



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS

ÚSTAV INFORMAČNÍCH SYSTÉMŮ

**EDUCATIONAL CROSS PLATFORM MOBILE GAME
FOR YOUNGSTERS**

VZDĚLÁVACÍ MULTIPLATFORMNÍ MOBILNÍ HRA PRO MLÁDEŽ

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR

AUTOR PRÁCE

DAVID JANEČEK

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. OLENA PASTUSHENKO

BRNO 2018

Brno University of Technology - Faculty of Information Technology

Department of Information Systems

Academic year 2017/2018

Bachelor's Thesis Specification

For: **Janeček David**
Branch of study: Information Technology
Title: **Educational Cross Platform Mobile Game for Youngsters**
Category: User Interfaces

Instructions for project work:

1. Get acquainted with tools for cross platform (Android / iOS/ web) game development and JavaScript game engines.
2. Study the principles of gamification and meaningful play.
3. Perform a research on education games development.
4. Develop a UI prototype of the educational mobile game.
5. Develop an educational game based on the research (1-4).
6. Perform a usability testing, evaluate the results and propose further extensions

Basic references:

- Salen, K.; Zimmerman, E.: Game design and meaningful play. Handbook of computer game studies, 2005.
- Deterding, S.; Dixon, D.; Khaled, R.; Nacke, L.: From game design elements to gamefulness: defining "gamification". Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11, 2011.
- Malone, T. W.: What Makes Things Fun to Learn?: a Study of Intrinsically Motivating Computer Games. Xerox Corporation, Palo Alto Research Center, 1981.
- Hunicke, R.; LeBlanc, M.; Zubek, R.: MDA: A Formal Approach to Game Design and Game Research, 2004.
- Hirsh-Pasek, K.; Zosh, J. M.; Golinkoff, R. M.; Gray, J. H.; Robb, M. B.; Kaufman, J.: Putting education in "educational" apps: lessons from the science of learning. Psychological Science in the Public Interest, 2015.

Requirements for the first semester:

Items 1 to 4.

Detailed formal specifications can be found at <http://www.fit.vutbr.cz/info/szz/>

The Bachelor's Thesis must define its purpose, describe a current state of the art, introduce the theoretical and technical background relevant to the problems solved, and specify what parts have been used from earlier projects or have been taken over from other sources.

Each student will hand-in printed as well as electronic versions of the technical report, an electronic version of the complete program documentation, program source files, and a functional hardware prototype sample if desired. The information in electronic form will be stored on a standard non-rewritable medium (CD-R, DVD-R, etc.) in formats common at the FIT. In order to allow regular handling, the medium will be securely attached to the printed report.

Supervisor: **Pastushenko Olena, Ing.**, DIFS FIT BUT
Beginning of work: November 1, 2017
Date of delivery: May 16, 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
L.S.
Fakulta informačních technologií
Ústav informačních systémů
612 06 Brno, Božetěchova 2

Dušan Kolář

Associate Professor and Head of Department

Abstract

This thesis is about developing cross platform educational game for mobile devices. First, it lays out related theory and analysis of existing solutions. That is followed by description of some ways to create such game. Finally, there is draft, development and testing of the game using Unity game engine.

Abstrakt

Tato práce se zabývá vývojem multiplatformní vzdělávací hry na mobilní zařízení. Nejdříve je uvedena teorie a rozbor existujících her. Dále jsou v práci zmíněny některé možnosti tvorby takové hry a konečně popis návrhu, tvorby a testování samotné hry s využitím herního enginu Unity.

Keywords

Mobile game, Cross platform, Unity game engine, 2D, Educational game, Gamification

Klíčová slova

Mobilní hra, Multiplatformní, Herní engine Unity , 2D, Vzdělávací hra, Gamifikace

Reference

JANEČEK, David. *Educational Cross Platform Mobile Game for Youngsters*. Brno, 2018. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Olena Pastushenko

Rozšířený abstrakt

Mobilní hry již existují více než dvacet let[1]. Za tuto dobu prošly skrz několik období rapidních změn, které byly umožněny vývojem technologií mobilních zařízení. Od prvních her na černobílých obrazovkách, jako Tetris nebo Snake, až k obrovskému množství různých žánrů her vytvořených pro chytré telefony. Nejvýznamnější milníky na této cestě byl přechod z fyzických klávesnic na dotykové, a snadná dostupnost internetového připojení. V dnešní době může téměř kdokoliv vytvořit mobilní hru a díky oficiálním distribučním službám získat potenciální miliony zákazníků.

Hry jsou, nebo alespoň po určitou dobu byly, součástí života každého člověka. Někdo se na ně dívá pouze jako zdroj zábavy, hry ale často mají i další užitečný význam. Mohou rozvíjet užitečnou schopnost nebo učit nové znalosti, a to především u dětí[2]. Cílem této práce je vytvořit hru pro mobilní zařízení, která má takový pozitivní přínos. Není však určena pro výuku ve školských institucích, ale k využití ve volném čase.

Z počátku jsem neměl jasnou vizi, jak by výsledný produkt měl vypadat. Mnoho jsem si vyjasnil při studování materiálů potřebných pro teoretickou část práce. Další motivací pro mě byl fakt, že většina matematických her na mobilní zařízení jsou ve skutečnosti pouze generátory příkladů, které uživatel musí řešit. Dalo by se diskutovat, zda se vůbec jedná o hry. Rozhodl jsem se tedy vytvořit hru, která bude zahrnovat více než jen příklady. Konkrétně 2D plošinovku ve které jsou příklady využity při postupu hrou.

V teoretické části jsou dále rozebrány některé populární možnosti tvorby multiplatformních her pro mobilní zařízení. Na základě tohoto průzkumu jsem se rozhodl využít herní engine Unity. Nabízí bohaté možnosti při tvorbě 2D her a umožňuje nasazení na všechny nejpoužívanější mobilní platformy. Jako programovací jazyk slouží C#. MonoDevelop je vývojové prostředí pro psaní kódu přibalené k Unity, ale také je možné ho propojit s programem Visual Studio, což je možnost kterou jsem uvítal. Visual Studio nabízí mnohem více funkcionality, obzvláště při využití pokročilých rozšíření jako je ReSharper.

Prvním krokem, po ujasnění myšlenek a nápadů, bylo vytvoření návrhu uživatelského rozhraní. Při jeho tvorbě byly brány v potaz znalosti získané během studia zmíněné teorie. Některé věci jsem však opomenul a v průběhu vývoje se části uživatelského rozhraní oproti návrhu poměrně dost změnily. K tomu přispělo i rozsáhlé testování, které vedlo i k mnoha dalším změnám. Obrázky využití ve hře pochází od autorů, kteří je poskytují na internetu k volnému využití.

Ve hře si uživatel může vybrat ze dvou postav za které možné hrát. Každá z nich má vlastní animace. Na výběr jsou také dva jazyky, český a anglický. Přeloženy jsou veškeré části hry.

Testování bylo provedeno na čtyřech mobilních zařízeních s více než dvaceti uživateli. Začalo probíhat hned z počátku implementace až do samotného konce. Získané informace vedly ke změnám ve fyzice, uživatelském rozhraní a dalším menším úpravám ve všech ohledech.

Výsledná hra obsahuje patnáct kol, přičemž první z nich je tutoriál. Sám o sobě vysvětlí vše, co hráč potřebuje vědět a není tedy třeba dalších instrukcí. Cílem v každém kole je nasbírat určitý počet klíčů a nalézt východ, jehož aktivací je kolo úspěšně dokončeno. Prostředí se podobá klasické 2D plošinovce s posouváním kamery spolu s hráčem. Ve hře jsou rozprostřeny “aktivační zařízení,” která jsou vyobrazena jako cedule. Při aktivaci této cedule se zobrazí okno se třemi příklady, pokud je hráč správně vyřeší, pak se okno zavře a ve světě se něco změní v uživatelův prospěch. Například je odstraněna překážka v cestě, plošina se uvede do pohybu, klíč který byl mimo dosah je nyní možné získat apod. Řešení těchto úkolů je nutné k úspěšnému dokončení kola. Příklady jsou náhodně generovány a

jejich obtížnost, tedy rozsah generovaných čísel a druh matematické operace, se dá upravit v nastavení, ale také se řídí číslem daného kola, přičemž obtížnost postupně roste. Obtížnost zvolená uživatelem má o mnoho větší vliv, než ta průběžná. Díky tomu mohou hru hrát uživatelé s výrazně rozdílnými matematickými znalostmi a dovednostmi. Hra je primárně určena pro děti ve věku od devíti do patnácti let. Nejjednodušší příklady zvládnou spočítat i děti mladší, ale mají problémy s ovládním a orientací ve hře. Není žádný důvod, proč by hru nemohli hrát i starší jedinci, ale procvičování takovýchto příkladů pro ně již není velkým přínosem.

Educational Cross Platform Mobile Game for Youngsters

Declaration

Hereby I declare that this bachelor's thesis was prepared as an original author's work under the supervision of Ing. Olena Pastushenko. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
David Janeček
May 14, 2018

Contents

1	Introduction	3
2	Gamification and Meaningful Play	5
2.1	Play	5
2.2	Rules	6
2.3	Goal	7
2.4	Meaningful Play	7
2.5	Gamification	8
3	Research on the Existing Educational Mobile Games and Development Tools	9
3.1	Existing Educational Games	9
3.1.1	Calculator: The Game	11
3.1.2	100 Logic Games - Time Killers	11
3.1.3	Lumosity	12
3.2	Mobile Platforms	13
3.2.1	iOS	13
3.2.2	Android	14
3.2.3	Windows 10 Mobile	14
3.3	Hybrid Applications	15
3.4	Mobile Games Market	15
3.5	Development Tools	16
3.5.1	Unity	16
3.5.2	Unreal Engine	17
3.5.3	GameMaker Studio	17
3.5.4	Phaser.io	17
4	Pre-development	19
4.1	Target Audience	19
4.2	Use Case	20
4.3	Chosen Technologies	20
5	Development	22
5.1	User Interface	22
5.1.1	Draft	22
5.1.2	Final version	23
5.2	Graphic Design	28
5.3	Animation	28

5.4	Difficulty	28
5.5	Text Localization	28
5.6	Code Conventions	29
5.7	Gameplay	29
6	Testing	31
6.1	Outcome of Testing	31
6.2	Devices	31
6.2.1	Android	32
6.2.2	Universal Windows Platform	32
6.2.3	iOS	33
7	Further Development	34
8	Conclusion	35
	Bibliography	36

Chapter 1

Introduction

Mobile games have been around only for about twenty years[1]. During this time they went through several phases of rapid changes that were enabled by the development of cellphone technology. From the first few games like Tetris and Snake on black and white displays to the large amount of genres of games made specifically for smartphones. The transition from physical buttons to today's omnipresent touchscreens and spread of the Internet were two of the most significant breaking points. With the current access to technology, nearly anyone can make a game and distribute it to potential millions of customers through official stores. Because of that, there is immense competition and prices driven to the ground. Most games are now free and focus on income from advertisements and in-game purchases.

Playing games is a part of everyone's life and often perceived just as a form of entertainment. Such perception is not always correct. Many games serve multiple purposes and help develop useful skills. This is especially true for children[2]. Game that is a product of this work is an attempt to achieve some of those positive effects while keeping it fun to play.

Voluntary learning through fun games is known to be very effective[3]. That is mostly because inner motivation is stronger than outer motivation. In general, getting students motivated has been a problem in education for a very long time[4]. Advancements in technology, utilizing games and gamification could help solve this problem, but it is important to proceed carefully. So far little is known about the effects of gamification elements in learning environment[5] and it might bring more problems than benefits. School systems are generally very old and reformations are hard to put in place, because they come with a risk of making things worse. Good example of that is current situation in Czechia regarding teaching of mathematics in primary schools. More than 700 of them started using alternative method put together by professor Milan Hejný[6]. Its aim is to increase students' interest in this mostly unpopular school subject, and also put more emphasis on developing logical and critical thinking. The method motivates students to discover their own solutions to problems, instead of getting them from the teacher and learning them without proper understanding. It also focuses on cooperation and teamwork[6]. After its initial success, the method is now facing a lot of criticism from both teachers and parents as it seems to slow down students' progress[7].

With that said, the aim of this work is to create an educational game for mobile devices that is not entirely focused on just one activity or task that could be used during lessons of specific subjects in educational institutions. My main motivation comes from the fact that mathematical games for mobile platforms usually provide nothing but equations to solve. I think that this is a shame and that no one can really enjoy these games for long. So I

decided to try to make a game that would incorporate more and if possible also improve logical thinking and problem solving of its users as I believe that is something that is lacking in our current educational system. At the same time, the game should remain entertaining and should be played in free time.

This thesis first lays out some information regarding the concept of games and play. That is followed by research of relevant market and current state of mobile game development. Further chapters are devoted to the process of making the game. From conceptual problems to development and testing.

Chapter 2

Gamification and Meaningful Play

Game is a form of play with a set of rules[8]. Games have been around for a very long time and exist in all human societies. Complex games, some more than 5000 years old, were found in archaeological excavations. Good example of such game is Senet, board game placed in Ancient Egypt in 3100BC[9]. Such findings show that games have a quintessential, underlying meaning in human society, otherwise they would not be around for so long.

Clinical psychologist and professor of psychology at the University of Toronto Jordan Peterson says that children’s ability to play games with others is a prerequisite for being socialized[10]. The key elements of games are cooperation, competition, rules, players’ agreement on said rules. He adds that “games are a simplified analog of life, and if you get good at playing games, what happens is that the games you play become increasingly complex and more and more real world like until they’re actually indistinguishable from acting in the world.” The idea that play has an active role in children’s learning has been reinforced by psychologists for a long time[2].

Since this thesis is about creating a mobile game, which is a kind of video game, it would seem reasonable to devote a paragraph to video games and their difference from normal, non-video games. While almost all normal games require more players, video games are often made just for a single player. They also require electricity to run, because player interacts with an electronic machine. Such device has to be capable of receiving input and providing feedback. This is done by some sort of input device, such as controller, keyboard or a joystick. Feedback is usually provided via a video screen which can come in many different forms and sizes.

2.1 Play

As mentioned at the beginning of this chapter, game is a form of play. In this section it would be discussed what constitutes play. Play is much older than human civilization. Animals play just like people. For example, young dogs often wrestle each other, but have no intention to harm each other, the same behavior can be seen in human children, especially boys. Specific results of psychological research of play differ. Origin and fundamentals of play have been described as a discharge of superabundant vital energy, by others as the satisfaction of some “imitative instinct”, or as a simple need for relaxation. Another research shows that play is a training and preparation of young creature for the serious work that comes in its adulthood. Those results partly overlap and it is possible to accept nearly all of them.[11]

It is clear that play is a very complex phenomenon. Its full psychological meaning is still a matter of debate and further research. We will not go into any more detail in this thesis.

2.2 Rules

Rules are foundations that games are built upon. They dictate what players can and cannot do, what the victory conditions are, and essentially define the entire game[12].

Good rules are explicit and unambiguous. At the same time they should be easy to comprehend and written down in such way that it does not take too long to read through them and start playing. This can be very hard to do for some games. Popular card game Bang! is quite complex, but handles rules exceptionally well. All cards have explanation on them, many using simple legend in a similar fashion to maps. This allows new players to start playing the game after learning only the most basic rules and pick up the rest while playing.



Figure 2.1: Card game Bang! using legend similar to maps to explain rules (Czech version)

In a game with more players, all of them have to operate under the same set of rules. Otherwise the game could break down. Therefore rules have to be enforced in some way just like in real world. In friendly casual setting, it is up to the players to make sure that everyone follows the rules. Special actor, referee, has to be introduced in more competitive setting. This is most common in sports. Cheating in online video games has been a problem since their origins. Early online games usually did not include any anti-cheat protection. Many players resorted to cheating which caused decline in overall number of players as the honest ones stopped playing.[13] That is why developers these days spend a lot of time to make their multiplayer games cheat-proof.

2.3 Goal

Even rich, responsive environments may be unappealing unless they provide an appropriate goal. Part of that is an uncertainty of outcome. Game is not challenging if the player is either certain to win or certain to lose. This can be achieved by implementing several difficulty levels.[12] Those can be set automatically by the program according to how well the player does or chosen by the player himself. In case of multiplayer games, players of similar skill level should be matched against each other to achieve ideal conditions.

Another option is having multiple goals with different difficulty. One goal can be to finish a level, and another is to finish the level as efficiently as possible. This usually involves some way of score-keeping. For example, the most efficient solution results in receiving three star rating, while finishing the level in slightly less efficient manner gives player only two stars and so on. The score can reflect other factors like number of tries, elapsed time, resources expended, etc.[12]

Not all games can utilize these methods to ensure uncertainty of outcome. Fortunately, there are more ways, some of which I will mention. It is possible to simply introduce randomness into the game[4]. This is main principle of all gambling games, but can be utilized in almost any game. Events, number and strength of enemies, loot or entire levels can all be randomly generated. This is easier to do for video games than for example board games. Obviously it is important to keep the randomness within some reasonable bounds. Another way to make the outcome of a game uncertain is by hiding information from the player, and selectively revealing it. Such features provoke curiosity and contribute to the challenge of the game.

2.4 Meaningful Play

Play does not come just from the game itself, but from the way that player interacts with it and with other players. Game is given meaning if player's choices and actions result in different outcomes and affect the overall system of the game[12]. This can be a problem in cooperative multiplayer games, where one player might play very well, but still lose the game, because his teammates performed poorly. Such outcome often leads to frustration especially in highly competitive environments.

The general idea is true for all games, but some games have more meaning than others. When player makes a move in chess, the whole board is affected. We can see an immediate outcome which is very clear - figure is taken out, king is under attack etc. Because the overall situation changes, there is also a long term outcome which is harder to spot and might not directly correspond with the immediate. Losing a figure can give a player an advantage in certain situations. It is this kind of deeper meaning that makes games more meaningful and engaging. [12]

Video games offer near infinite options in this regard as they are not limited by physical laws and space. Some of them may at times require player to make several precise actions per second, which demands absolute focus. This might be the reason why video games are more captivating than other games.

2.5 Gamification

Gamification is a process of adding game-like elements into something that is not a game, in order to encourage participation[14]. It is mostly used in business and marketing, but it has also seen a rise in popularity in eLearning[5]. But not everyone is successful with it. Gamification is often misused and poorly implemented by focusing solely on surface-level features[15]. Simply adding a load of achievements and trophies without further meaning to users does not do the trick, and only leads to unnecessary distractions.

User experience designer and game researcher Sebastian Deterding has put together a set of recommendations for successful usage of gamification[16]. He says that in order to get it right, there has to be a goal, ideally a customizable one, that the user wants to achieve in the first place. Goals also need to get increasingly difficult in a non-repetitive way. Another problem that designers might face is that if challenge is too easy, it leads to boredom. On the other hand if it is too hard, it leads to anxiety. The ideal spot which gives users the best feeling is in the middle of the road and provides decent, but achievable challenge.

Gamification via mobile apps is a fairly new way to create fun, engaging teaching environment. Some educational institutions are already experimenting with this approach. The reactions are mixed. Some argue that gamification leads only to a short-term motivation, while others see it as the future of education[5][15]. More time is still needed until we fully understand the advantages and disadvantages of gamification and its usage in education.

Chapter 3

Research on the Existing Educational Mobile Games and Development Tools

Automation and computerization caused major changes in the nature of human work. Two major sets of jobs can be identified that rose to significance. Jobs traditionally held by the lower class like janitors, waiters or security guards have grown in importance. Even greater job growth has taken place on the opposite side of the pay distribution - managers, doctors, lawyers and engineers[17]. What is common for both of these categories, is that all of these jobs are not easily replaceable by computers and machinery. Educational institutions have to adjust to this trend in order to make their students successful and competitive in their future working lives.

Usage of technology in schools is certainly on the rise, but it is not always done in regard to learning the newly required skills. Students collect information for reports from the Internet instead of the library, they type reports on word processors instead of typewriters, and make multimedia presentations instead of on poster boards. But the content and skills are largely the same[18]. New ways to tackle this and other problems of education are being created.

Overall the trend is to make education more fun and engaging. One possible approach to combine education with entertainment is described by the term “edutainment”[5]. It heavily relies on visual material and on narrative or game-like formats. The purpose of edutainment is to hold attention of the learners with colorful environments. Another approach is to create games that are designed to teach, sometimes also called serious games[15]. The product of this thesis falls into this category.

3.1 Existing Educational Games

It is hard to draw a clear line between games that are, and that are not educational. This section focuses on existing mobile games that have a teaching aspect to them or help improve certain skills (e.g. critical thinking, problem solving, memory, etc.) and are in some ways relevant to the product of this thesis. After a brief overview, some interesting games are described in detail.

There is a large variety of educational mobile games aimed at children. Most of them are made for preschoolers and teach simple things, such as colors, letters, numbers and

recognizing animals. All of the games in this category offer very similar exercises wrapped into different environments.

Another recurring theme which is the most relevant to my game are mathematical games. They come in varying complexity. The most basic ones are nothing more than math problem generators, which raises the question if they should even be considered games. A good example of that is Math Game. All it contains are 200 simple equations. There are a bit more complex games, for example Pure Math, which has different game modes and also score system. That is the most common state of math games right now. An interesting exception is an app called Math Fight, which is unique in the fact that it is a split screen multiplayer game. Players get a point for the fastest correct answer and lose a point for each incorrect answer. First to reach ten points wins the game. Another notable exception that I managed to find is Calculator: The Game which we will look at in detail in section [3.1.1](#).

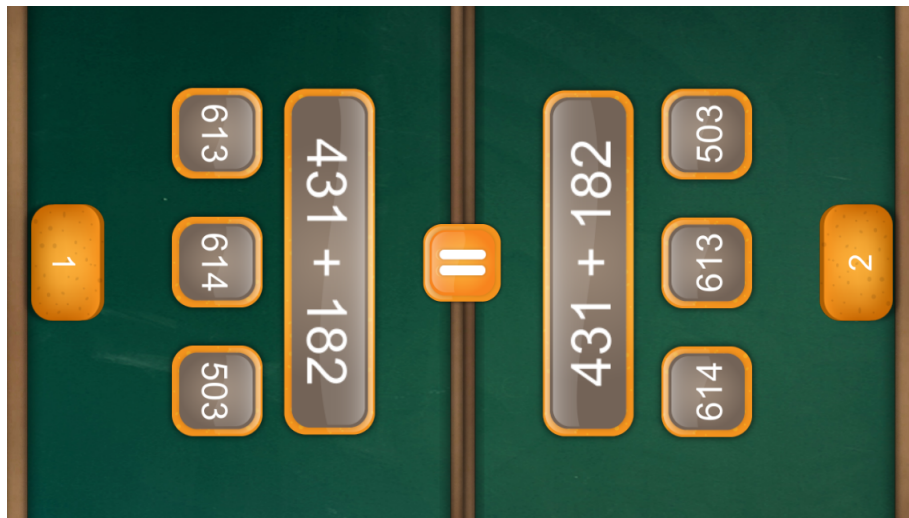


Figure 3.1: Math Fight, mathematical game for two players

Much more diverse are puzzle games. It might seem as a bit of a stretch to consider puzzle games as educational, but I included them for reasons mentioned in the first paragraph of this section. Almost all puzzle games have one type of task that repeats in levels of varying difficulty. We can use Sudoku, which exists also as a mobile game, as an example of this. It requires player to place numbers 1 to 9 into a 9x9 grid in such way, that each row, column and 3x3 box contain the same number only once. With different layout of starting numbers, there are near infinite possible levels with varying difficulty, but the overall activity stays the same and may become boring after a few levels. Some games try to fight this by introducing new features as the player progresses. Mobile game Cut the Rope does this. Goal of each level in this game is to deliver a candy to a little green monster that eats it. That can be done by cutting ropes that are attached to the candy. Along the way, player can pick up up to three stars for extra challenge. New features are introduced every few levels. These include different kinds of rope, bubbles that make the candy fly upwards, spikes and many more.

Last genre that I would like to mention are brain training apps. They usually consist of multiple games, and are meant to improve player's thinking and memory. These games often utilize gamification to motivate players and keep them playing regularly, which should

help deliver the desired effects. As a skeptical person, I have to say that many of these games look iffy and probably do not do as much good as they claim to. One of the most prominent games in this genre is Lumosity, which is why I decided to talk more about it in section 3.1.3.

3.1.1 Calculator: The Game

Calculator: The Game is unique mathematical game primarily aimed at children. It has very simple user interface, which is divided into two parts. Calculator display and buttons, hence its name. The game is divided into levels. In each level, the player is given a goal (number to be reached) and limited amount of moves to achieve this goal. In order to do that, player has to manipulate starting number with provided buttons. Functions of these buttons change every round and offer addition, subtraction, conversion, reversion and more. Last main feature of graphical user interface is “Clicky, the snarky assistant”. Clicky is an animated smiley face. It communicates with the player, presents the game and introduces newly unlocked features. Audio of the game is very simple. The only sound effects are button press and announcement of correct or wrong result.



Figure 3.2: Calculator: The Game level screen

3.1.2 100 Logic Games - Time Killers

As the name suggests, 100 Logic Games - Time Killers is actually a collection of games. All of the games share some similarities. There is always a grid-like playing board that contains some objects or is divided into sections. In every game, the player has to solve the puzzle by placing other objects into the board according to certain rules. Games utilize both the tiles and edges of tiles. I will explain first of the games in detail. It is called Parks and the rules are quite simple. Playing board is divided into colored sections - parks. Each park, row and column must have exactly one tree. A tree cannot touch another tree, not even diagonally. Player’s goal is to plant trees following these rules. It is possible to mark tiles where trees cannot be placed with a white dot. The limit of trees in each park, row and column is increased in advanced levels.

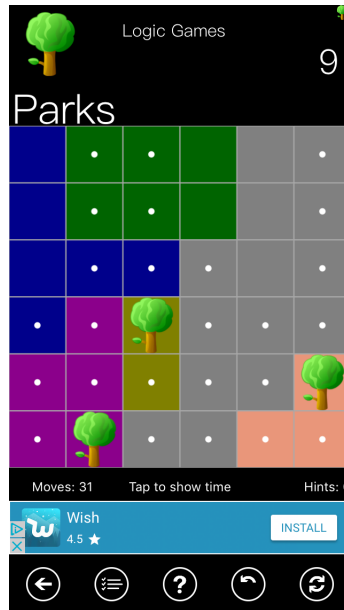


Figure 3.3: Parks, one of the 100 Logic Games

3.1.3 Lumosity

Lumosity is a mobile application and online program comprised of about fifty games that are meant to improve users' cognitive abilities. Those are divided into several categories including memory, attention, problem solving and math. Lumosity is, in my opinion, the best brain training app out there. Basic features are available for free, but many are accessible only with a premium account, which is understandable and common practice for applications that do not earn money through advertising. Lumosity is claiming that there is extensive research behind all of their games. While that may be true, they certainly went too far with their claims while advertising the product. Federal Trade Commission decided that Lumosity's advertisements deceived customers with unfounded claims. Jessica Rich, Director of the FTC's Bureau of Consumer Protection stated, "Lumosity preyed on consumers' fears about age-related cognitive decline, suggesting their games could stave off memory loss, dementia, and even Alzheimer's disease. But Lumosity simply did not have the science to back up its ads"[19]. This led to a \$2 Million settlement.

One of the Lumosity games in math category is called Raindrops. As the name suggests, there are raindrops with mathematical problems displayed in them. They are generated at the top of the screen and move downwards. Player has to enter correct answers to these problems to make the raindrops burst and disappear. Game is over when a raindrop reaches bottom of the screen. The game that I found most interesting and fun within Lumosity is called Train of Thought. Trains of different colors spawn from a tunnel over time. Player has to guide the trains to stations of corresponding color. That is done by changing direction of railroads by tapping on railroad switches. The game is designed to improve divided attention. Lumosity games automatically adjust to player's skill. For example in the Train of Thought game, speed at which the trains spawn changes based on the percentage of correctly directed trains.

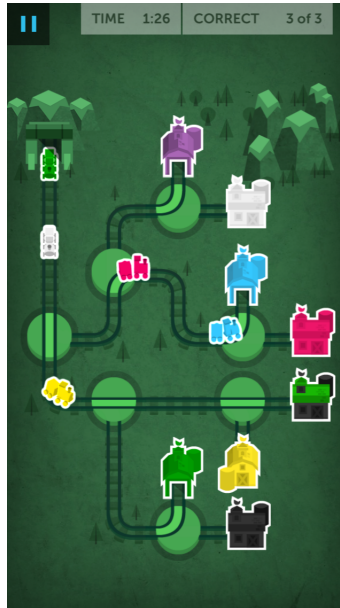


Figure 3.4: Lumosity game Train of Thought

3.2 Mobile Platforms

The platforms that I decided to target are smartphones running on operating systems iOS, Android and Windows 10 Mobile. The first two come as an obvious choice, as they are by far the most used mobile platforms[20]. I decided to target Windows 10 Mobile as well, because it is the only other relevant operating system backed by a major company, and also because I like the system and have it installed on my personal phone. At the time of making this decision, Microsoft had not yet ceased active development of the platform. Following sections detail each one of these platforms.

3.2.1 iOS

iOS is operating system developed by Apple. It runs on most of the company's mobile devices, such as iPhone, iPad and iPod. The most recent version is iOS 11 which introduced many new features. The user interface is based upon direct manipulation, using multi-touch gestures. Important part of iOS is Siri, an intelligent personal assistant. It supports a wide range of commands that users communicate with in natural language. Another core feature inseparable from iOS is the App Store. The store allows users to browse, buy and download applications developed with Apples's development kit. There are more than two million applications available[20].

Every device that is using macOS has an integrated development environment called Xcode. Xcode contains development tools for all Apple's operating systems[21]. It supports many programming languages including C++, Python and Java. Xcode is also used for publishing on the App Store.

3.2.2 Android

Android is operating system developed by Google. It is based on the Linux kernel and designed for smartphones and tablets. Several major manufacturers, including Samsung, Sony and LG, have their phones running on Android. Current version is 8.1 „Oreo“. It is a tradition for Android to name its versions. There is no built-in personal assistant like Siri, but a lot of similar applications can be downloaded from Android’s store - Google Play. Google Play offers the largest amount of applications of all stores[20]. Those have to be developed with Android software development kit and there are nearly three million of them. Android is dependent on it’s Java architecture and so it is also primary language for developing apps. Google Play also serves as a digital media store with music, magazines, books and movies.

3.2.3 Windows 10 Mobile

Windows Phone is an operating system developed by Microsoft. It was replaced by Windows 10 Mobile in 2015 which provided major upgrades to the system and focused on larger integration with Windows PC. With that came Universal Windows Platform which is a platform-homogeneous application architecture created to help develop universal applications that run on Windows 10, Windows 10 Mobile, Xbox and Hololens. Almost all smartphones using Microsoft’s operating systems have been manufactured by Nokia. Further development of the system is on halt and will most likely never continue, because of low interest of buyers and application developers.

Major development tool for all Microsoft products is Visual Studio. Thanks to the single Windows core in Universal Windows Platform, the same app can run on any Windows 10 device. Preferred languages are C#, Visual Basic, C++ and JavaScript.[22]

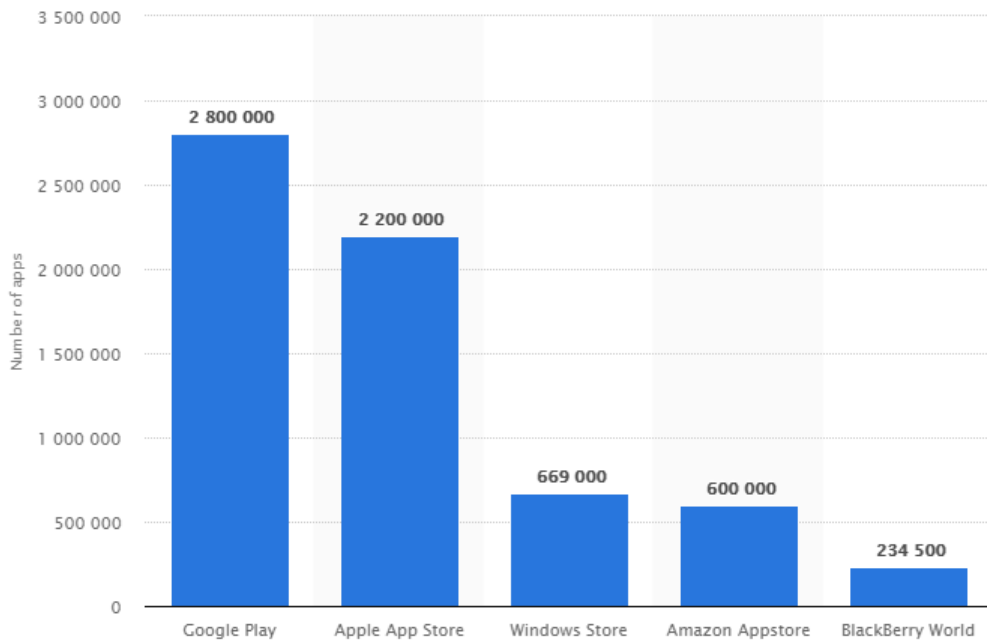


Figure 3.5: Number of apps in app stores as of March 2017[20]

3.3 Hybrid Applications

Another option when making an application for multiple platforms is to make a hybrid application. Hybrid apps use a combination of features from both native and web apps.[23] They are made using web technologies, HTML, CSS, JavaScript, which overcomes problems with platform specific differences, but they are downloaded and installed like native apps from app stores. Internet connection is usually required for full functionality of such app, but it can run even when it is not connected. Another advantage over purely web apps is that hybrid apps can utilize phone's camera, GPS and other devices.[24]

3.4 Mobile Games Market

Smartphones have become almost a necessity in people's lives. They are no longer just phones, but small, portable computers with capability to make phone calls. On top of everything that a computer can do, smartphones also serve as a handy camera, flashlight, compass, GPS navigation and more. According to research conducted by multiple organizations, mostly in The United States of America, people spend more than four hours a day using their mobile phones[25]. Most of this time is spent using apps. It is no wonder that modern cellular phones have also become a major player on the gaming market.

Income of all major app stores is mostly made by games. Mobile gaming industry generated over 30 billion dollars in revenue in 2015. Asia is the largest region based on mobile games revenue, three times larger than second North America. At the end of 2016, there was an average of 2.8 billion monthly active users of mobile games worldwide. Gamers on average play 3.6 games per month. The most popular genres are arcade and action games.[20]

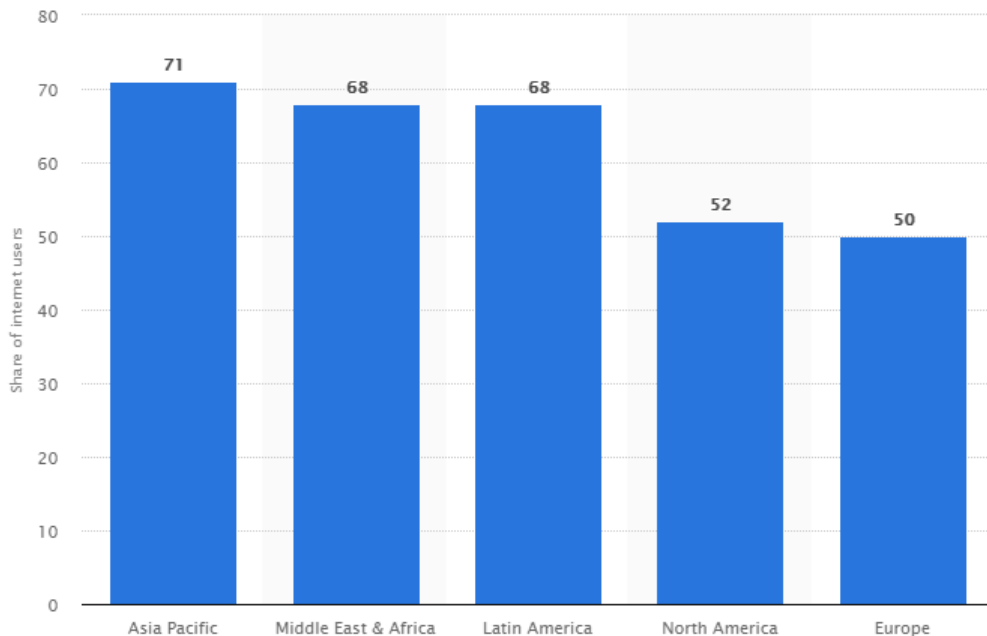


Figure 3.6: Percentage of Internet users who use a smartphone to play games worldwide as of 2nd quarter 2016, by region[20]

3.5 Development Tools

Creating applications for mobile devices has become easier than ever. It is especially true for games. There are many high quality development tools and assets that are available for free or at least have a free version that is lacking some features. Publishing games is no longer possible just for big companies, but for pretty much anyone with Internet access. Therefore even a single person can develop a successful game with a very small budget. As an outcome of all of this, mobile game development is rapidly evolving industry worth billions of dollars[26]. Several options that I considered for developing a mobile game are mentioned in this chapter in detail.

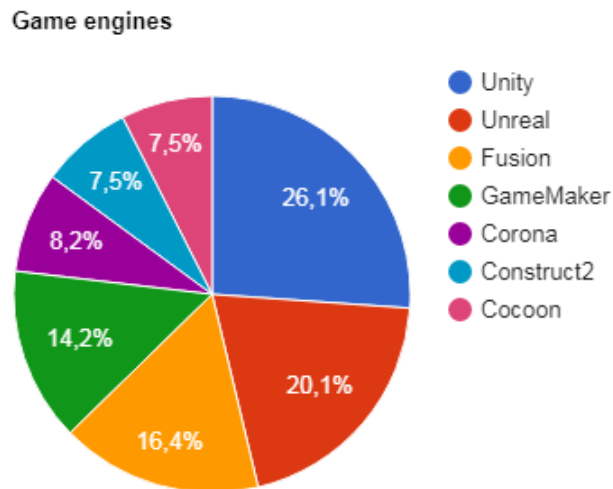


Figure 3.7: Percentage of usage of game engines in 2017[27]

3.5.1 Unity

Unity is a multipurpose game engine developed by Unity Technologies. It supports 2D and 3D graphics. It has a complex graphical user interface that features drag and drop functionality. Unity uses C# for scripting. Personal license is free and includes all core engine components. Plus and Pro licenses include extra features such as extended analytics, performance reporting, better skin editor UI, etc. They cost \$35 and \$125 per month respectively. All plans include continuous updates. Teams of 21 or more can also buy custom solutions. This option is broadly used by larger companies including giants like Blizzard, Electronic Arts or Ubisoft.[28]

Unity supports more than 25 platforms across mobile, desktop, console, TV, virtual reality and web. Developers of multiplayer games can use Unity-provided servers. It is very easy to monetize games with built-in advertising system. Unity is not so easy to pick up as some other tools, especially for people that are not familiar with C#. But it has a very well made, detailed documentation with countless examples and tutorials. There is a massive community surrounding Unity which produces even more materials. Some of the most popular games, like Hearthstone: Heroes of Warcraft, Rust or Kerbal Space Program, were made using Unity.

Unity Asset Store

The Unity Asset Store is a growing library of free and commercial assets created both by Unity Technologies and also members of the community. It offers a wide variety of assets, including everything from textures, models and animations to whole project examples. All assets can be accessed straight from the Unity Editor and downloaded and imported directly into a project. The store provides a browser-like interface which allows for easy navigation and search of items.

3.5.2 Unreal Engine

Unreal Engine is one of the oldest and most popular game engines. It was made by Epic Games primarily for first-person shooters, but it has branched out since then. Currently at major version 4, there is support for all most popular platforms, including consoles and virtual reality[29]. Its graphical user interface uses drag and drop system. In general, working with Unreal Engine and Unity is very similar. Many features have a different name, but are essentially the same. For example properties of game objects are shown in Details (Unreal) and in the same way in Inspector (Unity). Main coding language of Unreal Engine is C++, but it also offers alternative called Blueprints. Blueprints is a very powerful visual scripting system. It allows implementation of almost everything that can be done with C++. Assets can be downloaded from Unreal Engine Marketplace. Many notable games, such as Bioshock, Mass Effect or Borderlands, were made using Unreal Engine.

3.5.3 GameMaker Studio

Another commonly used tool is GameMaker Studio 2 which is the latest incarnation of GameMaker. Just like Unity and Unreal, it utilizes drag and drop system which makes the environment fairly similar. Unlike them, it is focused solely on 2D game development. It has functionality to create three-dimensional graphics and effects, but there is no actual third dimension[30]. GameMaker focuses on making it as easy to use as possible, which makes it a suitable tool for novice game developers. Even unexperienced programmers can fairly quickly learn its scripting language - Game Maker Language (GML). GML's syntax is similar to Ruby or Python. Its functionality is again tailored for simplicity, and is very limited compared to C++ or C#. Major downside of having its own programming language is that there is no advanced editor which would offer complex functionalities like refactoring. Big upside for 2D game developers working with GameMaker are built-in tools, such as tileset editor, which can make work on traditional game genres faster and easier. The real issue that I take with GameMaker is its price. There are many license options, some monthly, others permanent, but licenses are also platform specific. That means that in order to deploy to multiple platforms, it is necessary to get multiple licenses which results in hundreds of dollars in expenses.[31]

3.5.4 Phaser.io

Phaser is free open source framework for making HTML5 games. It uses a custom build of Pixi.js. In order to make actual applications, not just browser games, third party software has to be used. Phaser allows you to develop your games with JavaScript or TypeScript, which is a typed superset of JavaScript that compiles to plain JavaScript. Phaser has been split into three major versions. Last official release of Phaser 2 is Phaser 2.6.2. Open source

community continued to work on Phaser 2 and created community version that is currently at version 2.8.3. Main focus of developers is on Phaser 3, which has not yet been released and is in alpha testing. Source code of phaser is publicly available on github.[\[32\]](#)

It is fairly easy to learn to develop games with Phaser, especially for people who are already familiar with inner workings of HTML and JavaScript. Community around Phaser is very active and produces a lot of tutorials and other useful materials. However, the documentation could still use improvement. Overall, Phaser is a good tool and if I were to make a browser game, I would definitely use it.

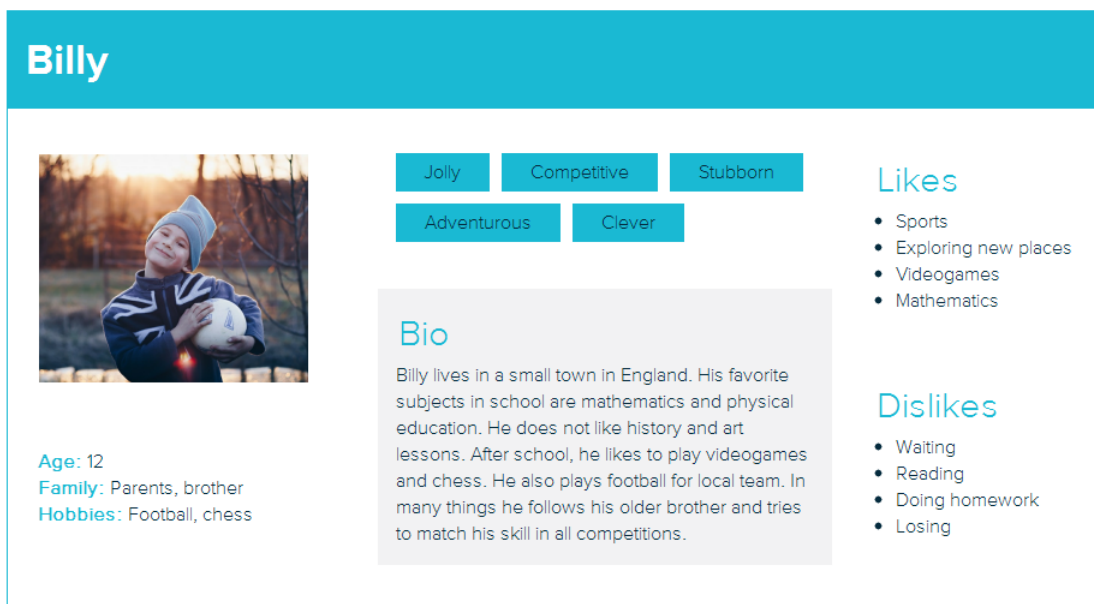
Chapter 4

Pre-development

After conducting research, I used the obtained information to create a more specific idea of the game that I was going to make. That is what this chapter is about. First of all, I made a more detailed specification of the target user group. During further work I always tried to keep this information in mind. Another very important step was to decide which technology, development environment, etc. I was going to use to create the game.

4.1 Target Audience

This game is made primarily for children that are around 9 to 15 years old. It would probably be too hard for anyone younger than that. As it tends to be with games like this, there is no real upper limit and it is entirely possible that someone of advanced age might enjoy the game as well.



The image shows a primary persona card for a character named Billy. The card has a teal header with the name 'Billy' in white. Below the header is a photograph of a young boy in a winter hat and jacket holding a soccer ball. To the right of the photo are five teal boxes containing personality traits: 'Jolly', 'Competitive', 'Stubborn', 'Adventurous', and 'Clever'. Below the photo, there are three lines of text: 'Age: 12', 'Family: Parents, brother', and 'Hobbies: Football, chess'. To the right of the photo is a 'Bio' section with a grey background, containing a paragraph about Billy's life in England, his school subjects, and his interests. To the right of the bio are two sections: 'Likes' with a bulleted list of 'Sports', 'Exploring new places', 'Videogames', and 'Mathematics'; and 'Dislikes' with a bulleted list of 'Waiting', 'Reading', 'Doing homework', and 'Losing'.

Billy

Age: 12
Family: Parents, brother
Hobbies: Football, chess

Personality Traits: Jolly, Competitive, Stubborn, Adventurous, Clever

Bio
Billy lives in a small town in England. His favorite subjects in school are mathematics and physical education. He does not like history and art lessons. After school, he likes to play videogames and chess. He also plays football for local team. In many things he follows his older brother and tries to match his skill in all competitions.

Likes

- Sports
- Exploring new places
- Videogames
- Mathematics

Dislikes

- Waiting
- Reading
- Doing homework
- Losing

Figure 4.1: Primary persona, created using xtensio.com

4.2 Use Case

The only role in this application is its user - the player. Several actions are available in main menu. These actions include setting language, choosing level and changing player character. While in a level, the user practices mathematics related skills and also improves perception, hand-eye coordination and problem solving.

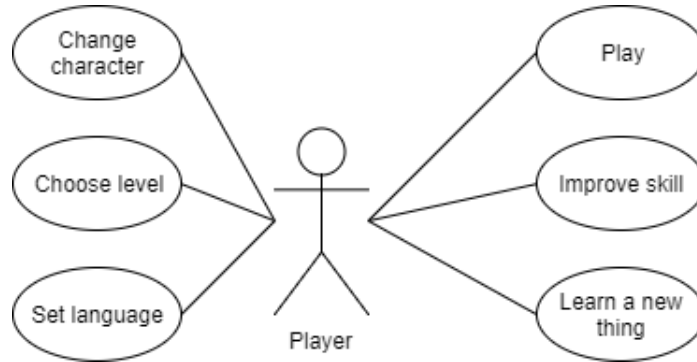


Figure 4.2: Use case diagram

4.3 Chosen Technologies

After researching development options that are available to me, it came down to two options. Either Unity or Unreal Engine. Both are top notch tools and are quite similar to each other. But when it comes to 2D, Unity has the upper hand. It contains useful tools made specifically for 2D development, like sprite editor. There is also much greater variety of 2D assets in Unity's asset store. I am familiar with both C# and C++, but prefer C#. So in the end I decided to use Unity.

MonoDevelop is integrated development environment supplied with Unity, but Unity can also be connected to Visual Studio without losing any features. There is no doubt that Visual Studio offers way more functionality. But it comes with a cost in the form of much higher demand on RAM and CPU, especially when using multiple extensions. Fortunately, my personal computer is quite good and I was able to use Visual Studio 2017 with Tools for Unity extension that makes Visual Studio correctly highlight and auto-complete Unity specific functions. Another very useful extension that I used is ReSharper by JetBrains. It is one of the most demanding extensions, but it is totally worth having. It runs on-the-fly code quality analysis and has a lot of other functions for code management and refactoring. Visual Studio is also used for deploying to Windows Phone devices.

Main part of projects made with Unity are scenes. Scenes contain objects and are mostly used as unique levels and menus. Basic objects are the most fundamental concept in Unity. All items, characters, cameras, etc. are game objects with specific components. Components can be thought of as properties of the game objects. The most basic component that every object has to have is the transform component, which defines object's position, rotation and scale in the scene. Other components define object's appearance, function, behavior, etc. Multiple game objects can be combined and used to create prefabs. Prefab acts as a template from which it is possible to create new object instances in the scene. Any edits made to a prefab are immediately reflected in all instances produced from it. But it is

also possible to override components and settings for each instance individually. Introducing scripts to the game is done by attaching them to game objects. Another approach is to create a scriptable object. Those usually don't have a visible representation in the game and are most useful for assets which are only meant to store data.

Chapter 5

Development

First step after sorting out ideas in my head was to create draft of user interface. When that was done, I gathered all the assets that I needed and I was ready to start with the implementation. I started by creating player game object and a few other objects. Basic components were added to them and essential game logic implemented. Soon after, I created multiple prefabs for platforms, environment and most importantly the activation device. At this point the game was “playable”, but nearly everything was edited in some way during further development.

When that was done, the following implementation was mostly pretty straight forward. I kept creating new levels and adding new features whenever an idea arose. Changes and fixes were continuously introduced based on feedback obtained from testing. Some of the steps that I made during development and interesting parts of implementation are described in detail in this chapter.

5.1 User Interface

User interface is important part of every application. Bad UI makes users reject even otherwise good apps. Critical problems like dysfunctional buttons are amongst the most frustrating. That is why decent amount of time and attention was invested into UI during development and testing. Main goal is to make well-arranged, interactive and easy to understand UI that requires minimum steps to achieve desired action. In case of UI, I believe that one look is worth a thousand words, therefore there are many pictures present in this section.

5.1.1 Draft

Menu

The entire application operates in landscape mode. Main menu allows user to choose level and player character. That can be done by using arrows to the right and left of overviews of level and character respectively. User can also alter language settings. All buttons in the game change color on press.

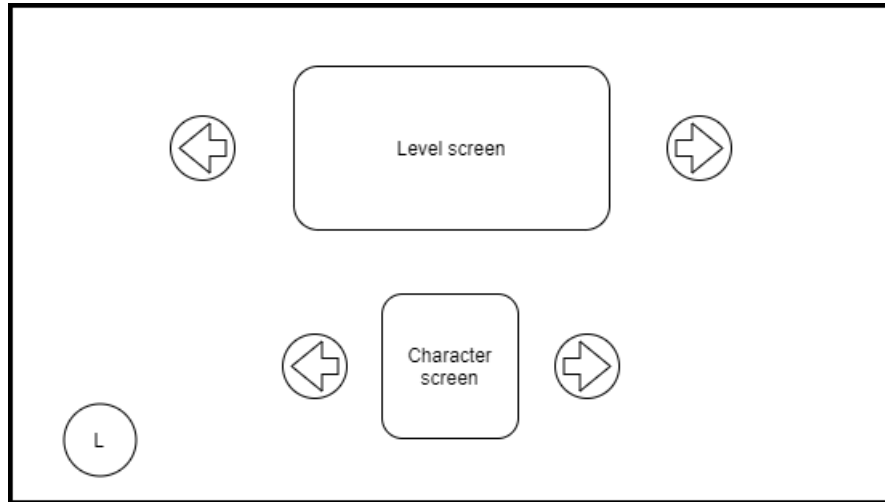


Figure 5.1: Draft of menu screen

In-game

While in a level, player can move in a traditional 2D side scrolling environment. There are buttons to move left, right and up (jump). Second button for right thumb is an action button that is used to interact with objects in the game. Last button is located in top-right corner of the screen and it serves as a “home” button - return to menu.



Figure 5.2: Draft of in-game screen

5.1.2 Final version

Several changes had to be made during development of user interface. The simplistic draft overlooked certain required features, such as more settings and not just language settings. Many other changes were initiated during testing to make UI easier to use and understand.

Menu

The “all in one” menu has been split into multiple sections. The initial window serves as a crossroad from which player can navigate to settings, character menu and level selector. Character selection would be done in the way presented in the draft, but since there are only two characters, they are simply displayed next to each other. Normal buttons change color from green to yellow on button press. In settings, the selected options keep a light blue color. Buttons in level selector, which have different colors based on their environment, all change to a light blue color on button press.



Figure 5.3: Main menu, “Play” button is pressed down



Figure 5.4: Level selection



Figure 5.5: Character selection

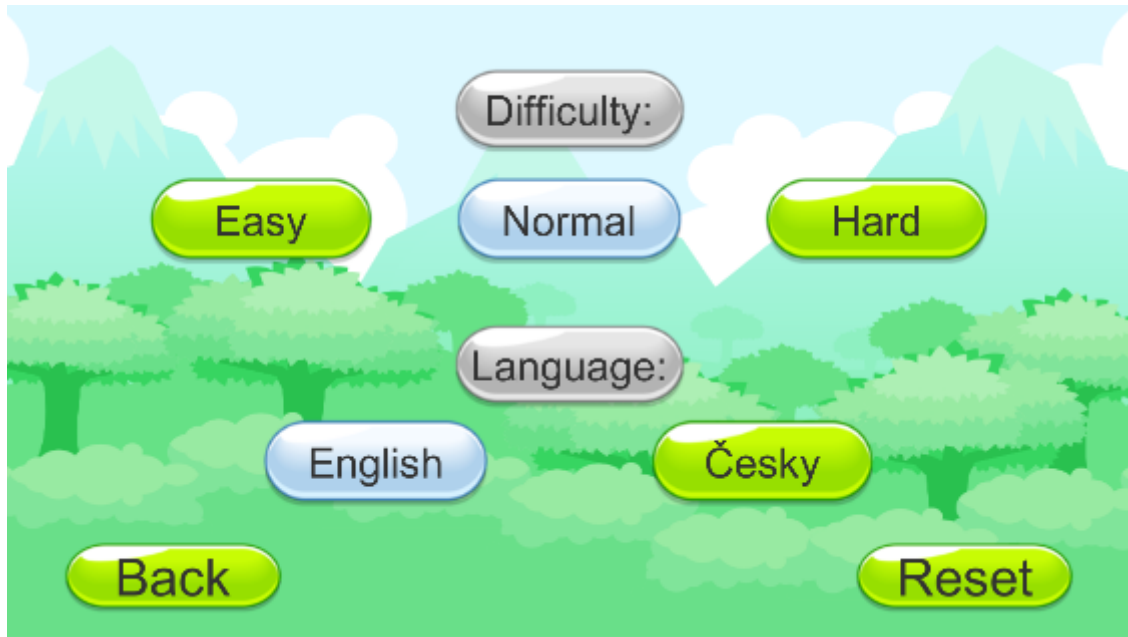


Figure 5.6: Settings

In-game

Unlike menu, the in-game UI has not changed from the draft at all. What was not mentioned in the draft is the activation device. When used, it displays a window across the whole screen with three math problems and X marks next to them indicating that the equation was not solved yet. Input field for the first equation is automatically activated. When a correct answer is entered, X mark on the corresponding line is changed to check mark, and the next input field is activated. Upon completing all of them, the window closes itself after 0.7 seconds which gives users time to observe the last check mark and therefore know that the answer was correct. Apart from the math problems, there is a button to close the activation device window, and a button to return to main menu. Those are hidden under keyboard if an input field is active. The keyboard takes a significant amount of space, especially on Android devices. That is why equations move up when the top equation is solved which guarantees that the next one will not be obstructed by the keyboard even on devices with smaller displays.

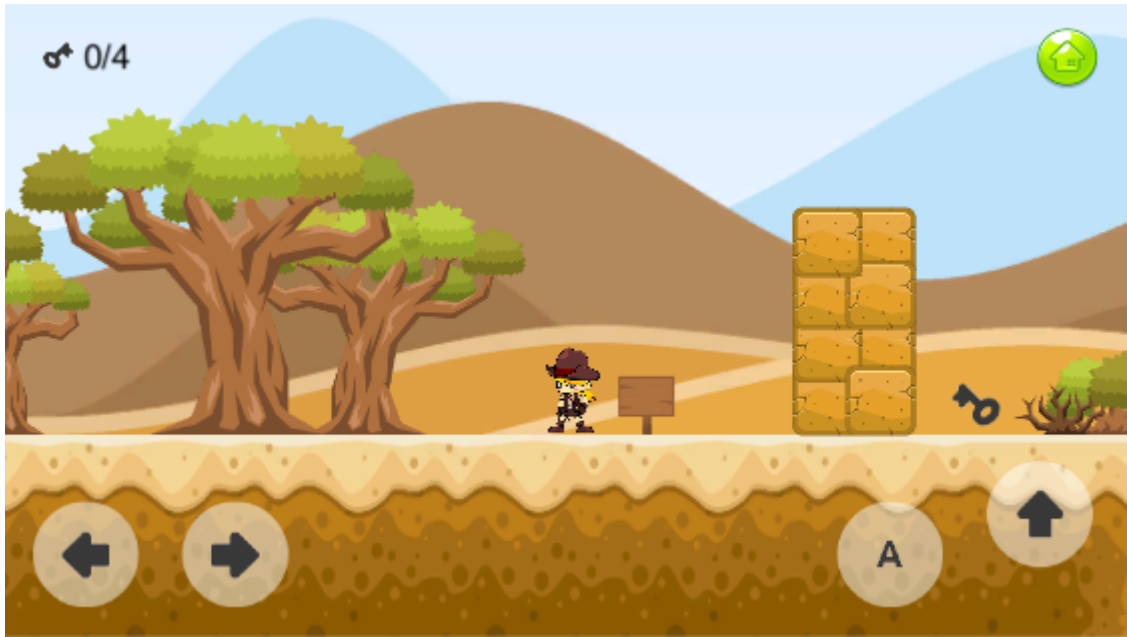


Figure 5.7: Figure demonstrating in-game UI

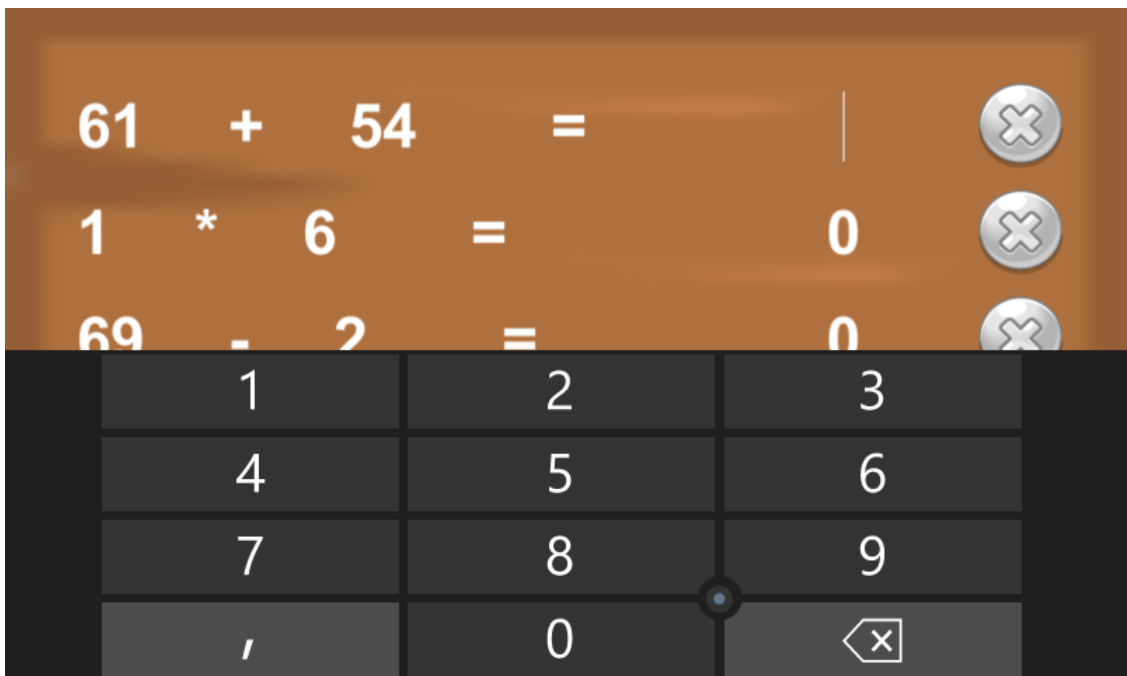


Figure 5.8: Screenshot of activation device UI with the first input field active, taken on Nokia Lumia 635

5.2 Graphic Design

As someone who has no experience in creating game art of any sort, I turned to the Internet community. There are quite a few talented artists offering part of their work for free under public domain license. It did not take long before I found the assets that I needed. Specifically mobile UI elements, a few different platformer tile sets that would go well together and some fitting characters. Absolute majority of assets that I used are made by Zuhria Alfitra and are available at www.gameart2d.com. Those include all platforms, characters and items specific to the three environments that are used in the game. The remaining assets come from www.kenney.nl and lionheart963.itch.io, some of them are also published on the Unity Asset Store.

Many artists that offer some of their work on the asset store for free often also have a personal web page, from which the assets can also be downloaded. In the end I found it more comfortable to use said sites instead of the store. Sometimes they include more assets, but the main advantage is that they can be downloaded without importing them into any project and even without having to open Unity editor. That allowed me to obtain multiple different sets of assets and later reorganize them and use just some of them without creating unnecessary mess in my project.

5.3 Animation

All animations used in the game are sprite based, that means that the animation is achieved by continuous switching between sprites in a loop. They were created using Unity's built-in tools. Of course the sprites have to be made in a way that makes this possible. Animations are assigned to a game object that is then visually represented by them. In my game, player character uses three different animations, specifically running, jumping and idle animation. Toggling between them is done by a little bit of code and an animator controller which can be managed in Unity's Animator tab. Animator controller has references to the animations and manages transitions between them using a state machine. Each character in the game has its own animations and animator controller.

5.4 Difficulty

Difficulty in the game affects the range that the numbers are generated in and also the operation type of mathematical problems. It is based on the research mentioned in chapter [Gamification and Meaningful Play](#). There are two kinds of difficulties, one affects the entire game and the other is specific to each individual level. The latter gets increasingly harder to provide steady challenge. The overall game difficulty is set by user and has much greater effect on both the range of generated numbers and the type of operations. That allows users with different age and knowledge to have challenging and satisfying experience.

5.5 Text Localization

Since I write this thesis in English, and all of my testers from the target age group do not speak English well, I decided to support multiple languages. Specifically Czech and English, but other languages could be added in the future.

Each supported language has its own JSON file. These files contain the same keys, but values specific to messages in their language. Core of localization is a LocalizationManager class that loads and parses the localized text from a file into a dictionary during launch of the application. Single instance of this dictionary is then kept in memory during runtime. Each text field that needs to be localized is assigned a parameter that matches one of the keys in the dictionary and its text is set to a value corresponding to this key. Default language is English, but it can be changed by user at any time in settings. When user changes the language, values from the file corresponding to the chosen language are loaded and replace the old ones.

This is a similar approach to the one proposed in official Unity tutorial on localization. In the tutorial, users have to choose language every time on start-up of the application which is impractical and also requires a restart if user wants to switch to a different language. Another feature of the tutorial is creating a text editor for the JSON files, which I find completely unnecessary and a waste of time.

5.6 Code Conventions

When writing code I followed common practices for Unity C# development combined with advice given in book Clean Code: A Handbook of Agile Software Craftsmanship by Robert C. Martin. Names of classes are nouns capitalized as CamelCase. Method names are verbs, and follow the same capitalization rule. Fields, variables, etc. start with a lower case letter (camelCase), on top of that booleans start with “is”. All names describe the meaning of what they represent. The goal of that is clear and expressive code that does not require extra comments to explain its meaning. Code is structured into smaller blocks and follows logical order from top to bottom. No documentation comments were used, the code on its own has no meaning. It has to be connected to the Unity project and game objects have to be binded to the appropriate variables in order to be complete.

5.7 Gameplay

The game is split into levels. Goal of each level is to retrieve certain amount of keys that is required to unlock exit and proceed to next level. Length of levels steadily increases, and with it also the number of keys that the player has to obtain. At first glance the game looks like a classical 2D platformer and to a certain extend it is. The twist comes in the form of activation devices that are visually represented as signs. These are frequently placed in all levels and they represent the educational factor of the game. When activated, the player is faced with three mathematical problems. If player successfully solves them, then there is a positive outcome. An obstacle is removed, platform starts to move, a key is made accessible, etc. Without completing these tasks, player cannot finish the level. The math problems are randomly generated and therefore different every time. If player misses a jump and falls down, he is returned to the starting position of current level. When that happens the level does not reset, already solved activation devices remain solved. The game is designed in such way, that the player cannot get stuck after falling. All keys and exit are always reachable from the starting position even after some activation devices were solved.

Following is an example of what a level could look like, see figure 5.9. There are three activation devices depicted as squares - red, yellow and blue. Player’s path is immediately blocked by an obstacle which is removed upon completion of task of the red device which

is right next to the player. Then he has to jump onto next platform and is faced with a gap that is too wide to simply jump over to the other side. Task of yellow device has to be solved and then the platform depicted with black arrows will start moving back and forth, which allows the player to get across. Finally, solving task of blue activation device will drop the key and the level can be finished by leaving through the exit.

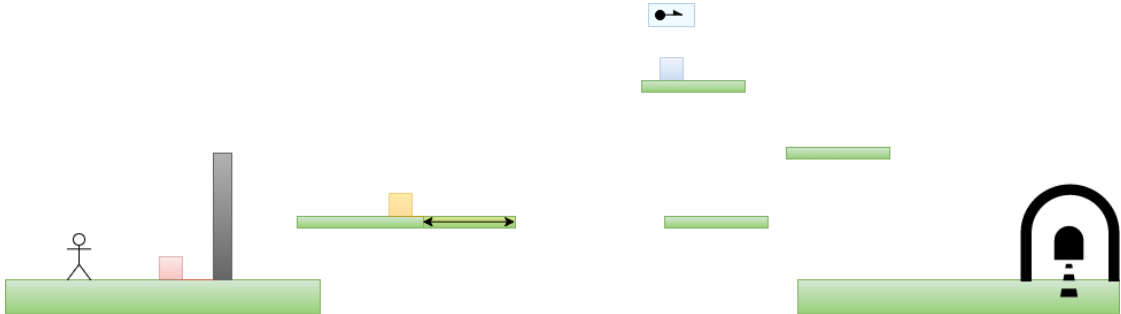


Figure 5.9: Example used to describe basic principles of the game

This example is simplified for the purpose of this paper. It is meant to only demonstrate the basic idea of the game. Actual levels are more complex, include more objects and overall look much different. The first level of the game is tutorial which explains everything that the player needs to know to play.

Chapter 6

Testing

The most basic testing to find out if the game works and to discover elementary bugs was mostly done by me, and was ongoing continuously throughout the development. Testing with other users also started as soon as possible. Since my little brother belongs to the target age group, he quickly became the main tester. Fortunately, he liked the game and did not get bored of it for a long time. He also helped me make it faster by constantly demanding new levels. The fact that all the testing equipment that I needed (cellphone) literally fits in my pocket made it easy to get to more users. I was able to use opportunities such as family visits to get more feedback. Some of my friends also wanted to try the game. That has also proven useful as those with more experience with mobile apps provided some recommendations that the kids could not even think of.

6.1 Outcome of Testing

All the gathered information was used to improve the game. Most of these changes are mentioned in this paragraph, except those that are already stated in section [User Interface](#). Originally, the character jumped on release of the jump button which in theory provides better accuracy, but it turned out to be confusing and hard to use, especially for less experienced players. So the button was changed to jump on tap rather than on release. Some users struggled with coordination of their left and right thumb which resulted in many failed jump attempts. Even though they learned and improved very fast, some small changes were made to physics. Player character now stays in the air longer and can be moved faster in order to make jumping easier. Several changes were made to the input of results of math tasks. Both the automatic selection and switching upon entering the correct result. Some users did not know the meaning of tutorial, so the text “begin here”, which can be seen in figure [5.4](#), was added next to it to help guide new players. Apart from that, many bugs have been found and fixed thanks to testing.

6.2 Devices

Cross platform application demands cross platform testing. Even though there is only one small section of code that is platform specific, there are some differences in behavior of code that is the same for all target systems. That is why I wanted to deploy and test the game on as many devices as possible. Most of the equipment that is available to me has

seen better days and did not work very well which hindered my work. Further subsections describe the testing on each specific platform.

6.2.1 Android

Unity is well equipped for testing on Android devices. No additional software is needed to deploy the application, except Android SDK and JDK of appropriate versions depending on the target device. Then, it is just a question of tweaking some settings and everything is ready. That is in theory... In reality many problems come up. Failing installations, missing files, conflicts between Android SDK Manager and Visual Studio SDK Manager are just some of the issues that I had to deal with. After sorting all of them out, everything worked fine.

Unfortunately the only Android devices I could get my hands on are worn down and fairly outdated in both software and hardware. On top of that, one of them (The Allview C6) also turns itself off spontaneously every few minutes, which made testing on it a nuisance. It was good to see that the game ran even on these old devices without any performance issues. There were some problems with input on Sony Xperia, sometimes the input window does not open automatically for a reason that remains a mystery to me.

Sony Xperia E1

- Operating system: Android 4.4.2 KitKat
- Display Size: 4 inches
- Resolution: 480 x 600
- RAM: 512 MB

Allview C6 Quad 4G

- Operating system: Android 4.4.2 KitKat
- Display Size: 4,5 inches
- Resolution: 480 x 854
- RAM: 1 GB

6.2.2 Universal Windows Platform

Unity cannot directly deploy applications to UWP devices. Instead it creates a solution for Visual Studio, which can be used for deployment just like any other solution. I was focused only on Windows 10 Mobile, but I have also ran the app on personal computer. Testing on UWP turned out the best for me. Deployment worked exactly as it should without any issues. The game also ran smoothly and without any problems on both devices that I used. Since they were also in good condition, they were the most used during user testing.

NOKIA Lumia 635

- Operating system: Windows 10 Mobile

- Display Size: 4,5 inches
- Resolution: 480 x 854
- RAM: 1 GB

NOKIA Lumia 640 LTE

- Operating system: Windows 10 Mobile
- Display Size: 5 inches
- Resolution: 720 x 1280
- RAM: 1 GB

6.2.3 iOS

In order to deploy Unity made application to an iOS device, one must generate Xcode project from Unity and then deploy it like any other Xcode project. This means that it is necessary to have Apple iMac with Xcode. I do not own such device so I planned to use Macs that are available for students at my university. Unfortunately the operating system of those devices is outdated (version 10.11.6), which is below the minimum requirements (version 10.13.4 or newer[21]) for Xcode. The hardware is also old and does not allow upgrade of the operating system. After finding that out, I decided to use a virtual machine to deploy my game. I successfully installed the virtual machine with current version of macOS and managed to open my project with Xcode. Unfortunately I could never get it to recognize my phone and so I was never able to deploy the game to an iOS device even though I had one ready.

iPhone SE

- Operating system: iOS 11.3
- Display Size: 4 inches
- Resolution: 640 x 1136
- RAM: 2 GB

Chapter 7

Further Development

Just like with any product, there is still a lot that could be done to improve the game. The most significant improvement, in my opinion, is addition of sound. Right now the game is lacking any sort of sound or music, because I was not able to find a place which would offer suitable sound for my game for free. Many people nowadays play mobile games in places where they have to have sound off, such as in public transport, waiting rooms or during school lessons, which makes it somewhat less crucial. But I believe that sound is an important part of any game and should be present if possible.

The most obvious possibility for further development is to add more characters, levels, contraptions and environments. In this regard, the development process could go on forever. Another feature that could be added are translations into more languages, since implementation of localization is done in such way that makes it very easy. Then there are some small changes that would make sense only in case the game would be published to a store. Amongst the most significant of those would be locking of levels and unlocking them as the player progresses, and also addition of trophies or achievements.

When it comes to game mechanics, there is one thing that I would like to change. Moving platforms require player to move along with them, otherwise the platform slips from under player's feet and makes them fall. This should be changed to make the player move with the platform automatically. Even though my main tester is against this change, I believe it would improve overall game experience.

While coming up with ideas for this game, I was playing with the idea that activation devices (signs) would incorporate more than just mathematical problems, but also some logic and puzzle games. Implementing whole mini-games within the main game turned out to be too complicated and too time demanding for me to manage during school year. But it is something that I can keep in mind and maybe add later.

Chapter 8

Conclusion

As the name of this work indicates, the goal was to create an educational cross platform mobile game. At the beginning, I had only a rather vague idea of the game that I wanted to make, and nearly no clue how to do it. All of it became very clear during my work on theoretical part of the thesis. First, I studied theory of games and what makes them engaging and fun to play. That was followed up by research of the most significant mobile platforms and games that are available on them, with focus on educational ones. I looked into different ways of making such games, analyzed my options and decided to use Unity game engine.

The final product meets said requirements. It is a combination of a 2D side scrolling platformer and a mathematical game. It has a total of fifteen levels, one of them being tutorial which teaches players all they need to know to play. The entire game fully supports two languages - English and Czech. Player has to find a certain amount of keys and locate exit in each level to finish it. It is necessary to solve randomly generated mathematical problems along the way in order to succeed. Difficulty of those tasks can be adjusted in the menu, but it also gets increasingly harder as the player progresses through levels.

Testing played a major role during development and resulted in several changes and improvements. It was conducted on two Android and two Windows Phone devices with over twenty users. Not all of them belong to the target audience group, which turned out to be a good thing, as experienced users were able to come up with ideas that the children could not. Developing applications for mobile platforms and testing them requires a lot of different devices and software, even when the application itself is cross-platform and done using a single program. With that necessarily come complications. I had to spend a fair bit of time solving problems that should not have appeared in the first place.

One does not have to pay much attention to make the observation that cellphone industry and everything revolving around it are going through a phase of incredibly fast development. Number of mobile users keeps growing and the trend will most likely not stop for some time. All of this and other factors contribute to the astonishing speed of technological progress in this field. Many things have changed significantly during the time that I was working on this thesis. The most notable being the official end of Windows 10 Mobile development. For many, this was not a surprise as the platform failed to attract enough app developers, but it was a hit for those that decided to devote their time and resources to make apps for it. This shows how important it is for developers to be versatile and not focus on just one platform as it might not be around when their release is ready.

Bibliography

- [1] Koivisto, E. M. I.: *Mobile Games 2010*. CyberGames '06. Murdoch University. 2006. ISBN 86905-901-7.
- [2] Hirsh-Pasek, K.; Zosh, J. M.; Golinkoff, R. M.; et al.: *Putting Education in "Educational" Apps: Lessons From the Science of Learning*. *Psychological Science in the Public Interest*. vol. 16. 2015.
- [3] [Online; visited 10.1.2018].
Retrieved from: <https://medium.com/@c0ntinuity/teaching-through-voluntary-play-87a91c1bf680>
- [4] Malone, T.: *What Makes Things Fun to Learn?: A Study of Intrinsically Motivating Computer Games*. Cognitive and instructional sciences series; CIS-7. Xerox, Palo Alto Research Center. 1980.
- [5] Khaddage, F.; Lattemann, C.; Acosta-Diaz, R.: *Mobile Gamification in Education Engage, Educate and Entertain via Gamified Mobile Apps*. Association for the Advancement of Computing in Education (AACE). 2014.
- [6] [Online; visited 5.5.2018].
Retrieved from: <https://www.h-mat.cz/en/>
- [7] Mach, J.: *Nová matematika? Nechápou ji rodiče ani učitelé*. [Online; visited 5.5.2018].
Retrieved from: <https://www.novinky.cz/domaci/470989-nova-matematika-nechapou-ji-rodice-ani-ucitele.html>
- [8] [Online; visited 20.12.2017].
Retrieved from: <https://en.oxforddictionaries.com/definition/us/game>
- [9] Radoff, J.: *History of Social Games*. [Online; visited 15.12.2017].
Retrieved from: <https://web.archive.org/web/20150118061155/http://radoff.com:80/blog/2010/05/24/history-social-games/>
- [10] Peterson, J.: *2016 Lecture 02 Maps of Meaning: Playable and non-playable games*. 2016. [Online; visited 27.12.2017].
Retrieved from: <https://youtu.be/RcmWssTLFv0>
- [11] Huizinga, J.: *Homo Ludens*. International Library of Sociology Series. Routledge. 2003. ISBN 9780415175944.
- [12] Salen, E., K.; Zimmerman: *Rules of Play: Game Design Fundamentals*. The MIT Press. 2003. ISBN 0262240459, 9780262240451.

- [13] [Online; visited 13.1.2018].
Retrieved from: <https://www.lifewire.com/cheating-in-online-games-1983529>
- [14] [Online; visited 27.1.2018].
Retrieved from: <https://www.merriam-webster.com/dictionary/gamification>
- [15] Van Eck, R.: *Digital game-based learning: Still restless, after all these years.* *EDUCAUSE Review*. vol. 50. 2015.
- [16] Deterding, S.: *Meaningful Play: Getting Gamification Right.* GoogleTechTalks. Jan 2011. [Online; visited 8.11.2017].
Retrieved from: <https://youtu.be/7ZGCPap7GkY>
- [17] Levy, F.; Murnane, R. J.: *The New Division of Labor: How Computers Are Creating the Next Job Market.* Princeton University Press. 2004. ISBN 9780691124025.
- [18] Klopfer, E.: *Augmented Learning: Research and Design of Mobile Educational Games.* Augmented Learning. MIT Press. 2011. ISBN 9780262516525.
- [19] [Online; visited 1.5.2018].
Retrieved from: <https://www.ftc.gov/news-events/press-releases/2016/01/lumosity-pay-2-million-settle-ftc-deceptive-advertising-charges>
- [20] [Online; visited 14.1.2018].
Retrieved from: <https://www.statista.com>
- [21] [Online; visited 1.4.2018].
Retrieved from: <https://developer.apple.com/xcode/ide/>
- [22] [Online; visited 31.1.2018].
Retrieved from: <https://msdn.microsoft.com/en-us/library/dn975273.aspx>
- [23] Rouse, M.: [Online; visited 27.1.2018].
Retrieved from: <http://searchsoftwarequality.techtarget.com/definition/hybrid-application-hybrid-app>
- [24] Bristowe, J.: *What is a Hybrid Mobile App.* [Online; visited 27.1.2018].
Retrieved from:
<https://developer.telerik.com/featured/what-is-a-hybrid-mobile-app/>
- [25] [Online; visited 27.1.2018].
Retrieved from: <https://hackernoon.com/how-much-time-do-people-spend-on-their-mobile-phones-in-2017-e5f90a0b10a6>
- [26] [Online; visited 14.1.2018].
Retrieved from: <https://newzoo.com/insights/articles/the-global-games-market-will-reach-108-9-billion-in-2017-with-mobile-taking-42/>
- [27] [Online; visited 14.1.2018].
Retrieved from: <https://artjoker.net/blog/mobile-game-development-in-2017-best-tools-and-advice/>
- [28] [Online; visited 25.12.2017].
Retrieved from: <https://store.unity.com>

- [29] [Online; visited 16.1.2018].
Retrieved from: <https://www.unrealengine.com/en-US/what-is-unreal-engine-4>
- [30] [Online; visited 15.1.2018].
Retrieved from: <https://docs.yoyogames.com/>
- [31] [Online; visited 5.5.2018].
Retrieved from: <https://www.yoyogames.com/>
- [32] [Online; visited 20.12.2017].
Retrieved from: <https://github.com/photonstorm/phaser>