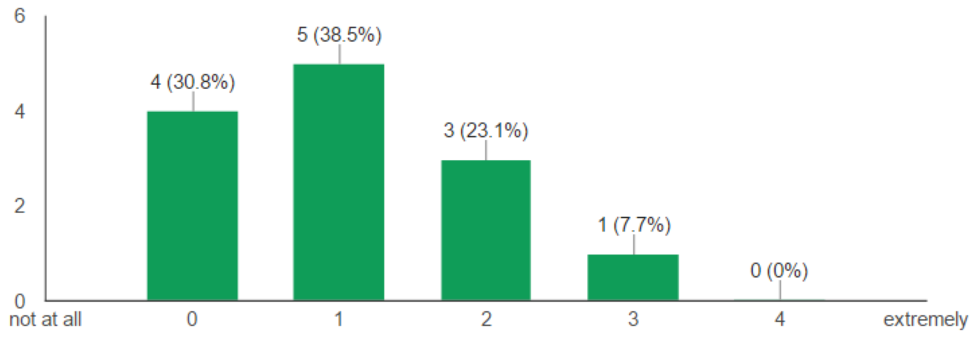
It gave me a bad mood (13 responses)



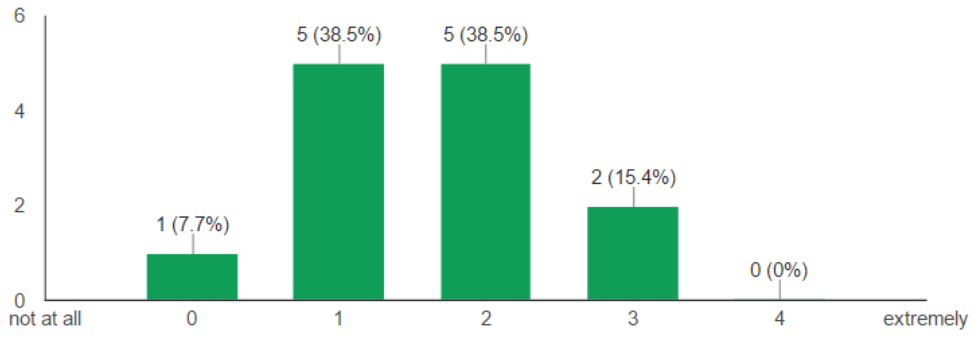
I thought about other things (13 responses)



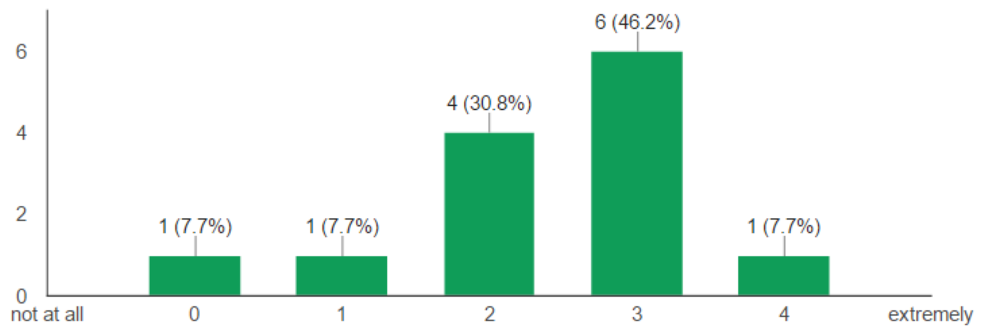
I found it tiresome (13 responses)



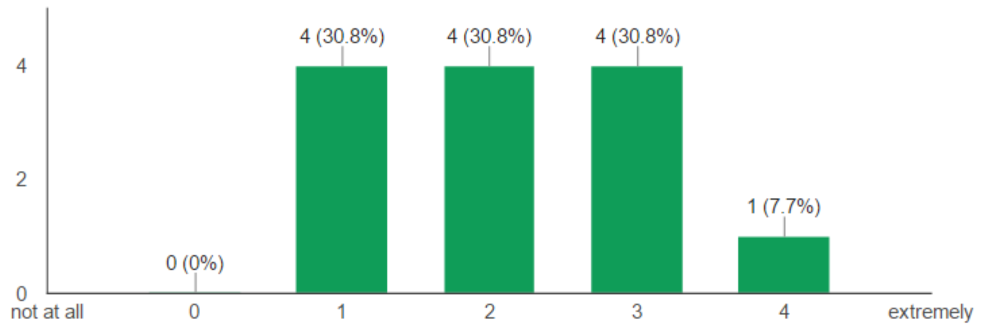
I felt competent (13 responses)



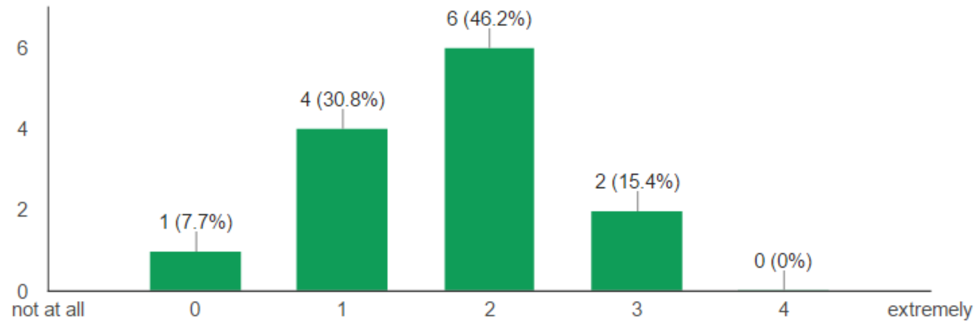
I thought it was hard (13 responses)



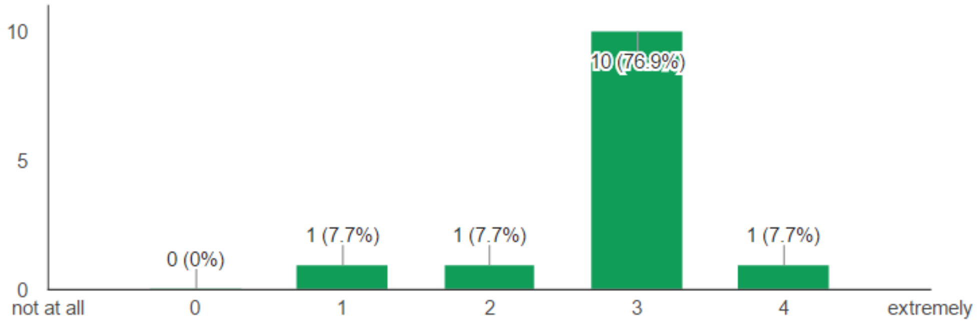
It was aesthetically pleasing (13 responses)



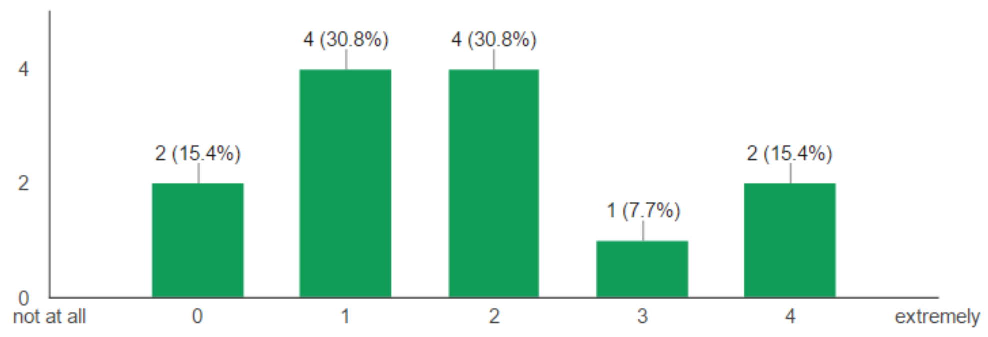
I forgot everything around me (13 responses)



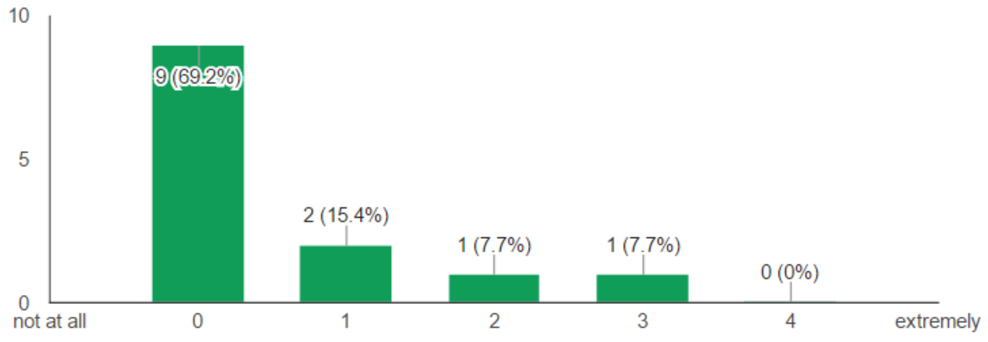
I felt good (13 responses)



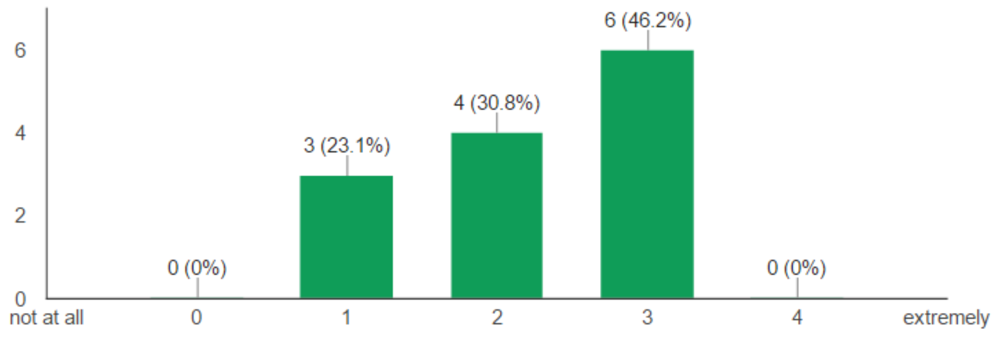
I was good at it (13 responses)



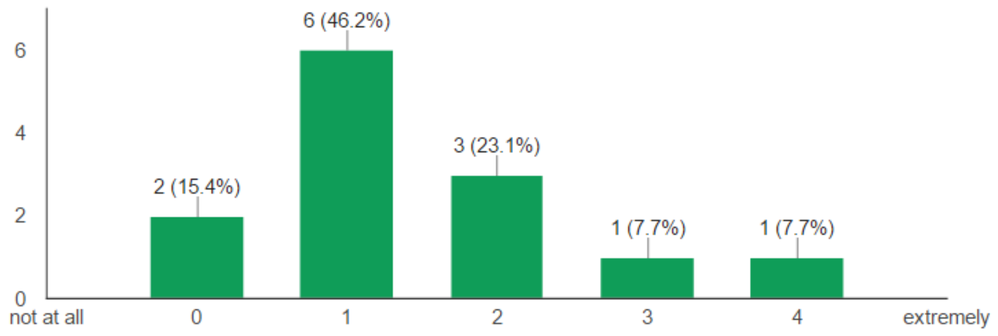
I felt bored (13 responses)



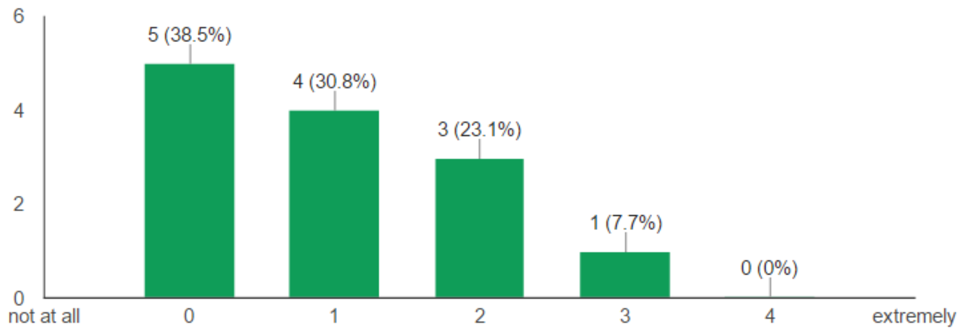
I felt successful (13 responses)



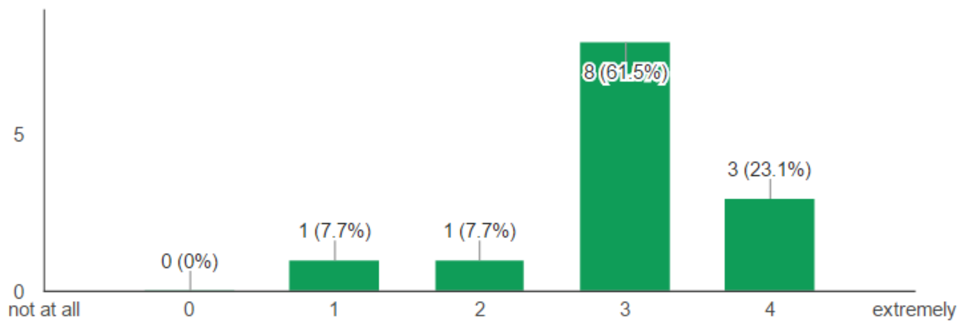
I felt imaginative (13 responses)



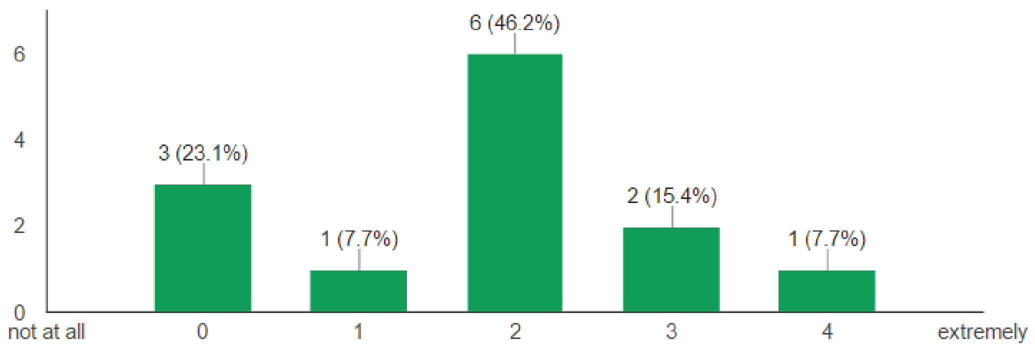
I felt that I could explore things (13 responses)



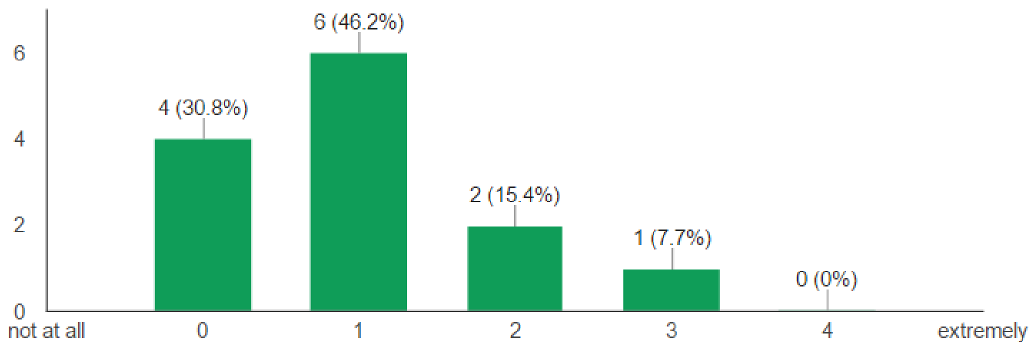
I enjoyed it (13 responses)



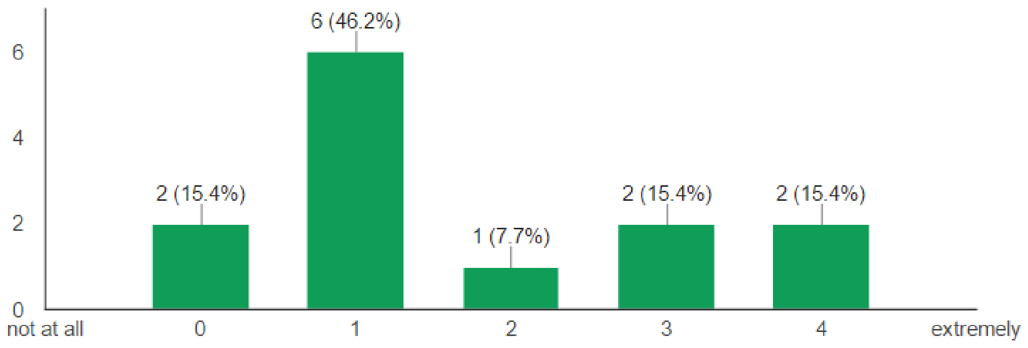
I was fast at reaching the game's targets (13 responses)



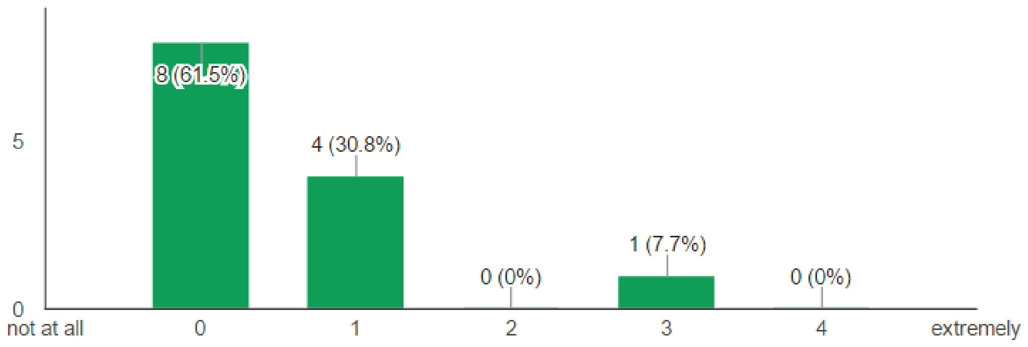
I felt annoyed (13 responses)



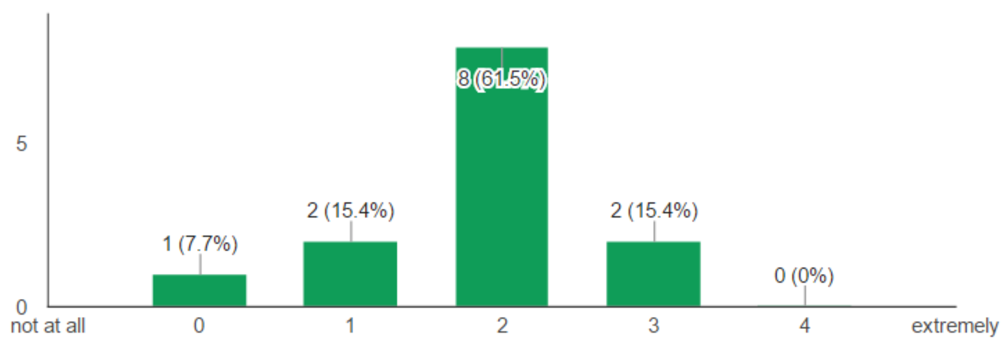
I felt pressured (13 responses)



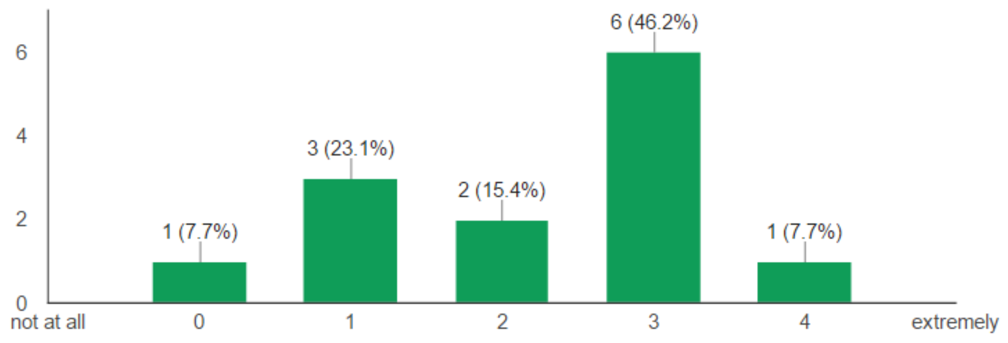
I felt irritable (13 responses)



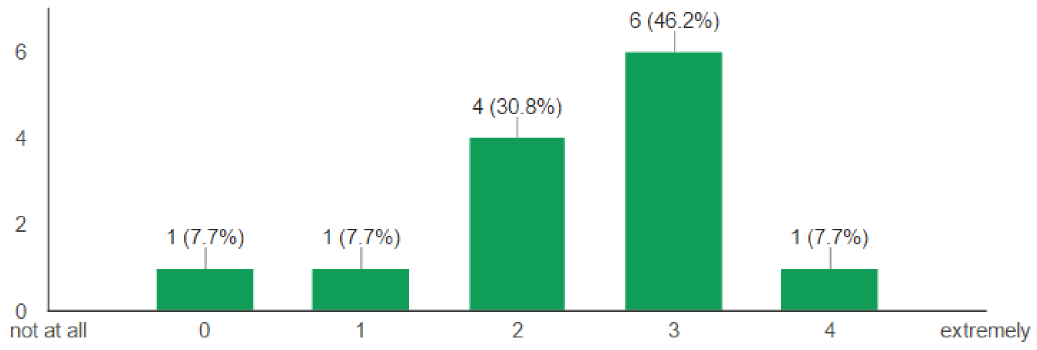
I lost track of time (13 responses)



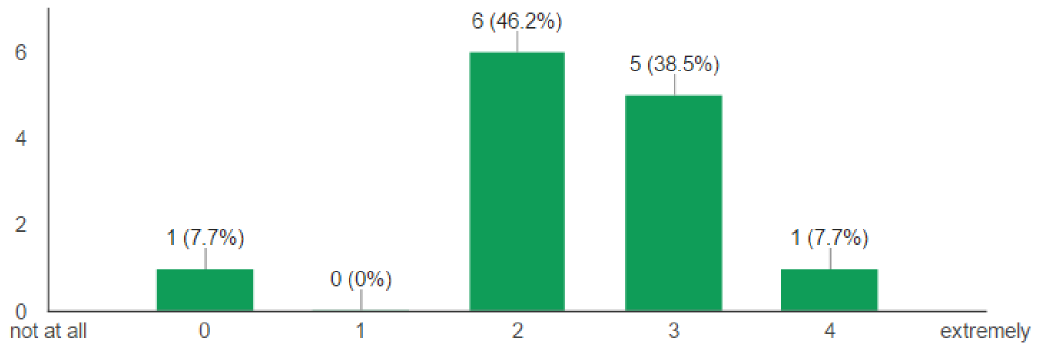
I felt challenged (13 responses)



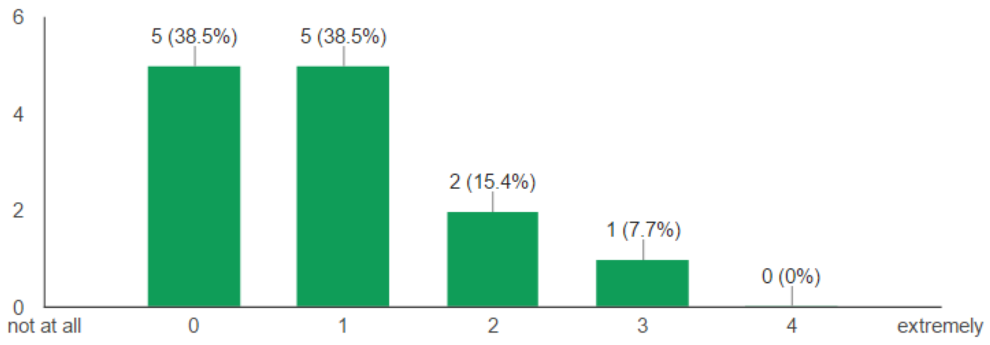
I found it impressive (13 responses)



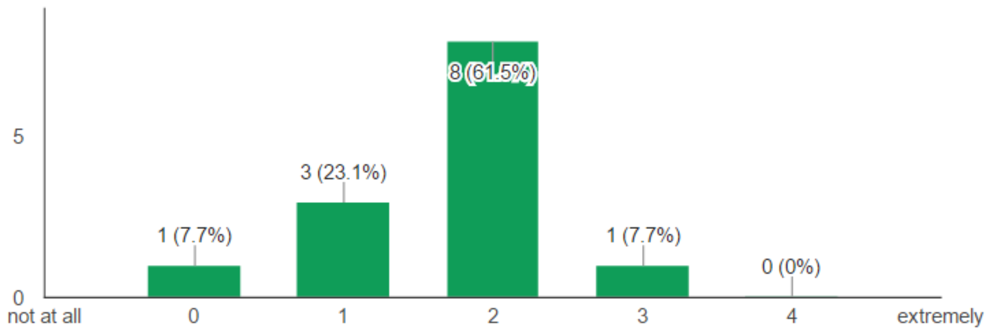
I was deeply concentrated in the game (13 responses)



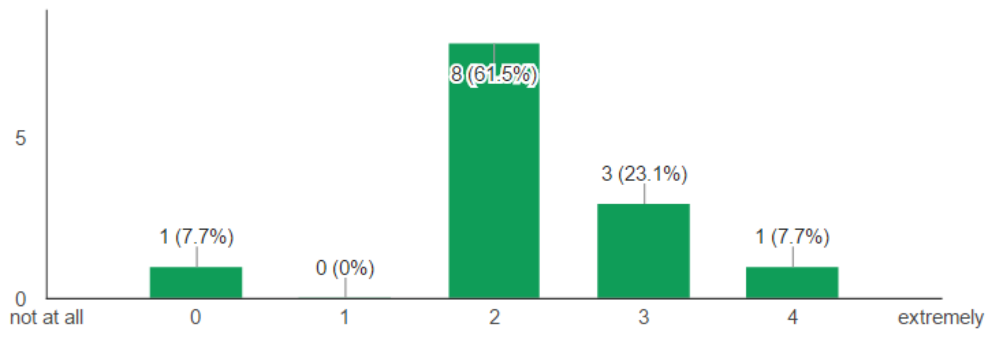
I felt frustrated (13 responses)



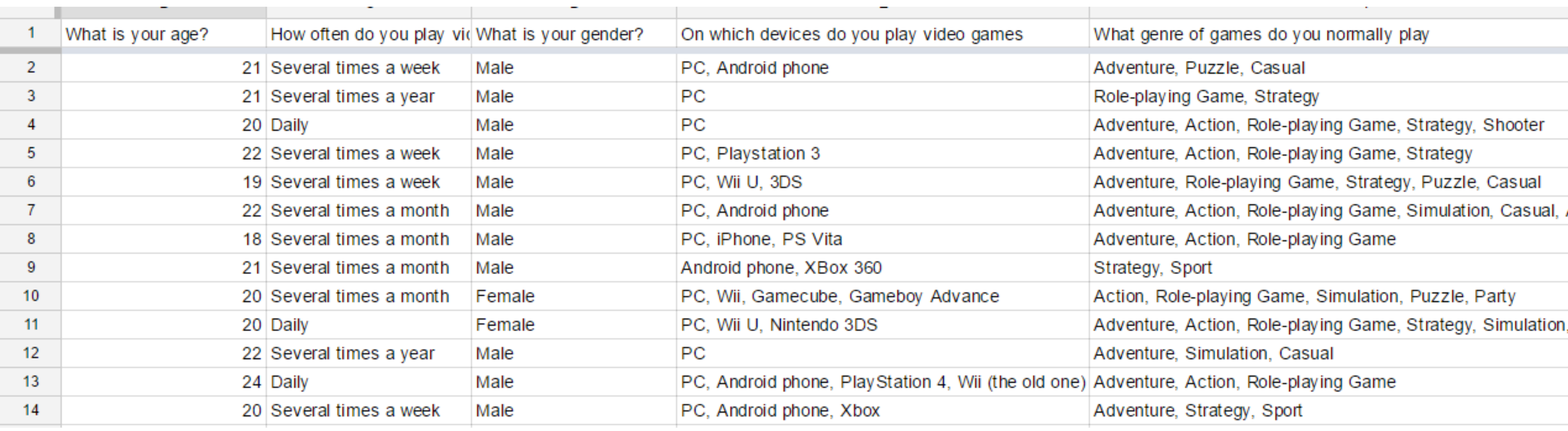
It felt like a rich experience (13 responses)



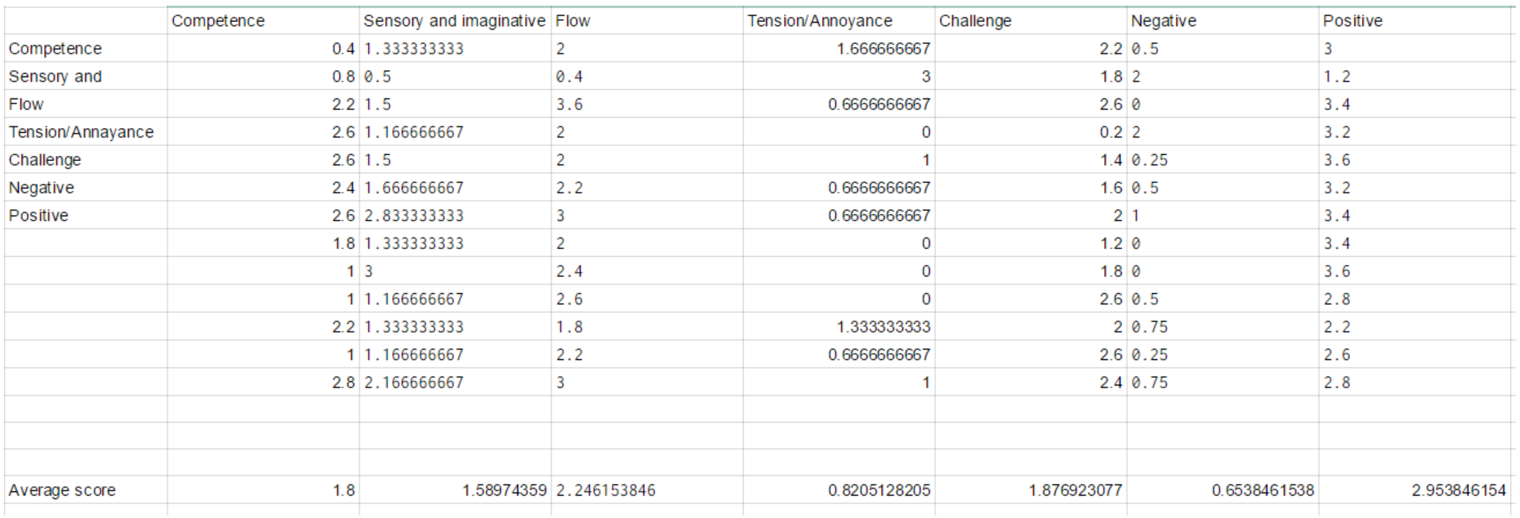
I had to put a lot of effort into it (13 responses)



Appendix I - Average results and other data from survey



1	What is your age?	How often do you play vi	What is your gender?	On which devices do you play video games	What genre of games do you normally play
2	21	Several times a week	Male	PC, Android phone	Adventure, Puzzle, Casual
3	21	Several times a year	Male	PC	Role-playing Game, Strategy
4	20	Daily	Male	PC	Adventure, Action, Role-playing Game, Strategy, Shooter
5	22	Several times a week	Male	PC, Playstation 3	Adventure, Action, Role-playing Game, Strategy
6	19	Several times a week	Male	PC, Wii U, 3DS	Adventure, Role-playing Game, Strategy, Puzzle, Casual
7	22	Several times a month	Male	PC, Android phone	Adventure, Action, Role-playing Game, Simulation, Casual,
8	18	Several times a month	Male	PC, iPhone, PS Vita	Adventure, Action, Role-playing Game
9	21	Several times a month	Male	Android phone, XBox 360	Strategy, Sport
10	20	Several times a month	Female	PC, Wii, Gamecube, Gameboy Advance	Action, Role-playing Game, Simulation, Puzzle, Party
11	20	Daily	Female	PC, Wii U, Nintendo 3DS	Adventure, Action, Role-playing Game, Strategy, Simulation,
12	22	Several times a year	Male	PC	Adventure, Simulation, Casual
13	24	Daily	Male	PC, Android phone, PlayStation 4, Wii (the old one)	Adventure, Action, Role-playing Game
14	20	Several times a week	Male	PC, Android phone, Xbox	Adventure, Strategy, Sport



	Competence	Sensory and imaginative	Flow	Tension/Annoyance	Challenge	Negative	Positive
Competence	0.4	1.333333333	2	1.666666667	2.2	0.5	3
Sensory and	0.8	0.5	0.4	3	1.8	2	1.2
Flow	2.2	1.5	3.6	0.6666666667	2.6	0	3.4
Tension/Annayance	2.6	1.166666667	2	0	0.2	2	3.2
Challenge	2.6	1.5	2	1	1.4	0.25	3.6
Negative	2.4	1.666666667	2.2	0.6666666667	1.6	0.5	3.2
Positive	2.6	2.833333333	3	0.6666666667	2	1	3.4
	1.8	1.333333333	2	0	1.2	0	3.4
	1	3	2.4	0	1.8	0	3.6
	1	1.166666667	2.6	0	2.6	0.5	2.8
	2.2	1.333333333	1.8	1.333333333	2	0.75	2.2
	1	1.166666667	2.2	0.6666666667	2.6	0.25	2.6
	2.8	2.166666667	3	1	2.4	0.75	2.8
Average score	1.8	1.58974359	2.246153846	0.8205128205	1.876923077	0.6538461538	2.953846154

Appendix J - Notes made during interview

Person 1:

This person thought the GUI was unclear and did not understand the controls that well. The person did like the fact that a lot of different attacks were accessible. The game reminded this testperson of a well know brawler called Super Smash Bros. in which the goal is also to defeat the other players in a similar setting. The tester liked the concept and would like to try it again in the future if the is made more clear and accessible.

Person 2:

This tester said that the game did not contain any challenge. This person also stated that the game and the interface was unclear and that this should be fixed by inserting a story which explains the setting and gives a brief description of how to play the game. The story could also help with the sense of immersion.

Person 3:

This participant took the time to learn the controls in the ad hoc phase so it could understand how the game worked because the interface lacked the requirements to do so. While figuring out how the game works, the tester found out that certain attacks could be used in a combination. The tester liked this concept and took extra time to find more combinations and to figure out which attacks would do the most damage.

Person 4:

This person liked the game because it was a short experience and could offer a lot of depth. The player did not know if they would play the game in longer burst without the fear of the game becoming boring. The tester also said the game was unclear and the GUI could use some help. The suggestions the tester made for the GUI were making the lifebar bigger, show the amount of lives left, improve the graphical appearance of the GUI in general, add a tutorial, let all the icons match with the attacks, and limit the amount of attacks that could be done because the players were given too much choice.

This person came up with a new gameplay idea which contained a limitation to 3 attacks and make the attacks evolve during the match by using the cards in the deck as extra attributes for the three main attacks.

Person 5:

This tester liked the game in general but also thought the game and presentation was a bit unclear. A possible solution for this problem would be to use the same terms in the game that is also used in the shooter genre.

Person 6:

This person thought the game was ok and could act as fun basis for a game that has more meat to it. It also mentioned the problems other persons pointed out regarding the user interface.

Person 7:

This person also mentioned that the controls are hard to master. This tester liked the weapons and mentioned that more weapons is more havoc equals more fun. The tester would have like more distinct attacks and also stated a lot of attacks could be nerfed (balanced). The player likes the concept and would like to play a version later on. The player also mentioned the arena should be fixed by adding more platforms.

Idee: limit access to deck and add timer

Person 8:

The grenade attacks were hard to control and to hit persons with so the tester would like to see a fix for this. “More graphics for the attacks would not do the game any harm”; a statement made by the tester. The player liked the concept but the tester thought it needed more work. Also the GUI was a point that was bugging this person.

Person 9:

This person did not like the UI and thought the game was to hard to control

Person 10:

This person also liked the depth the attack system had, but also mentioned it is not the kind of game the tester normally would play so the tester felt like the suggestions she would make would do more harm than good.

Person 11:

This person wants to see the jump button remaped to the spacebar. The tester lost but still felt successful. The tester mentioned the controls are hard and should be more in line with shooters or controls used by pc games in general. An example could be to equip cards with 1,2,3,4 and use the cards by pressing one of the regular attack buttons.

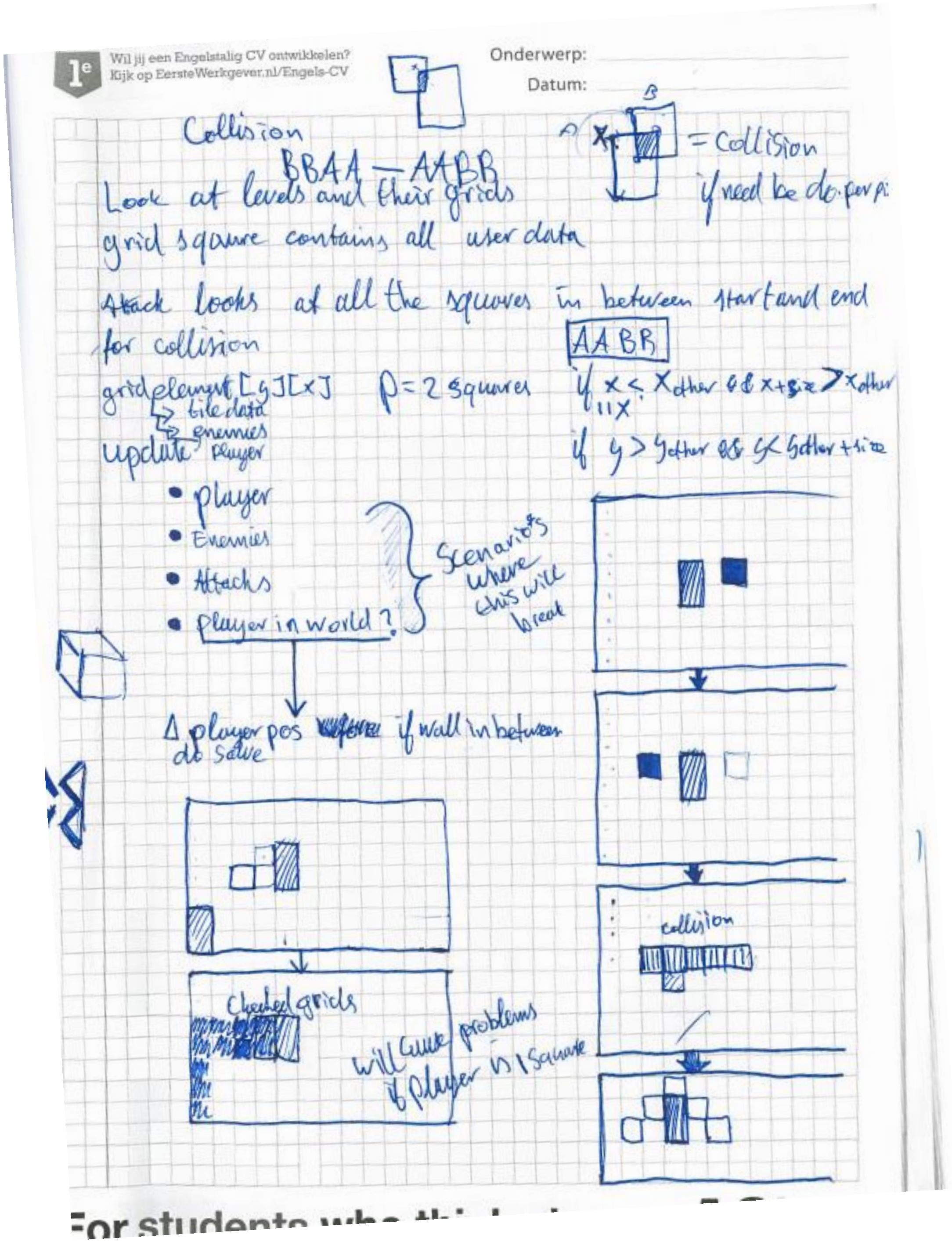
Person 12:

This tester mentioned it missed depth because the urge to buttonmash was big. The game would be better if the attacks had more impact and the player could not move freely as quick as they do now after doing an attack.

Person 13:

Like the combo system and was pleasantly surprised some attacks could spawn other attacks. The tester complimented the possibilities the attacks included, and like the way some of the attacks would match. On the other side this tester mentioned some combinations did not make sense and that the GUI is still unclear. Would like to play the game in a stage when more of the game is finished.

Appendix K - Notes not processed in the thesis



Wil jij een Engelstalig CV ontwikkelen?
Kijk op EersteWerkgever.nl/Engels-CV

Onderwerp:

Datum:

Collision

BBAA — AABB

= Collision

if need be do. per p:

Look at levels and their grids

grid square contains all user data

Attack looks at all the squares in between start and end for collision

$\boxed{\text{AABR}}$

gridelement [y][x]

↳ tile data

↳ enemies

p = 2 squares

if x < Xother && x+size > Xother

|| X

if y > Yother && y < Yother + size

update player

- player
- Enemies
- Attacks
- player in world?

} Scenario's where this will break

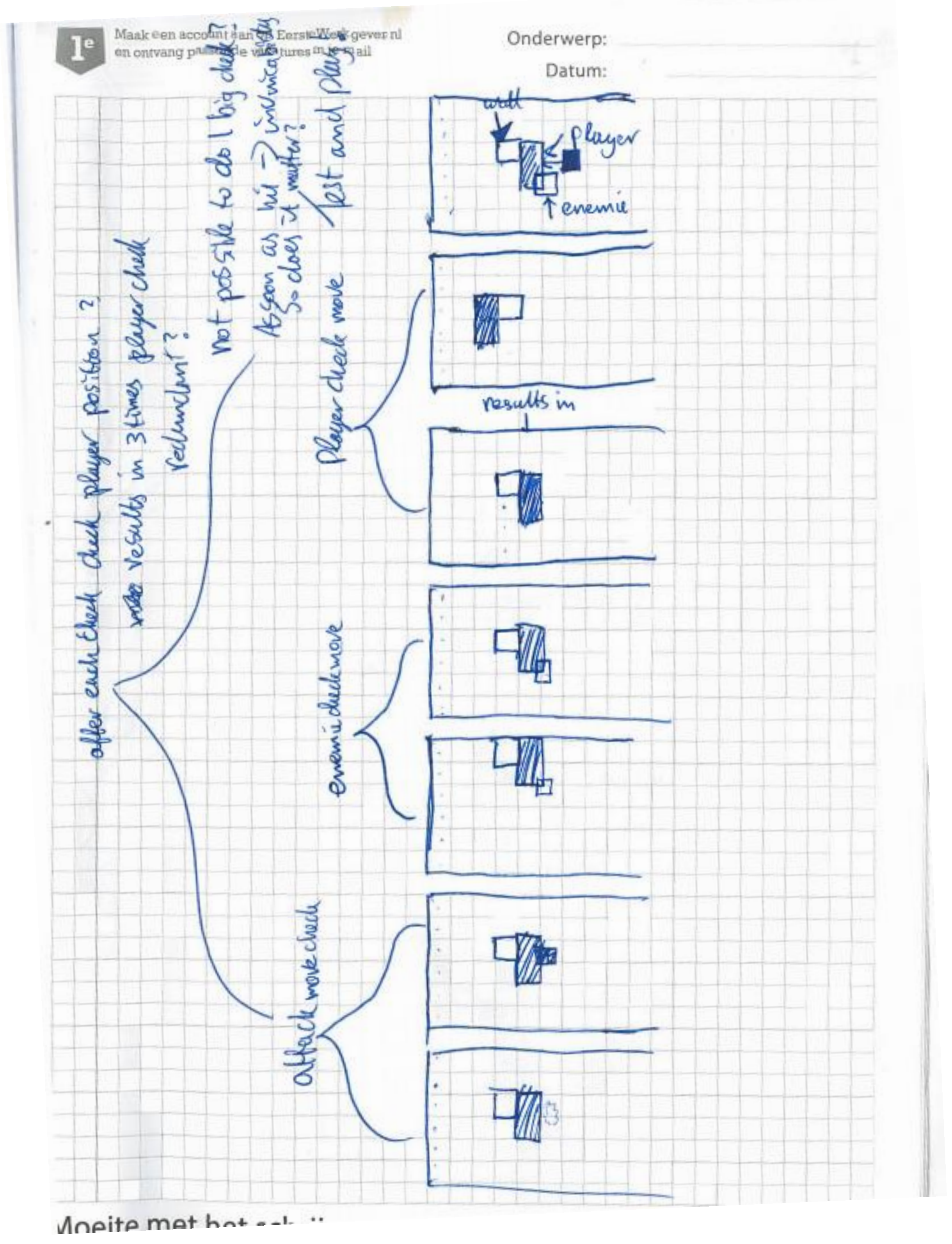
Δ player pos ~~before~~ if wall in between do save

collision

Checked grids

will cause problems if player is 1 square

For students who thi


Maak een account aan op EersteWerkgever.nl
en ontvang passende vacatures in je mail
Onderwerp:
Datum:
after each check check player position ?
~~vake~~ results in 3 times player check
redundant ?
not possible to do 1 big check?
As soon as hit -> invincability
So does it matter?
test and play
attack move check
enemie check move
Player check move
results in
wall
player
enemie
Moeite met het

Datum:

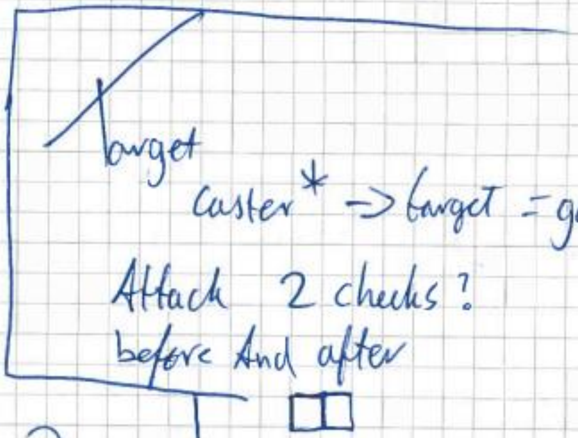
Combo werkt door animatie te veranderen

→ andere damage in combostring?

→ Combo als apparte behaviour?

raycast?

Gridelement



caster* → target = gameObject

Attack 2 checks?
before And after

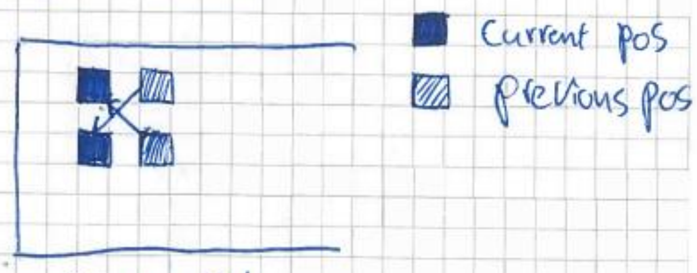
Attack:

```
update(){
    do animation
    add Velocity
    checkCollision()}
```

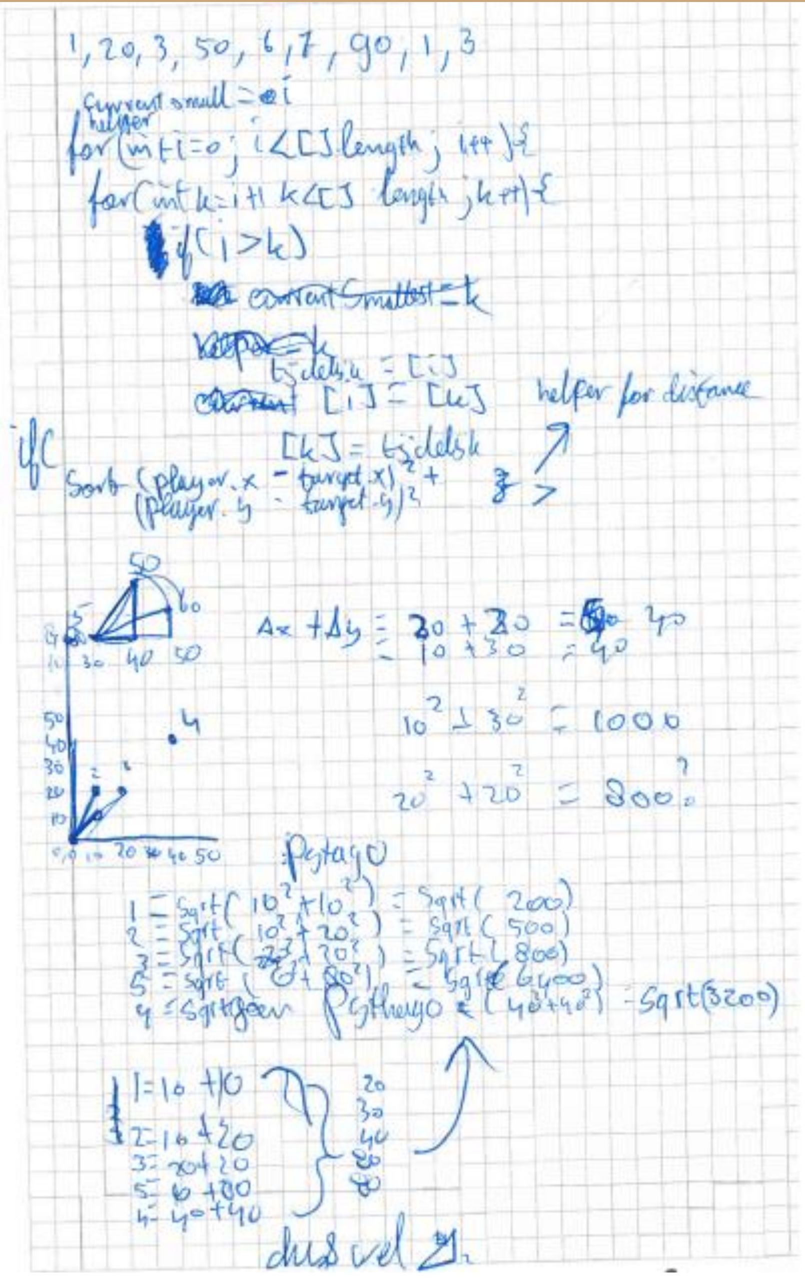
keep previous and new position to check collision

→ attack manager needs to know level →
*current level

Attackmanager:
Check Collision
update grid?



Would collide



1, 20, 3, 50, 6, 7, 90, 1, 3

```
current small = i
helper
for(int i=0; i<[].length; i++){
  for(int k=i+1 k<[] length; k++){
    if(i>k)
      ~~current Smallest = k~~
      ~~[illegible] = k~~
      tijdelijk = [i]
      ~~[illegible]~~ [i] = [k]
      [k] = tijdelijk
```

helper for distance

if(
Sqrt (player.x - target.x)^2 +
(player.y - target.y)^2
}
>

50
60
50
30
40
50
.4
50
40
30
20
10
0,0
10
20
40
50

$\Delta x + \Delta y$ = 30 + 20 = 50 40
= 10 + 30 = 40

$10^2 + 30^2$ = 1000

$20^2 + 20^2$ = 800?

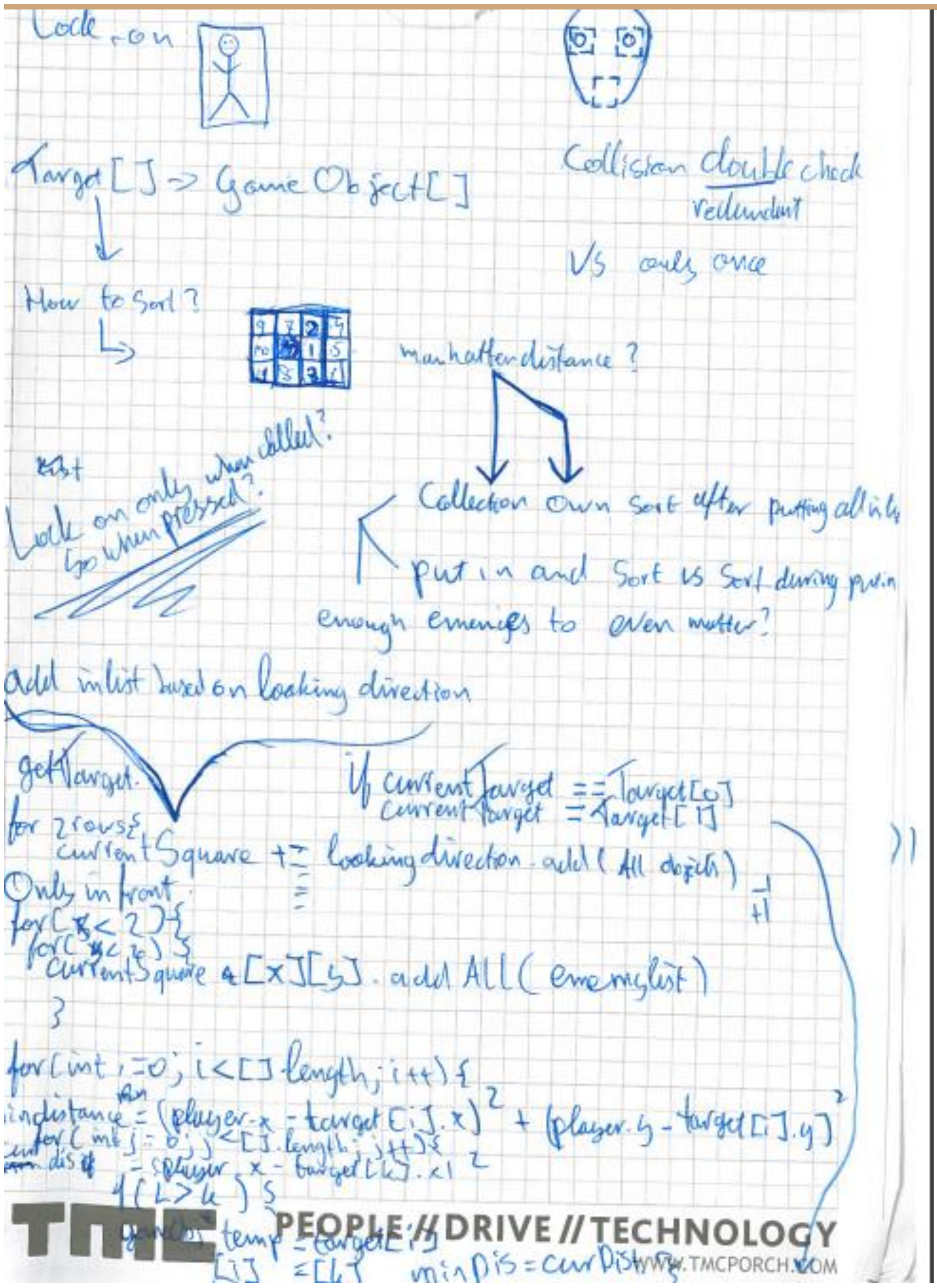
Pytago

1 = Sqrt($10^2 + 10^2$) = Sqrt(200)
2 = Sqrt($10^2 + 20^2$) = Sqrt(500)
3 = Sqrt($20^2 + 20^2$) = Sqrt(800)
5 = Sqrt (0 + 80^2) = Sqrt(6400)
4 = Sqrt[illegible] Pythagoras ($40^2 + 40^2$) = Sqrt(3200)

1 = 10 + 10
2 = 10 + 20
3 = 20 + 20
5 = 0 + 80
4 = 40 + 40

20
30
40
80
80

dus wel Δ



Lock-on

Target[] -> GameObject[]

How to Sort?

manhatten distance?

Collision double check redundant

Vs only once

~~List~~

Lock on only when called?

Go when pressed?

Collection own sort after putting all in lis

put in and sort vs sort during putin

enough enemies to even matter?

add in list based on looking direction

getTarget.

If currentTarget == Target[0]
currentTarget = Target[1]

```
for 2 rows{
    currentSquare += lookingdirection.add(All obj...)
Only in front
for(x< 2){
  for(y< 2){
    currentSquare[x][y].add All(enemylist)
  }
for(int i=0; i<[].length; i++){
mindistance = (player.x - target[i].x)^2 + (player.y - target[i].y)^2
  for(int j = 0; j<[].length; j++){
curdis = (player.x - target[k].x)^2
  if(L>k){
    temp = target[i]
    [i] = [k]
    minDis = curDis
```

PEOPLE // DRIVE // TECHNOLOGY
WWW.TMCPORCH.COM

gaat goed tot _ aanval 3 groot

Dan komt GameObject ⟼ Collidable List GameObject

Voor Character geen probleem.
desnoods check aan twee kanten.

Aanval op aanval kan dit een proble
worden.

↳ testen

↳ bepaalde een scenario's vermijd

↳ Dingen doen.

:-(

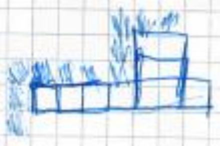
~~gridelement~~

levelElement.push_back(unit);

beide kanten checken ⟶ redundant check

Links en rechts to

player hit? } dag redundant check



Bibliography:

[1] J. Schell, *The art of game design: A book of lenses*, First Edition ed. Amsterdam: Elsevier/Morgan Kaufmann, 2008.

[2] "The Science of Playtesting", *GameSpot*, 2016. [Online]. Available: http://www.gamespot.com/articles/the-science-of-playtesting/1100-6323661/. [Accessed: 15- Apr- 2016].

[3] C. Clanton, “An Interpreted Demonstration of Computer Game Design,” in CHI 98 Conference Summary on Human Factors in Computing Systems, 1998, pp. 1–2.

[4] C. Crawford, S. Peabody, T. Art, C. G. Design, W. W. Web, D. Loper, and S. Peabody, “The Art of Computer Game Design by Chris Crawford,” 2003.

[5] M. A. Federoff, “Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games,” 2002.

[6] T. W. Malone, “What Makes Things Fun to Learn? Heuristics for Designing Instructional Computer Games,” in Proceedings of the 3rd ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems, 1980, pp. 162–169.

[7] R. J. Pagulayan, K. R. Steury, B. Fulton, and R. L. Romero, “Funology,” M. A. Blythe, K. Overbeeke, A. F. Monk, and P. C. Wright, Eds. Norwell, MA, USA: Kluwer Academic Publishers, 2004, pp. 137–150.

[8] K. Jegers, “Pervasive Game Flow: Understanding Player Enjoyment in Pervasive Gaming,” Comput. Entertain., vol. 5, no. 1, 2007.

[9] N. Lazzaro, “Why We Play Games: Four Keys to More Emotion Without Story”, 2004

[10] G. N. Yannakakis and J. Hallam, “Capturing Player Enjoyment in Computer Games,” in *Advanced Intelligent Paradigms in Computer Games*, Springer Berlin Heidelberg, 2007, pp. 175–201.

[11] J. Aderud, “Flow and Playing Computer Mediated Games - Conceptualization and Methods for Data Collection.”, 2005

[12] "10 Heuristics for User Interface Design: Article by Jakob Nielsen", *Nngroup.com*, 2016. [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/. [Accessed: 15- Apr- 2016].

[13] H. Desurvire, M. Caplan, and J. A. Toth, “Using heuristics to evaluate the playability of games,” in *CHI ’04 Extended Abstracts on Human Factors in Computing Systems*, 2004, pp. 1509–1512.

[14] T. Fullerton, “Chapter 9 Playtesting,” in Game Design Workshop: A Playcentric Approach to Creating Innovative Games, 2nd. Burlington, Elsevier, 2008, ch. 9, pp. 248–271

[15] S. J. Johansson, "What Makes Online Collectible Card Games Fun to Play?," Digital Games Research Association, 2009.