

**Czech University of Life Sciences Prague**

**Faculty of Economics and Management**

**Department of Information Technologies**



**Bachelor Thesis**

**Mobile Application Development**

**Volodymyr Pukha**

© 2022 ČZU Prague



## BACHELOR THESIS ASSIGNMENT

Volodymyr Pukha

Informatics

Thesis title

**Mobile Application Development**

---

### Objectives of thesis

The main goal of this work is to analyze and select preferred tools and environments to develop a mobile application and choose optimal platform for its release based on selected requirements and criteria.

Partial goals are:

- Study the literature and gather information from professional information sources.
- Compare different approaches to mobile application development and publishing by evaluating and comparing suitable options.
- Develop and launch a prototype application.

### Methodology

The theoretical part of the work is based on the study and analysis of professional and scientific information sources. Different platforms for mobile application release will be analyzed and compared using multiple-criteria decision analysis based on chosen criteria.

An experimental prototype application will be developed and launched utilizing the most suitable variant. Based on the synthesis of knowledge of the theoretical part and evaluation of the results of the practical part, the conclusions of the work will be formulated.

## The proposed extent of the thesis

40-50

## Keywords

Mobile applications, Android, iOS, Unity, MCDA, Play Market, App Store

---

## Recommended information sources

Buttfield–Addison, Paris; Manning, Jonathon; Nugent, Tim. Unity Game Development Cookbook: Essentials for Every Game. O'Reilly; 1st edition, 2019. ISBN 978-1491999158

Lewis, Shaun; Dunn, Mike. Native Mobile Development. O'Reilly Media, Inc.; 1st edition, 2019. ISBN 9781492052876

McWherter, Jeff; Gowel, Scott. Professional Mobile Application Development. Wrox; 1st edition, 2012. ISBN 978-1118203903

Mukherjea, Sougata. Mobile Application Development, Usability, and Security. IGI Global; 1st edition, 2016. ISBN 978-1522509455

---

## Expected date of thesis defence

2021/22 SS – FEM

## The Bachelor Thesis Supervisor

Ing. Jan Pavlík

## Supervising department

Department of Information Technologies

Electronic approval: 16. 8. 2021

**doc. Ing. Jiří Vaněk, Ph.D.**

Head of department

Electronic approval: 5. 10. 2021

**Ing. Martin Pelikán, Ph.D.**

Dean

Prague on 13. 03. 2022

### **Declaration**

I declare that I have worked on my bachelor thesis titled "Mobile Application Development" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the bachelor thesis, I declare that the thesis does not break any copyrights.

In Prague on 14.03.2022

---

## **Acknowledgement**

I would like to thank Ing. Jan Pavlík for his advice and support during my work on this thesis.

# Mobile Application Development

## Abstract

The theoretical part of the thesis focuses on several topics. The first one is a description and comparing of mobile application categories, including the difference in their development approaches, costs, maintenance, other important criteria. The second one is development tools description and comparison. The third one covers mobile application digital distributions platforms, mainly consists of comparative platforms' analysis. The last topic is focused on application development lifecycle. It considers important steps and factors which are considered in the practical part of the thesis. The practical part of the thesis consists of three main stages. The first one covers multiple-criteria decision analysis which defines which development tool and release platform are preferred choices for developing and releasing mobile application according to selected requirements and criteria. The second part focuses on mobile application development using selected development tool. The third part covers developed mobile application release on the selected digital distribution platform.

**Keywords:** Mobile applications, Android, iOS, Unity, MCDA, Play Market, App Store

# Vývoj mobilních aplikací

## Abstrakt

Teoretická část práce se zaměřuje na několik témat. Prvním z nich je popis a porovnání kategorií mobilních aplikací, včetně rozdílů v přístupech k jejich vývoji, nákladech, údržbě a dalších důležitých kritériích. Druhým je popis a porovnání vývojových nástrojů. Třetí se zabývá platformami pro digitální distribuci mobilních aplikací, spočívá především ve srovnávací analýze platforem. Poslední téma je zaměřeno na životní cyklus vývoje aplikace. Zohledňuje důležité kroky a faktory, které jsou uvažovány v praktické části práce. Praktická část práce se skládá ze tří hlavních etap. První z nich zahrnuje vícekritériální rozhodovací analýzu, která definuje, který vývojový nástroj a platforma pro vydání jsou preferovanou volbou pro vývoj a vydání mobilní aplikace podle zvolených požadavků a kritérií. Druhá část se zaměřuje na vývoj mobilní aplikace pomocí vybraného vývojového nástroje. Třetí část se zabývá vydáním vyvinuté mobilní aplikace na vybrané digitální distribuční platformě.

**Klíčová slova:** Mobilní aplikace, Android, iOS, Unity, MCDA, Play Market, App Store



# Table of content

<b>1</b>	<b>Introduction</b>	<b>13</b>
<b>2</b>	<b>Objectives and Methodology</b>	<b>14</b>
2.1	Objectives	14
2.2	Methodology	14
<b>3</b>	<b>Literature Review</b>	<b>15</b>
3.1	Mobile application categories	15
3.2	Mobile application types	16
3.2.1	Native mobile applications	16
3.2.2	Web mobile applications	17
3.2.3	Hybrid mobile applications	17
3.2.4	Native, Web and Hybrid applications development criteria comparison	18
3.2.5	Cross-platform development	19
3.3	Development tools	20
3.3.1	Unity	20
3.3.2	Unreal Engine	20
3.3.3	GameMaker: studio	21
3.3.4	Godot	21
3.4	Mobile operating systems	22
3.5	Mobile applications digital distribution platforms	23
3.5.1	Google Play Store	24
3.5.2	App store	24
3.5.3	Comparison Google Play vs. App Store	25
3.6	Application development lifecycle	27
3.6.1	Mobile application development process for iOS and Android	27
3.7	Development requirements and criteria	30
3.7.1	Expediency	30
3.7.2	Cost of development	31
3.7.2.1	Hardware cost	31
3.7.2.2	Developer Accounts	32
3.7.2.3	Licenses	32
3.8	Summary	32
<b>4</b>	<b>Practical Part</b>	<b>34</b>
4.1	Multiple-criteria decision analysis	34
4.1.1	Scenario	34
4.1.2	MCDA of development tools	34

4.1.2.1	Assigning weights chosen criteria for development tools.....	34
4.1.2.2	Defining set of alternatives for development tools.....	36
4.1.2.3	Assigning points for alternatives criteria for development tools.....	36
4.1.2.4	Result of development tools' MCDA.....	40
4.1.3	MCDA of release platforms.....	40
4.1.3.1	Assigning weights chosen criteria for release platforms.....	40
4.1.3.2	Defining set of alternatives for development tools.....	41
4.1.3.3	Assigning points for alternatives criteria for release platforms.....	42
4.1.3.4	Result of release platforms' MCDA.....	45
4.2	Prototype application development.....	45
4.2.1	Hardware specifications.....	45
4.2.2	Unity installation.....	46
4.2.3	Mobile game prototype development.....	49
4.2.3.1	Game idea.....	49
4.2.3.2	Game development.....	49
4.2.3.3	Building game.....	54
4.3	Application release.....	55
4.3.1	Creating Google Play developer account.....	55
4.3.2	Creating an application.....	55
<b>5</b>	<b>Results and Discussion.....</b>	<b>59</b>
<b>6</b>	<b>Conclusion.....</b>	<b>60</b>
<b>7</b>	<b>References.....</b>	<b>61</b>

## List of pictures

Figure 1 - App Store Revenue Share by Category.....	16
Figure 2 - Market share of mobile operating systems in Czech Republic.....	22
Figure 3 - App Store vs Google Play Store page layouts.....	23
Figure 4 - Number of applications available in leading app stores.....	25
Figure 5 - Software Development Life cycle principles.....	28
Figure 6 - Mobile Internet: average daily time spent in the US, App vs Browser.....	31
Figure 7 – Unity plans and pricing.....	46
Figure 8 – Unity Editor official releases.....	47
Figure 9 – Unity Editor modules.....	47
Figure 10 – Unity project creation templates.....	48

Figure 11 – Unity project build settings .....	48
Figure 12 – Project scenes .....	49
Figure 13 – Joystick Pack asset .....	49
Figure 14 – GameManager script .....	50
Figure 15 – MainGame scene hierarchy .....	51
Figure 16 – Game development environment.....	51
Figure 17 – Tiles Drop main menu .....	52
Figure 18 – Tiles Drop levels screen .....	52
Figure 19 – Tiles Drop gameplay .....	53
Figure 20 – Tiles Drop win screen.....	53
Figure 21 – Project player build settings .....	54
Figure 22 – Project build options.....	54
Figure 23 – Creating an application on Google Play .....	55
Figure 24 – Tasks for setting up the app.....	56
Figure 25 – App category and tags .....	57
Figure 26 – Application test countries and release creation .....	57
Figure 27 – Production release review state .....	58
Figure 28 – Tiles Drop page in the Google Play Store .....	58

## List of tables

Table 1 - Mobile platforms developer accounts conditions comparison.....	32
Table 2 - Game engines license prices comparison.....	32
Table 3 – Development tools criteria weights .....	35
Table 4 – Development tools alternatives criteria parameters.....	38
Table 5 – Development tools criteria points .....	38
Table 6 – Development tools criteria points transformation .....	39
Table 7 – Development tools ideal and nadir alternatives.....	39
Table 8 – Development tools criterial matrix R .....	39
Table 9 – Development tools’ MCDA result.....	40
Table 10 – Release platforms criteria weights.....	41
Table 11 – Release platforms alternatives criteria parameters .....	44
Table 12 – Release platforms criteria points .....	44
Table 13 – Release platforms criteria points transformation .....	44

Table 14 – Release platforms ideal and nadir alternatives .....	44
Table 15 – Release platforms criterial matrix R .....	45
Table 16 - Release platforms' MCDA result .....	45

# 1 Introduction

The popularity of mobile applications grows with the growth of the smartphone's owners. Nowadays mobile applications are an essential part of day-to-day human's life. The key of the success is accessibility. People can access their mobile applications anywhere where they can access their smartphones: at home, at work, on the street, in bed, while eating etc. People now launch an average of at least 9 apps per day, 30 apps per month. Global consumers are now spending an average of 4.2 hours per day using apps on their smartphones (Sarah Perez, 2021). As a result, mobile applications market expands and develops quickly. App usage is still growing at a steady rate. Mobile apps are expected to generate \$469,198 million revenue worldwide in 2022 (Statista Research Department, 2021). Around 5 million applications are available on the Apple App Store and Google Play Store.

Due to the mobile application industry growth a lot of development tools and several app distribution platforms were created. Which development tool to use to develop a mobile application depends on the kind of an application and on which release platform it is going to be released. There are a lot of factors that impacts on the decision which options to choose while developing mobile application, such as hardware cost, license price, tool complexity etc. The thesis is focused on the defining the best solution for the choice decision questions that occur during application development lifecycle. By analyzing selected criteria and preferences it is possible to find best alternative using Multiple-criteria decision analysis (MCDA). Such method structures and solves decision and planning problems involving multiple criteria. It was firstly proposed by Bernard Roy in 1968. Specifically Simple additive method was used in the practical part of the thesis. This method is the best known and simplest MCDA method.

It was decided to develop and release mobile game, as the mobile games category generates the main part of the mobile applications' revenue. Whole logic of the mobile game developed was written in the C# programming language.

## **2 Objectives and Methodology**

### **2.1 Objectives**

The main goal of this work is to analyse and select preferred tools and environments to develop a mobile application and choose optimal platform for its release based on selected requirements and criteria.

Partial goals are:

- Study the literature and gather information from professional information sources.
- Compare different approaches to mobile application development and publishing by evaluating and comparing suitable options.
- Develop and launch a prototype application.

### **2.2 Methodology**

The theoretical part of the work is based on the study and analysis of professional and scientific information sources. Different platforms for mobile application release will be analyzed and compared using multiple-criteria decision analysis based on chosen criteria.

An experimental prototype application will be developed and launched utilizing the most suitable variant. Based on the synthesis of knowledge of the theoretical part and evaluation of the results of the practical part, the conclusions of the work will be formulated.

## 3 Literature Review

### 3.1 Mobile application categories

Currently, there are a lot of different categories of mobile applications available on various mobile markets. Among them it is possible to pick out 6 main types of mobile applications.

Common categories of mobile applications according to (Bridget Poetker, 2019):

- **Educational applications**

The purpose of an educational application is to educate and inform. It can be news application or application for learning language.

- **Lifestyle applications**

The purpose of a lifestyle application is simply to provide you information how to spend your leisure time with pleasure.

- **Social media applications**

This type of applications gives you the ability to connect with people and expand your social circle. These applications used to post images, text posts, share videos, facilitate conversations and more.

- **Productivity applications**

This type of applications also known as business applications. They are developed to organize complex tasks for you. The purpose of this applications is to increase your productivity.

- **Entertainment applications**

The purpose of entertainment applications is to fill your time. Usually they include text, audio or video content.

- **Game applications**

Game applications are the most popular type of mobile applications. Moreover, mobile games' consumer spending rate is the highest among all other types of mobile applications. In addition, game applications are the most profitable ones. Mobile gaming industry is growing with enormous rates. In 2018 mobile gaming accounted for more than half of all gaming industry revenue (Mediakix, 2020). One of the main reason mobile games are so popular is because they are easily accessible. People often spend times playing mobile

games on their way to the work or study place. The reason is simply to shorten time and get some positive emotions.

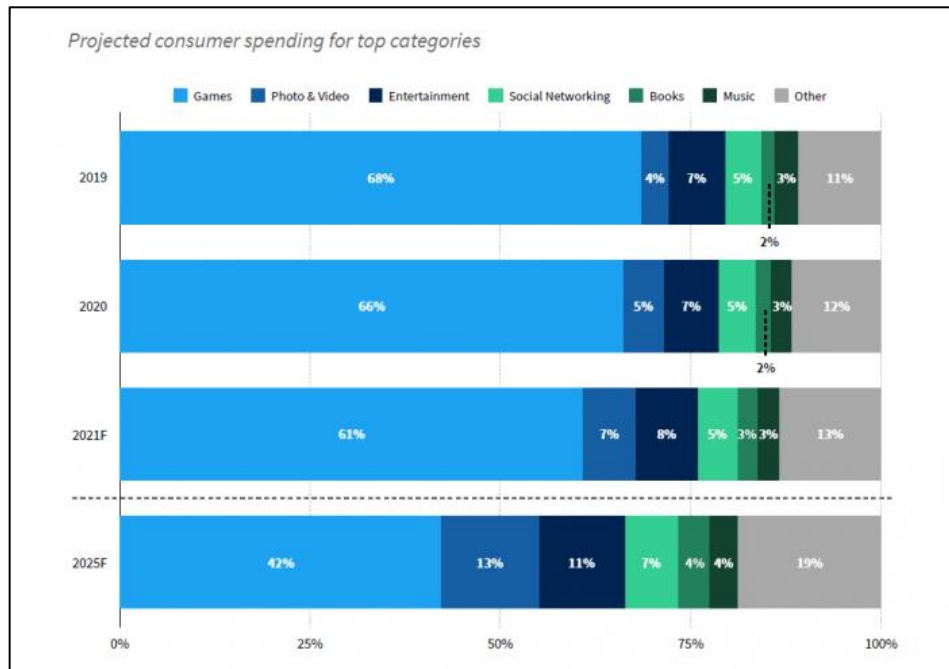


Figure 1 - App Store Revenue Share by Category. Source: [blog.udonis.co](http://blog.udonis.co)

## 3.2 Mobile application types

As previously mentioned, the demand for the mobile applications development increases every year. Through the design and development process in building applications, software developers tend to have three different approaches to develop an application according to (Lionel Valdellon, 2020):

- Native mobile applications
- Mobile web applications
- Hybrid mobile applications

### 3.2.1 Native mobile applications

Native application is a software application built in a specific programming language for a specific device platform, for iOS or Android. Native applications for Android are written in Java. Native iOS applications are written in Swift or Objective-C. Apple and Google provide developers their own development tools and standardized Software Development Kit. The main advantage of the native mobile application is that it is located locally on the device and as a result it is faster and more responsive to users. Moreover,



native applications have access to device's camera, microphone, GPS, accelerometer, swipe gestures, notifications etc.

Disadvantages of the native applications is that they are not cross-platform, and the cost of the native application development is higher than the cost of development of the hybrid applications.

### **3.2.2 Web mobile applications**

Web mobile applications are not real applications. In fact, it is websites that look and feel like native applications but are not implemented as such. As a result, web applications do not need to be downloaded like native or hybrid applications. Web applications load in browsers like Chrome, Safari or Firefox and they do not take up any memory or storage on the user's device.

The vast majority of Web application development is done using JavaScript, CSS, and HTML5. Unlike native iOS or Android application, there is no Software Development Kit for a developer to work with. There are templates and frameworks for developing web applications like Angular, React, and Vue.js that can be used to get a quick start.

### **3.2.3 Hybrid mobile applications**

This kind of applications is the most preferable choice among many companies. Hybrid applications are partially native applications and partially web applications at the same time (Mukherjea Sougata, 2016). They can be downloaded from the App Store or Google Play Market. As advantage hybrid mobile applications are cross-platform and have access to the same device features as native mobile applications have. Comparing to native applications, hybrid applications development cost is much lower. This is caused mainly by cross-platform development. All hybrid mobile applications are developed through the HTML5 programming language.

### 3.2.4 Native, Web and Hybrid applications development criteria comparison

Each of these types of apps has their advantages and disadvantages:

- **Speed**

Native apps win the speed competition. Responsiveness is a key to usability.

- **Maintenance**

Maintaining a web or a hybrid application is as simple as maintaining a web page. Maintaining a native application can be complicated not only for users, but also for developers. Changes in native application version must be packaged and updated on the mobile distribution platform. For such update some additional review from the App Store or Google Play Market is needed, what takes additional time.

- **Development cost**

It is cheaper to develop hybrid and web applications. Native application development would demand a lot of time, effort, costs, and resources unlike the hybrid mobile apps with shared backend code or cross-platform app that has a reusable codebase (Satinder Singh, 2021).

- **Installation**

For average user some motivation needed to install a native or a hybrid application. Installing a web application is actually just creating a bookmark from the browser on the home screen. In fact, people do not use bookmarks that much on mobile, so it is less familiar to users, despite it is simpler than downloading a new application from the store.

- **Offline functionality**

Native application is best if your application supposed to work when there is no internet connectivity. In-browser caching is available in HTML5, but it is still more limited than what you can get when you go native.

- **Device features**

Although web applications s can take advantage of some features, native applications (and the native components of the hybrid applications) have access to the full paraphernalia of device-specific features, including GPS, camera, gestures, and notifications.

- **Platform independence**

While different browsers may support different versions of HTML5, if platform independence is important, then it is definitely a better idea of achieving it with web and hybrid applications than with native applications.

- **Approval process, content restrictions and fees**

Dealing with a third party that imposes rules on the content and design can be taxing both in terms of time and money. Native and hybrid applications must pass approval processes and content restrictions imposed by application stores, whereas the web is free for all.

### 3.2.5 Cross-platform development

One outstanding development technique is cross-platform development. The developed application is considered to be native mobile application type but can be deployed to multiple supported platforms without rewriting code. For example, a developer can create a game in Unity Editor using C# programming language, but deploy the application to more than 20 platforms, such as iOS, Android, PS5, Xbox One etc.

The benefits of cross-platform development (Anastasiya Marchuk, 2021):

- **Cost reduction**

There is no need to hire specialists of different skillsets and programming languages as the same code base can be used to build application for different platforms

- **Faster development**

As the same code base used to develop an application it significantly decreases the time needed to deploy an application to different platforms

- **Maintenance**

Only one app to focus on for different platforms makes it much easier to maintain and update it.

Among the disadvantages of cross-platform development the main one is application speed. Cross-platform app needs an additional rendering process and abstraction layer. As a result, comparing to the true native application - cross-platform app is slower. Moreover,

developers tend to have difficulties to access some of the smartphone parts, such as camera, microphone, geolocation etc.

To conclude cross-platform development approach nowadays is the most beneficial and preferred way to develop mobile games. And to use this approach game engines comes to be extremely helpful.

### **3.3 Development tools**

A game engine delivers the main functionality of the game. It is a framework that provides the tools and the structure that a game requires. Currently, there are available various game engines for mobile games development. Among them 2 the most popular engines are Unreal Engine and Unity. Which engine to choose comes down to personal preferences and intended purposes.

#### **3.3.1 Unity**

Unity is a platform created by Unity Technologies for creating and operating interactive, real-time 3D content. In 2020 71% of the top 1,000 mobile game were made with Unity (Unity, 2021).

##### **Advantages:**

- Unity supports 27 platforms
- Supports coding in C# and JavaScript
- 2D and 3D support
- Powerful graphical engine

##### **Pricing:**

- Free for Students and Beginner Startups
- \$40 / Month for Unity Plus Edition
- \$150 / Month for Unity Pro Edition
- \$4000 / Month for Unity Enterprise Addition

#### **3.3.2 Unreal Engine**

Unreal Engine is a complete suite of development tools for anyone working with real-time technology developed by Epic Games. Written in C++, the Unreal Engine features

a high degree of portability, supporting a wide range of mobile, desktop, console and virtual reality platforms.

**Advantages:**

- Graphical capabilities are far ahead of the competition
- Free and open-source software
- More tools and functionalities for a wide variety of situations

**Pricing:**

- Free for Beginners and startups
- Pay 5% royalty, if gross revenue would be \$1,000,000 or more.

### **3.3.3 GameMaker: studio**

GameMaker: studio is famous because of its no-code development approach. It doesn't require programming knowledge to use. But the problem with GameMaker and other point-and-click engines is that developers are much more limited than with other engines (Renana Dar, 2021).

**Advantages:**

- Primary strength in making 2D games
- Beginners friendly, drag-and-drop creation tool
- Good debugging functionality

**Pricing (CZ region):**

- \$30 annually for Creator version with Desktop platforms Exports
- \$60 annually for Indie version with Mobile platforms exports

### **3.3.4 Godot**

Godot is free-to-use open-source through the MIT license engine. This engine is great for both 2D and 3D games development.

**Advantages:**

- Easy to use
- Beginners friendly
- Active community support

**Pricing:**

- Absolutely free

There is still a great amount of available and powerful game engines and each of them has its own advantages and disadvantages. To decide which game engine to use it is needed to analyze a lot of factors. such as how big is a project, which feature the project requires, what platforms are the main target, how big is a budget of the project. For example, Unreal Engine seems to be a better option for games needing advanced graphical performance and for big companies with experienced development teams, while Unity seems to be a better option for a standalone beginner developer or a small company.

### 3.4 Mobile operating systems

Android and iOS are 2 the most popular mobile operating system used nowadays. For each of them exist many various types of development tools and environments. The main difference while developing an application for these platforms are different design principles and programming languages. Moreover, Android developers tend to adapt application design for a huge number of devices and display sizes, while iOS developers have less diversity of supported devices. One of the top priorities is to develop convenient and functional user-friendly application. Android and iOS provide a set of frameworks, along with a set of patterns, in order to achieve this goal (Lewis Shaun; Dunn Mike, 2019).

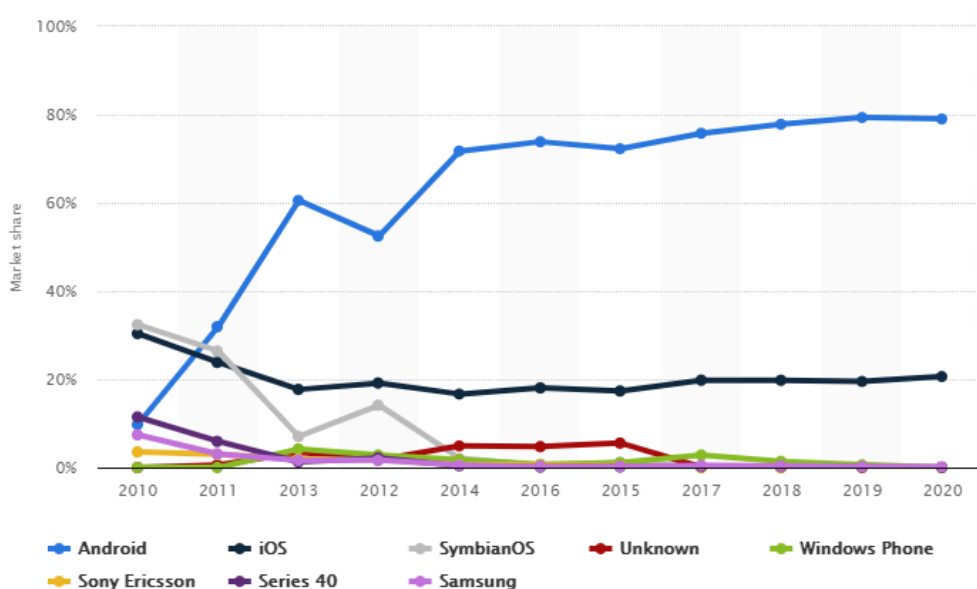


Figure 2 - Market share of mobile operating systems in Czech Republic from 2010 to 2020.

Source: statista.com

### 3.5 Mobile applications digital distribution platforms

Mobile application digital distribution platforms are made with the purpose of searching, downloading, and installing applications. Two the most used platforms nowadays are Google Play Store and App Store. On both of them application page has the same elements:

- text description
- icon or logo
- gallery of screenshots
- user ratings and reviews
- an introductory video or trailer (mostly for popular, frequently downloaded applications).

But this does not mean that they are the same. In fact, they have many factors that makes them quite different. Apple's App Store has 2.2 million applications and games, while Google Play Store has 3.5 million as of 2021 (L. Ceci, 2021).

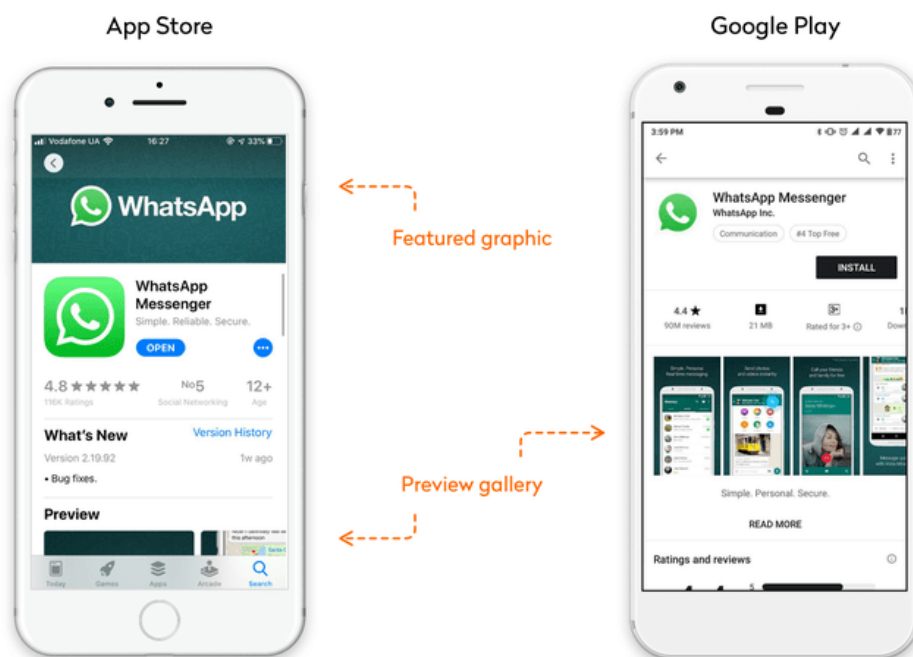


Figure 3 - App Store vs Google Play Store page layouts. Source: yalantis.com

### **3.5.1 Google Play Store**

Google play Store is a digital distribution service operated and developed by Google. It was developed and launched in 2008 with intention to distribute applications and other digital media for Android devices. By 2009, it contained only 2,300 apps, but a year later, 80,000 apps were available, and the total number of downloads exceeded 1 billion. Annual Google Play Store applications revenue reached \$38.6 billion in 2020 (Mansoor Iqbal, 2022). Google provides Google Play Pass which allows you to download hundreds of apps and games for free, without ads or iAPs (in-app purchases). It costs \$4.99 per month (or \$29.99 per year). Google Play Store comes pre-installed on Android devices.

### **3.5.2 App store**

App Store is an apps distribution platform operated and developed by Apple Inc. The App Store was opened in 2008, with an initial 500 applications available, but in 3 months their number increased to 3,000, and downloads reached 100 million. In October 2007 Apple announced an SDK that allowed developers to create native iPhone applications. Many argue that Apple's decision to allow developers to create native applications was based on the fact that the Android platform was going to be hitting the market soon.

Apple maintains strict design standards, which are detailed and updated online. IOS interface documentation and general mobile design strategies are available from Apple, including design strategies and case studies (McWherter Jeff; Gowel Scott, 2012). Apple can reject an application from the official App Store because of design problems.

Some of the Apple's current guidelines:

- iPhone users generally hold from the bottom of the device, so main navigation items should be in reach of user thumbs.
- Target areas for controls should be a minimum of 44 x 44 points.
- Support standard iOS gestures, such as swiping down from the top to reveal the Notification Center.
- Larger iPad screens are great for custom multi-finger gestures, but non-standard gestures should never be the only way to reach and use important features.



### 3.5.3 Comparison Google Play vs. App Store

#### Free and Paid Apps in Stores

App Store generates a higher number of shares as compared to Google Play Store. In 2020, App Store collected \$72.3 billion, and Google Play Store earned \$38.6 billion in the same period.

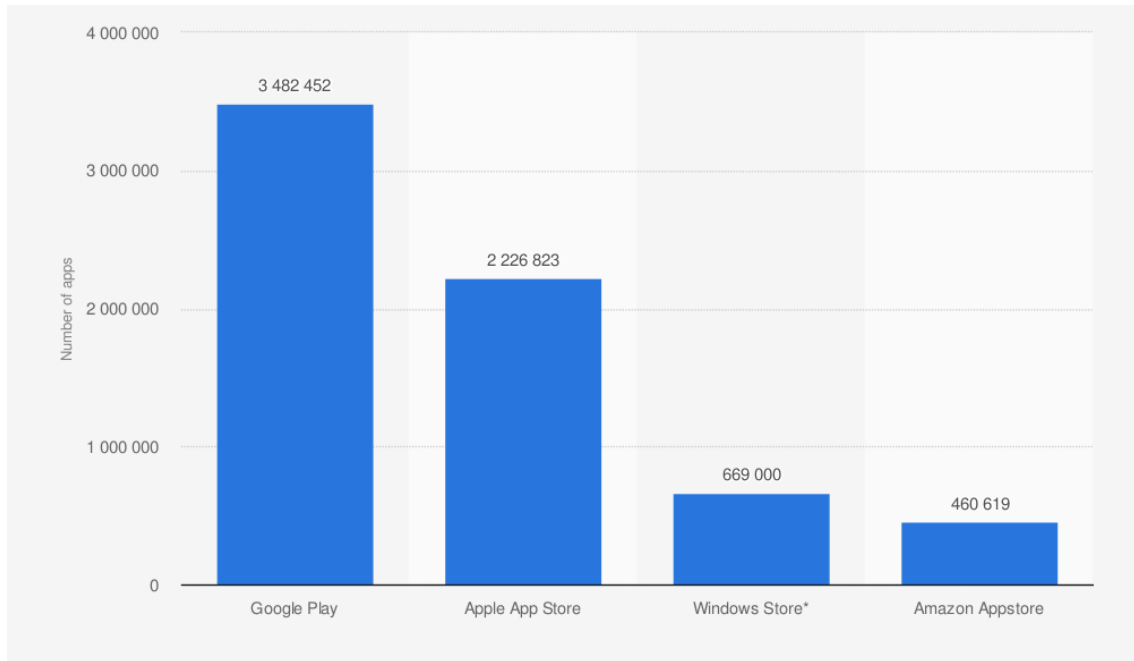


Figure 4 - Number of applications available in leading app stores as of 1st quarter 2021.

Source: statista.com

#### Types of Apps

Most of the iOS applications work smoothly on iPhone and iPad devices but many apps are built for smartphones, and rest are for the iPad tablets only. Android smartphones although powers a high number of smartphones and tablets operating different RAMs, CPUs, and other hardware configurations. That is yet another reason why Android applications developers need to perform more customization, optimization and scaling to suit various devices better.

#### Distribution platform

While Android is often criticized for the quality of its apps, Apple maintains tight control over iOS apps before they make it to the App Store. In fact, only in 2020, Apple reportedly rejected 150,000 apps submitted to their store (Marko Strizic, 2021). On the one

hand, this makes it difficult for everyone to post their software products, and on the other hand, it creates a database of proven applications. If App Store approves the program - it may be included in the list of recommended by the editorial office of the store and appear on the main page.

Financially, the App Store has been and remains more attractive to developers. Apple allows for more flexible customization of app monetization. For example, it is possible to make a paid program free for a specified time, then return its original cost. Such a trick is unfortunately not available on the Google Play store. As practice shows, the paid model of App Store app monetization is implemented an order of magnitude better and the developer, having initially made his product paid, has a chance to earn more than on Google Play.

### **Categorization**

The categorization system of App Store is less advanced as compared to Google Play Store. Recently, App Store has started to take over the categorization style of Google Play Store. The popular categories at Google Play Store are games, education, business, finance, tools, music & audio etc. App Store, on the other hand, is based on the simple phrase “There is an app for that.” Now App Store has started to apply categories system.

### **Reviews and Refunds**

Searching an application on Google Play Store is quite easily. Google’s Integrated search system was developed in order to increase searching efficiency. App Store has also added Google+ to its search system. It makes it easy for the users whether to download or skip any app if they have reviews or recommendations by people they already know.

### **Application approval process**

The approval process for the App Store can be long and drawn out, while the approval process for Google Play Store can be quite easy. Most of the people have the believe that App Store is safer as compared to Google Play Store. That is true because App Store takes lots of time in reviewing any application and then uploading on the main platform. Developers must be aware of the rules and ensure apps are error-free. Getting application approved is the biggest problem developers face while developing for the iOS App store. App Store security criteria are based on a high-quality standard that is why it takes so much

review time. Google Play Store contains some low-quality apps, and its reviews and uploading process is not as complex as compared to App Store.

## **Conclusion**

The iOS App Store and the Google Play Store are the big players in the application industry. Both have their own characteristics. Both have wide audiences and popular platforms, and both have formed excellent developer resources and user bases. While Google powers a bigger mobile device market than Apple, the App Store brings in more profits and has more monetization opportunities for developers. Every application developer first requires analyzing the requirement of each store and understand what exactly they want from their app before they go ahead to develop applications for any of these stores.

## **3.6 Application development lifecycle**

The Mobile Application Lifecycle contains different stages from development of the application to its testing and deployment. Creating mobile applications has some unique challenges (Mukherjea Sougata, 2016). The developer first needs to decide whether to create native, web or hybrid application. Research shows that out of the two dozen apps each of us have on our phones, we spend 80% of our time on just five of them.

### **3.6.1 Mobile application development process for iOS and Android**

Firstly, some preliminary evaluation of the cost and timing of project development should be done before creating future development cycle plan. Mobile App Development Lifecycle is based on Soft Development Life cycle principles. It consists of several stages such as Analysis, Design, Development (or Implementation), Testing, Deployment, and Maintenance.

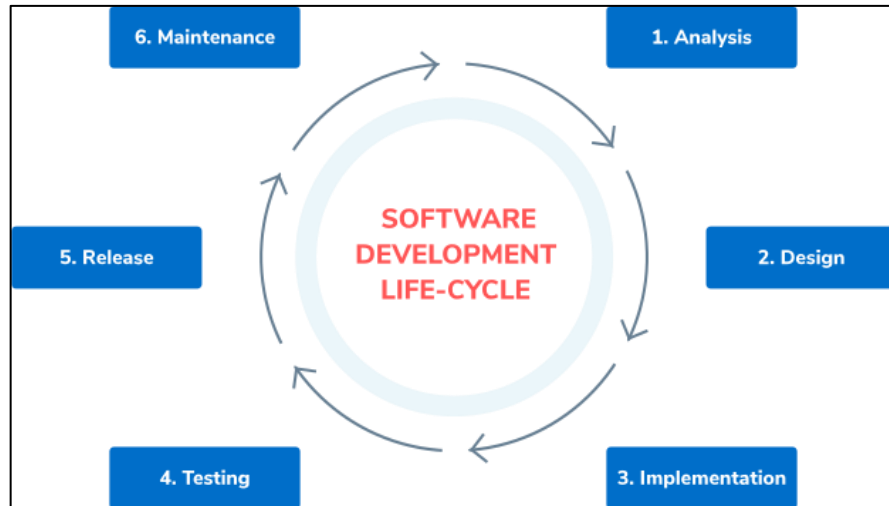


Figure 5 - Software Development Life cycle principles. Source: sumatosoft.com

### **Analysis**

Usually starts after the decision to make an application is done, based on the evaluation of the cost and time of project development done. It includes such stages:

- Preliminary collection of requirements
- Preparing preliminary budget and timeline estimates
- Understand what functions and use cases need to be implemented in the application

### **Design**

The main purpose of this step is to polish the application concept. Study the competitive environment and find best solutions for the application design. Here developer can focus on future application features. List key features that are desired to be in the application, specify technologies that could be used achieve full scope of desired functions (camera, microphone, GPS). At this stage developer decide what platform will be used as an application digital distribution place. It includes such stages:

- Form an interactive prototype of the application
  - screens interface layouts
  - navigation between application screens
- Write a technical assignment - a large document containing a complete description of all the requirements for project development.
- Choose a programming language and development tools

- Decide on monetization strategy
- Develop promotion strategy
- Design Product Requirements Document

## **Development**

The development stage of a mobile application can be launched when the design and technical assignment are ready. It Includes:

- Application interface layout
- Implementing navigation between screens
- Programming logic

Development is the most expensive and active phase of the mobile application development cycle. This stage is finished with a ready-to-use product.

## **Testing**

Development and testing stages are tightly connected. Usually testing include general product testing on each development phases, such as: alpha, beta, release versions. Collecting First True User Opinion and feedback is a part of product stabilization.

Testing stage includes:

- Functional tests
- Security tests
- Design tests
- Productivity tests
- Tests using targeted in design stage devices
- Regression tests (testing of all application features on concrete development version)
- User acceptance test (collecting feedback from the focus group)

## **Release**

Publishing to App Store / Google Play takes place after development is complete.

It Includes:

- App Store / Google Play Store developer account registration
- Designing application pages in App Store / Google Play Store
- Preparing and uploading application preview information, screenshots etc.
- Uploading builds
- Passing moderation

For many development teams, their first introduction to accessibility starts with an audit, lawsuit, or other notification of accessibility issues in a released application. The common reaction to these events is remediation, which triggers a QA-driven process of identifying issues and sending them back to development to be fixed (Mukherjea Sougata, 2016).

## **3.7 Development requirements and criteria**

Choosing optimal mobile platform for a mobile application development and release depends on several requirements and criteria.

### **3.7.1 Expediency**

Mobile Website vs. Mobile Application. In some cases, there is no need to develop a mobile application. A mobile application can be an opportunity to improve interaction with customers, create brand awareness, and even create additional revenue. But if the objectives of the app are unclear, customers can be upset, and money can be lost (McWherter Jeff; Gowel Scott, 2012).

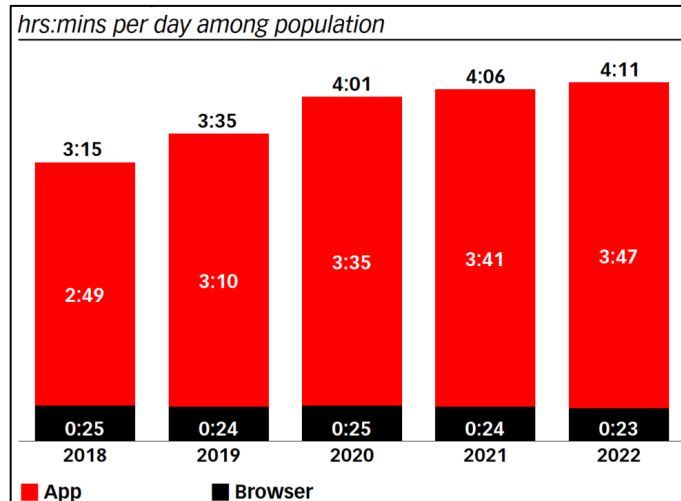


Figure 6 - Mobile Internet: average daily time spent in the US, App vs Browser.

Source: emarketer.com

Developing mobile application instead of mobile website is a better solution if:

- Graphics and processing power required
- Application needs access to device's camera, microphone, address book, media library etc.
- Processing payments using the market
- Designing a game

### 3.7.2 Cost of development

Development of a mobile application is associated with many different costs. To develop an application each developer needs some hardware and software. Developer needs devices to test the developed mobile application. Finally, to release application on any public market developer will need some licenses and developer account.

#### 3.7.2.1 Hardware cost

If you are seriously considering mobile development, you need to know that the emulator and simulators are great, but not perfect, so you'll need one of each of the types of devices you want to develop for (McWherter Jeff; Gowel Scott, 2012). Application behavior can differ when testing on emulator and on the device being emulated. Developing a mobile application, the developer should consider costs of development associated with hardware. It includes build station, testing devices, computer where to develop.

### 3.7.2.2 Developer Accounts

<b>Platform</b>	<b>Conditions</b>
App store	Enrolment cost \$99 per membership year. Strict App Store Review Guidelines to publish an application
Google play	One-time \$25 registration fee

Table 1 - Mobile platforms developer accounts conditions comparison. Source: author

### 3.7.2.3 Licenses

<b>Game engine</b>	<b>Price/Conditions</b>
Unreal engine	Free if revenue is less than \$100K in last 12 months
Unity	Free, charges %5 royalty after first \$1 million USD gross revenue
GameMaker: Studio	Price depends on platform, from Free to \$810 annual

Table 2 - Game engines license prices comparison. Source: author

## **The Bottom Line**

Total cost per developer to create, maintain and distribute mobile applications for all the platforms you can expect to pay a few thousand dollars just for the minimum infrastructure (McWherter Jeff; Gowel Scott, 2012).

## **3.8 Summary**

Development of mobile application is a complex and composite process. Mobile usage popularity is still growing at a steady rate due to the growing number of smartphones owner and industry development itself. As a result, there are tons of different approaches, development tools, frameworks, application types exist. To make the best decisions during the application development lifecycles it would be very handful to know differences between them, know best application use cases and to have a method how to compare and define the best decision according to the available recourses and existing goals. This thesis is dealing exactly with such kind of problems. At this point mobile application categories, types, development tools, operating systems and digital distribution platforms were described and compared. Making the best choices during the design stage of application development



lifecycle will result in the numerous benefits: saving time and money, efficient goals achieving.

## 4 Practical Part

### 4.1 Multiple-criteria decision analysis

To perform Multiple-criteria decision analysis Simple additive weighting method was chosen. The purpose of the analysis is to choose the best alternative. To assign weights each criterion was categorized according to such personal importance preference allocation:

- Very important – 4 points
- Important – 3 points
- Neutral – 2 points
- Low importance – 1 point

#### 4.1.1 Scenario

All the procedures in the MCDA and requirements and criteria selected were done and weighted respectively according to the predefined scenario. My thesis is looking on the problem from a perspective of the person, who makes the first application. The main goal of this developer is to gain practical experience in mobile application development and get acquainted with application release procedure. The developer does not want to invest much money in this idea as the main goal is not profit-oriented, so any income from the application release is not expected. This person also has access to free university library where the professional literature for application development and release was taken.

#### 4.1.2 MCDA of development tools

##### 4.1.2.1 Assigning weights chosen criteria for development tools

Development tools criteria:

- **License price**

Considerate all pricing conditions for the development tool, but the most important is pricing for the specific need, personal preference.

- **Editor complexity**

Based on the built-in tools provided by the engine and which programming language is used.

- **Market share (popularity)**

Indicates how much developers choose this tool among others, serves as an additional useful decision factor.

- **3D support**

Some of the game engines are 2D only and some supports 3D game development.

- **Tool power**

Based on provided amount and possibilities of built-in tools, increase in tool power also increases editor complexity. Importance of this criterion depends on personal preference and intended project size or possibilities.

- **Cross-platform development**

Ability to build the same coded application for both android and iOS operating systems using the same development tool.

- **Community support**

Based on the total amount of users on the related development tool forums.

After each criterion was categorized – weights were calculated in such way:

$$\text{Criterion weight} = \text{Criterion importance points} / \text{Sum number of all criteria points.}$$

Following table was created.

Criteria	Importance	Points	Weight
License price	Very important	4	0,2
Editor complexity	Very important	4	0,2
Market share (popularity)	Important	3	0,15
3D support	Important	3	0,15
Tool power	Important	3	0,15
Cross-platform development	Neutral	2	0,1
Community support	Low importance	1	0,05

Table 3 – Development tools criteria weights. Source: author

According to the scenario predefined in 4.1.1 the most important criteria chosen for the development tool were license price and editor complexity. The least important criterion chosen was community support.

#### 4.1.2.2 Defining set of alternatives for development tools

For development tools alternatives 3 the most popular game engines were chosen (Dustin Tyler, 2022):

- Unreal Engine
- Unity
- GameMaker: studio

#### 4.1.2.3 Assigning points for alternatives criteria for development tools

To perform Multiple criteria decision analysis using simple additive weighting method points for each alternative criterion were assigned respectively. Conditions for points assigning were chosen according to the scenario predefined in 4.1.1.

- **License price** – minimization function
  - Unity – has Personal plan which is free if revenue or funding is less than \$100K in the last 12 months. Advanced plans costs money and provides additional features. **Points assigned: 1**
  - Unreal engine – Publishing license is free to use but charges 5% royalty when your product succeeds. The first \$1 million USD of lifetime gross revenue your product makes is royalty-exempt. **Points assigned: 3**
  - GameMaker: studio – costs \$60 annually for Indie version which includes mobile platforms exports. **Points assigned: 10**

- **Editor complexity** – minimization function
 

According to Game engine comparison guide users found Unity to be slightly easier to use than Unreal engine, thanks to its native C# coding language (Evelyn Trainor-Fogleman, 2021). GameMaker: studio provides Drag and Drop feature which allows no-code game development experience.

  - Unity. Low editor complexity. **Points assigned: 4**
  - Unreal engine. High editor complexity. **Points assigned: 10**
  - GameMaker: studio. Lowest editor complexity. **Points assigned: 1**
  
- **Market share (popularity)** - maximization function
  - Unity. 48% (Mantra Malhotra, 2021). **Points assigned: 10**
  - Unreal engine. 13% (Mantra Malhotra, 2021). **Points assigned: 3**
  - GameMaker: studio. 4% (Mantra Malhotra, 2021). **Points assigned: 1**
  
- **3D support** - maximization function
  - Unity. Yes. **Points assigned: 10**
  - Unreal engine. Yes. **Points assigned: 10**
  - GameMaker: studio. Partially, 3D is not intended but could be realized. **Points assigned: 2**
  
- **Tool power** - maximization function
  - Unity. It usually takes longer time to achieve AAA-quality results comparing to Unreal Engine. **Points assigned: 9**
  - Unreal engine. Usually used for AAA-game projects due to its higher amount of features and possibilities. **Points assigned: 10**
  - GameMaker: studio. Usually used for small projects, learning in game development, engine's flexibility and power is significantly lower. **Points assigned: 3**

- **Cross-platform development** - maximization function
  - Unity. Yes, more than 20 platforms supported. **Points assigned: 10**
  - Unreal engine. Yes, more than 16 platforms supported. **Points assigned: 8**
  - GameMaker: studio. Yes, 15 platforms are supported for additional fees. **Points assigned: 5**
  
- **Community support** – maximization function
  - Unity. 278 000 members on reddit. **Points assigned: 10**
  - Unreal engine. 157 000 members on reddit. **Points assigned: 5**
  - GameMaker: studio. 68 000 members on reddit. **Points assigned: 3**

Following tables of criteria and points assigned were composed for development tools.

Simple additive weighting method							
Game engine	Market share (in %)	License price	3D support	Cross-platform development	Community support	Tool power	Editor complexity
Unity	48	Free	Yes	Yes	278000	High	Low
Unreal engine	13	Free, royalty fees	Yes	Yes	157000	High	High
GameMaker: studio	4	\$60 yearly	No	Yes	68000	Low	Lowest
Weights	0,15	0,2	0,15	0,1	0,05	0,15	0,2
	max	min	max	max	max	max	min

Table 4 – Development tools alternatives criteria parameters. Source: author

points							
Unity	10	1	10	10	10	9	4
Unreal engine	3	3	10	8	5	10	10
GameMaker: studio	1	10	2	5	3	3	1
	max	min	max	max	max	max	min

Table 5 – Development tools criteria points. Source: author

Afterwards minimization function criteria points were transformed into maximization functions points.

transformation							
Unity	10	9	10	10	10	10	6
Unreal engine	3	7	10	8	5	9	0
GameMaker: studio	1	0	2	5	3	3	9
	max	max	max	max	max	max	max

Table 6 – Development tools criteria points transformation. Source: author

Setting ideal and nadir alternatives for development tools.

Game engine	Market share (in %)	License price	3D support	Cross-platform development	Community support	Tool power	Editor complexity
H (Ideal variant)	10	9	10	10	10	10	9
D (Nadir variant)	1	0	2	5	3	3	0

Table 7 – Development tools ideal and nadir alternatives. Source: author

Calculation of standardized criterial matrix R for development tools.

matrix R							
Unity	1	1	1	1	1	1	0,67
Unreal engine	0,22	0,78	1	0,6	0,29	0,86	0
GameMaker: studio	0	0	0	0	0	0	1
weights	0,15	0,2	0,15	0,1	0,05	0,15	0,2

Table 8 – Development tools criterial matrix R. Source: author

#### 4.1.2.4 Result of development tools' MCDA

Unity game engine turned out to be the best alternative based on selected requirements and criteria among defined set of alternatives for development tools.

Result	Score	Choice number
Unity	0,93	1
Unreal engine	0,54	2
GameMaker: studio	0,20	3

Table 9 – Development tools' MCDA result. Source: author

#### 4.1.3 MCDA of release platforms

##### 4.1.3.1 Assigning weights chosen criteria for release platforms

Release platform criteria:

- **Developer account cost**

Based on the cost itself and payment system (one-time fee vs. regular payments).

- **Releasing procedure complexity**

Based on how difficult it is to pass the before-release application review platform process, how much time it usually takes.

- **Number of users**

Allows to predict possible earnings on one exact platform comparing to another one.

- **Money spend rate**

Also allows to predict possible earnings on one exact platform comparing to another one.

- **Hardware cost**

Based on the hardware cost that is needed during application development cycle(testing/developing/building). For example, to build an application for iOS MacOS-based computer is needed.



- **Promotion features**

Based on which and how much conditions application needs to meet to get free promotion by the platform.

- **App monetization customization**

Based on how flexible monetization of the game could be – for example make an application being free for a limited amount of time.

- **Fees for iAP**

Based on the transaction fees for in-app purchases.

According to the predefined scenario in 4.1.1 the most important criteria chosen for the release platform was developer account cost. The least important criterion chosen were app monetization customization and fees for iAP.

Following table was created.

<b>Criteria</b>	<b>Importance</b>	<b>Points</b>	<b>Weight</b>
Developer account cost	Very important	4	0,2
Releasing procedure complexity	Important	3	0,15
Number of users	Important	3	0,15
Money spend rate	Important	3	0,15
Hardware cost	Important	3	0,15
Promotion features	Neutral	2	0,1
App monetization customization	Low importance	1	0,05
Fees for iAP	Low importance	1	0,05

Table 10 – Release platforms criteria weights. Source: author

#### 4.1.3.2 Defining set of alternatives for development tools

For release platforms alternatives two the most used platforms were chosen:

- Google Play
- App Store

#### 4.1.3.3 Assigning points for alternatives criteria for release platforms

To perform Multiple criteria decision analysis using simple additive weighting method points for each alternative criterion were assigned respectively.

- **Developer account cost** – minimization function
  - Google Play. One-time \$25 fee. **Points assigned: 1**
  - App Store. \$99 annually. **Points assigned: 10**
  
- **Releasing procedure complexity** – minimization function
  - Google Play. Easy approval process, developers can be more creative and be free to experiment. **Points assigned: 1**
  - App Store. It is not easy to get an application approved by the App Store, it can be rejected for slight errors. Developers must be sure that their apps fit Apple's standards (Priya Viswanathan, 2020). **Points assigned: 10**
  
- **Number of users** – maximization function
  - Google Play. 2.8 billion active users in 2020. **Points assigned: 10**
  - App Store. 1 billion active users in 2020. **Points assigned: 1**
  
- **Money spend rate** (per user annually) – maximization function
  - Google Play. Revenue in 2020 = \$38.6 billion.  
Money spend rate = Revenue / Number of users = ~\$14.  
**Points assigned: 1**
  - App Store. Revenue in 2020 = \$72.3 billion. Money spend rate = ~\$72.  
**Points assigned: 10**
  
- **Hardware cost** – minimization function  
Estimated on average building station + testing device prices.
  - Google Play. Average android device price is \$261. Average PC price is \$1000. Total cost = \$1261. **Points assigned: 1**
  - App Store. Average iOS device price is \$758. Average macOS device price is \$1432. Total cost = \$2190. **Points assigned: 10**

- **Promotion features** – maximization function
  - Google Play. Google provides paid Ads Universal App Campaigns which will show promoted application in Google Play, Search on Google.com, YouTube. Application also can be feature on Google Play for free but due to higher number of apps on this platform – the competitiveness is much higher compared to App Store. **Points assigned: 1**
  - App Store. Once the application is approved – it has a good chance of being promoted through multiple channels, for example, being featured on the popular App category, App of the week, and more. App store also provides ability to have application being paid promoted. **Points assigned: 10**
  
- **App monetization customization** – maximization function
 

Both platforms provide almost same functionality for app monetization. Business models: free, freemium, paid, paymium, in-app purchase types. Both distribution platforms provide free trial functionality for the paid apps.

  - Google Play. **Points assigned: 4**
  - App Store. **Points assigned: 6**
  
- **Fees for IAP** – minimization function
 

Both distribution platforms have 30% standard commission on apps and in-app purchases of digital goods and services. Both Google Play and App store have reduced to 15% commission fees program for developers, but with slightly different conditions.

  - Google Play. Service fee is 15% for the first \$1M of earnings each year. **Points assigned: 4**
  - App Store. Service fee is 15% if developers make less than \$1 million in annual net app revenue and they are new to the App Store. **Points assigned: 6**

Following tables of criteria and points assigned were composed for release platforms.

Simple additive weighting method								
Release platform	Developer account cost	Releasing procedure complexity	Number of users	Money spend rate	Hardware cost	Promotion features	App monetization customization	Fees for iAP
Google Play	\$25 One-time fee	Low	2.8 billion	\$11	\$1261	Yes	Flexible	15-30% fee
App Store	\$99 annually	High	1 billion	\$72	\$2190	Yes	Flexible	15-30% fee
Weights	0,2	0,15	0,15	0,15	0,15	0,1	0,05	0,05
	min	min	max	max	min	max	max	min

Table 11 – Release platforms alternatives criteria parameters. Source: author

points								
Google Play	1	1	10	1	1	1	4	4
App Store	10	10	1	10	10	10	6	6
	min	min	max	max	min	max	max	min

Table 12 – Release platforms criteria points. Source: author

Afterwards minimization function criteria points were transformed into maximization function points.

transformation								
Google Play	10	10	10	1	10	1	4	6
App Store	1	1	1	10	1	10	6	4
	max	max	max	max	max	max	max	max

Table 13 – Release platforms criteria points transformation. Source: author

Setting ideal and nadir alternatives for release platforms.

Release platform	Developer account cost	Releasing procedure complexity	Number of users	Money spend rate	Hardware cost	Promotion features	App monetization customization	Fees for iAP
H (Ideal variant)	10	10	10	10	10	10	6	6
D (Nadir variant)	1	1	1	1	1	1	4	4

Table 14 – Release platforms ideal and nadir alternatives. Source: author

Calculation of standardized criterial matrix R for release platforms.

matrix R								
Google Play	1	1	1	0	1	0	0	1
App Store	0	0	0	1	0	1	1	0
weights	0,2	0,12	0,15	0,15	0,15	0,1	0,05	0,05

Table 15 – Release platforms criterial matrix R. Source: author

#### 4.1.3.4 Result of release platforms' MCDA

Google Play release platform turned out to be the best alternative based on selected requirements and criteria among defined set of alternatives for release platforms.

Result	Score	Choice number
Google Play	0,67	1
App Store	0,30	2

Table 16 - Release platforms' MCDA result. Source: author

## 4.2 Prototype application development

### 4.2.1 Hardware specifications

During the practical of the thesis computer with following specifications was used:

- Processor: AMD FX-8350 4 GHz
- RAM: 8 GB
- GPU: GEFORCE GTX 1650 SUPER 4 GB
- SSD: HYPERX 240 GB
- OS: Windows 10

The mobile phone which was used for testing during the application development is Meizu M15 Lite:

- Processor: Qualcomm Snapdragon 626
- RAM: 4 GB
- OS: Android 7.1.2 Nougat

## 4.2.2 Unity installation

The prototype application development begins with installation of the development tool – according to performed MCDA the best alternative according to selected requirements and criteria. To access unity software developer firstly needs to select a plan and create a developer account. It was done via following link: [store.unity.com](https://store.unity.com)  
According to predefined scenario and selected criteria personal plan was chosen.

**Plans and pricing**

We offer a range of plans for all levels of expertise and industries.  
All plans are royalty-free.

Individual Teams

**Student**  
Learn the tools and workflows professionals use on the job  
Free  
[Sign up](#)  
Eligibility:  
Students 16 years and older who are enrolled in an accredited educational institution and can provide consent to the collection and processing of their personal information.

**Personal**  
Start creating with the free version of Unity  
Free  
[Get started](#) [Learn more](#)  
Eligibility:  
Revenue or funding less than \$100K in the last 12 months

**Unity Learn**  
Master Unity with expert-led live sessions and on-demand learning  
[Start learning](#)

Figure 7 – Unity plans and pricing. Source: [store.unity.com](https://store.unity.com)

Next step was Unity Hub installation. Unity Hub is Unity Editor manager which allows to manage multiple installations of the Unity Editor, create new projects, and access your work. To be able to use Unity Hub and Unity Editor - Unity ID is needed. It was created using following link: [id.unity.com](https://id.unity.com)

Next step was choosing Unity Editor version. It is recommended to install version which is marked LTS. It stands for Long Term Support version which rolls up the quality features and improvements made for Unity editor across the year by Unity Technologies into a single installation. 2019.4.35f1 LTS Unity Editor version was chosen.

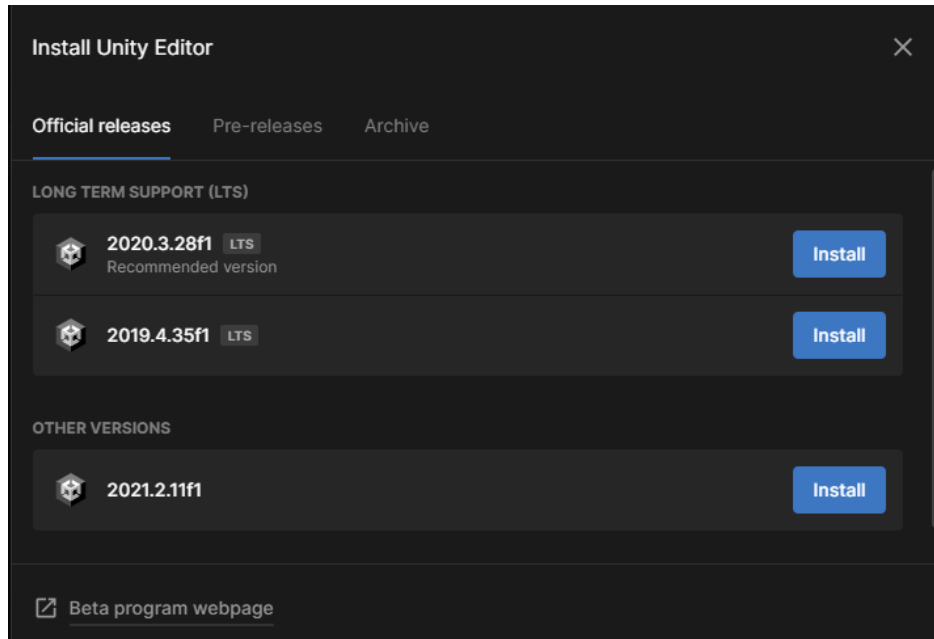


Figure 8 – Unity Editor official releases. Source: author

To build an application for Android it was obligatory to add following modules to the Unity Editor while installation:

- Android Build Support
  - Android SDK & NDK Tools
  - OpenJDK

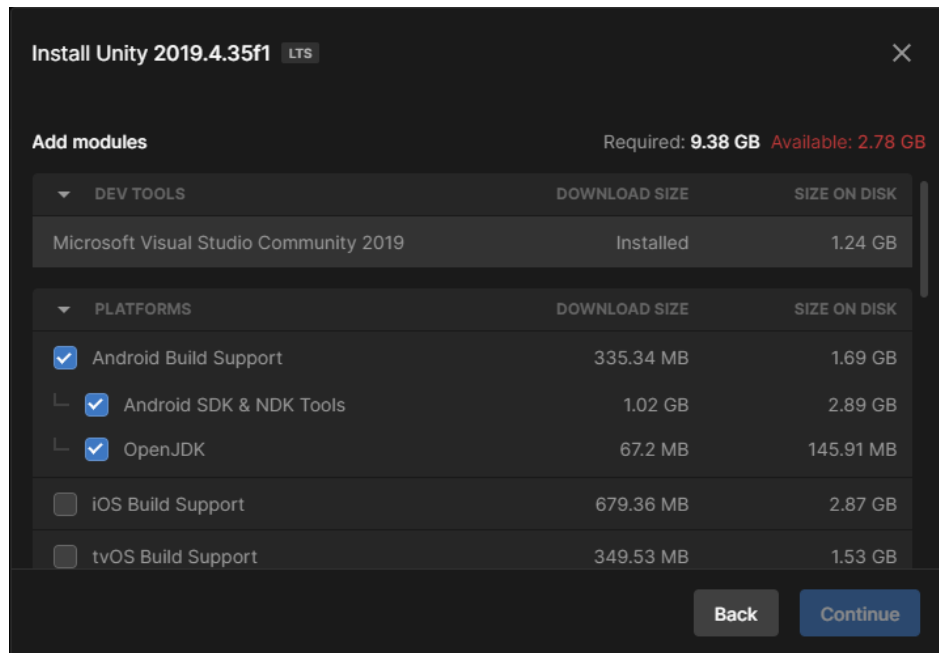


Figure 9 – Unity Editor modules. Source: author

Next step was project creation. Unity Hub provides a few different templates for project creation. 3D Mobile template was chosen.

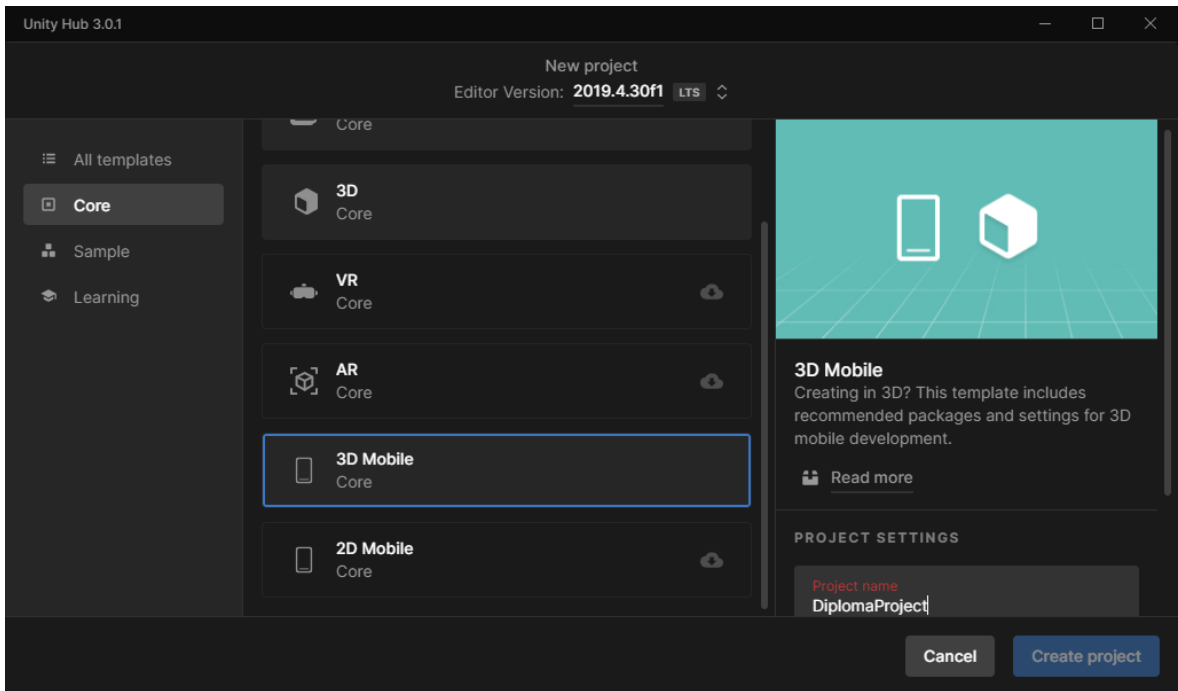


Figure 10 – Unity project creation templates. Source: author

Platform was switched to Android in Build Settings on first Unity Editor launch.

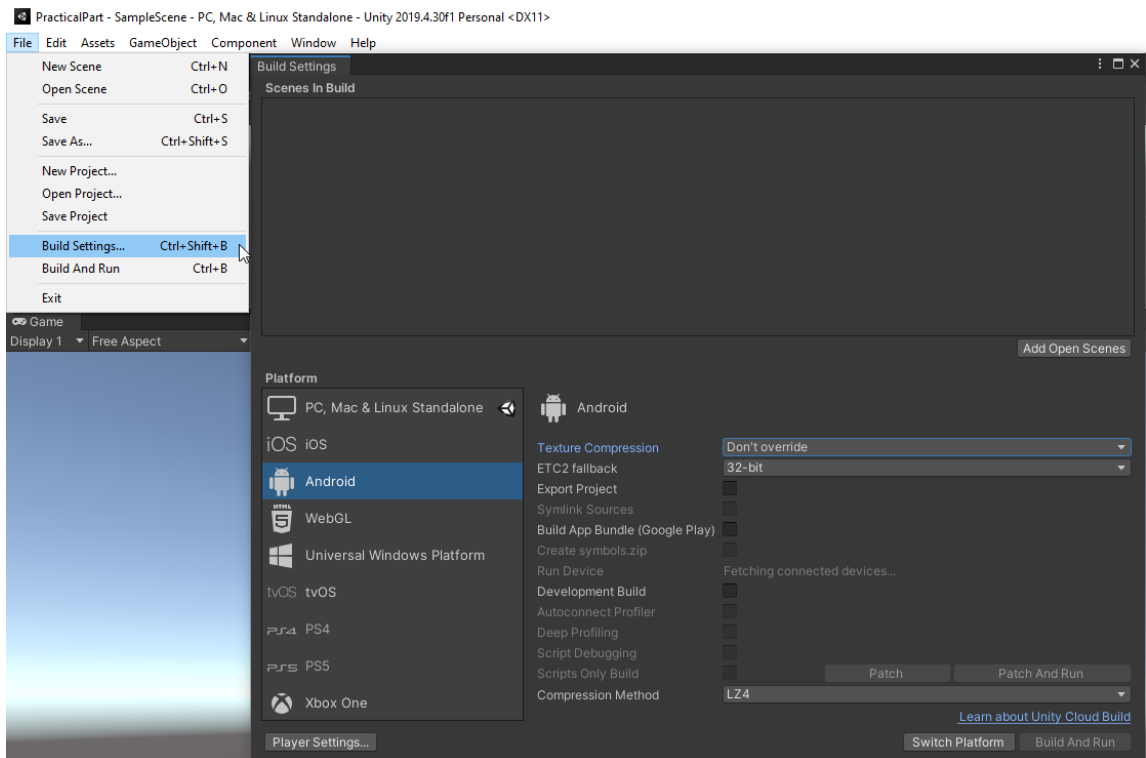


Figure 11 – Unity project build settings. Source: author



At this point Unity Editor was already prepared for developing and building the prototype application for Android.

### 4.2.3 Mobile game prototype development

#### 4.2.3.1 Game idea

Title of the game is “Tiles Drop”. The main idea of the game is there are a lot of tiles on the level between start and finish platforms and only a part of them are real ones. The player’s mission is to find the real path and reach the finish platform.

#### 4.2.3.2 Game development

The development started with creating two scenes: MainGame and Menu.

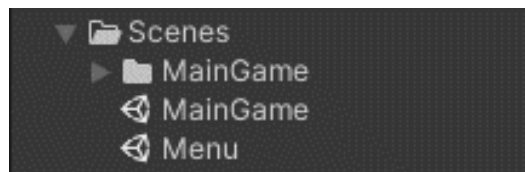


Figure 12 – Project scenes. Source: author

To the MainGame scene free Joystick Pack was imported from Unity Asset Store. The purpose of this pack is to let player control the character via mobile controller input system.

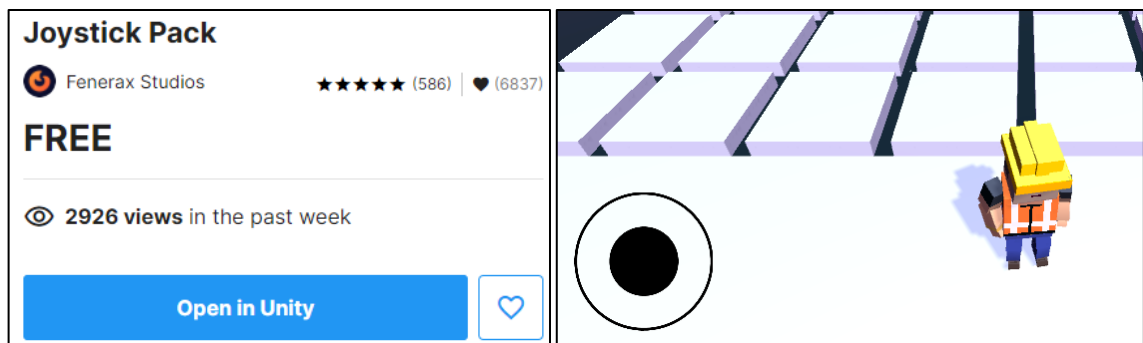


Figure 13 – Joystick Pack asset. Source: author

GameManager object and script were created. The purpose of GameManager is to control the whole game flow.

```
GameManager.cs
Assembly-CSharp
GameManager

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.Advertisements;
5 using UnityEngine.SceneManagement;
6
7 [Component Unity | Count: 2]
8 public class GameManager : MonoBehaviour
9 {
10     public GameObject gameOverMenu;
11     public GameObject joystick;
12     public GameObject youWonMenu;
13     public GameObject points;
14
15     [Component Unity | Count: 0]
16     void Start()
17     {
18         Advertisement.Initialize("4618017");
19     }
20
21     [Component Unity | Count: 1]
22     public void GameOver()
23     {
24         gameOverMenu.gameObject.SetActive(true);
25         points.SetActive(true);
26         GameObject.FindGameObjectWithTag("energy").GetComponent<TMPPro.TextMeshProUGUI>().text = "Energy: " + PlayerPrefs.GetInt("energy");
27         GameObject.FindGameObjectWithTag("points").GetComponent<TMPPro.TextMeshProUGUI>().text = "Total points: " + PlayerPrefs.GetInt("points");
28         joystick.gameObject.SetActive(false);
29     }
30
31     [Component Unity | Count: 1]
32     public void YouWon()
33     {
34         youWonMenu.gameObject.SetActive(true);
35         points.SetActive(true);
36         GameObject.FindGameObjectWithTag("energy").GetComponent<TMPPro.TextMeshProUGUI>().text = "Energy: " + PlayerPrefs.GetInt("energy");
37         PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") + GameObject.FindGameObjectWithTag("level").GetComponent<LevelGenerator>().gameLevel);
38         GameObject.FindGameObjectWithTag("points").GetComponent<TMPPro.TextMeshProUGUI>().text = "Total points: " + PlayerPrefs.GetInt("points");
39         joystick.gameObject.SetActive(false);
40     }
41 }
```

Figure 14 – GameManager script. Source: author

Key game elements were developed and implemented to the MainGame scene:

- Player
- Environment
- Level Generator
- Canvas (UI)

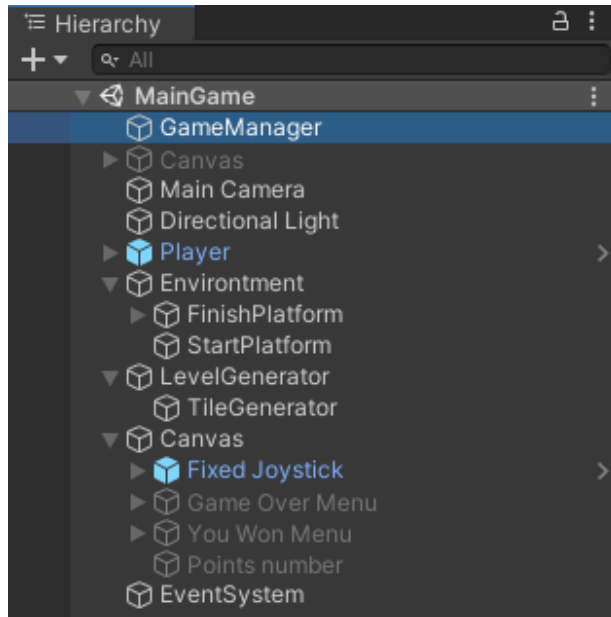


Figure 15 – MainGame scene hierarchy. Source: author

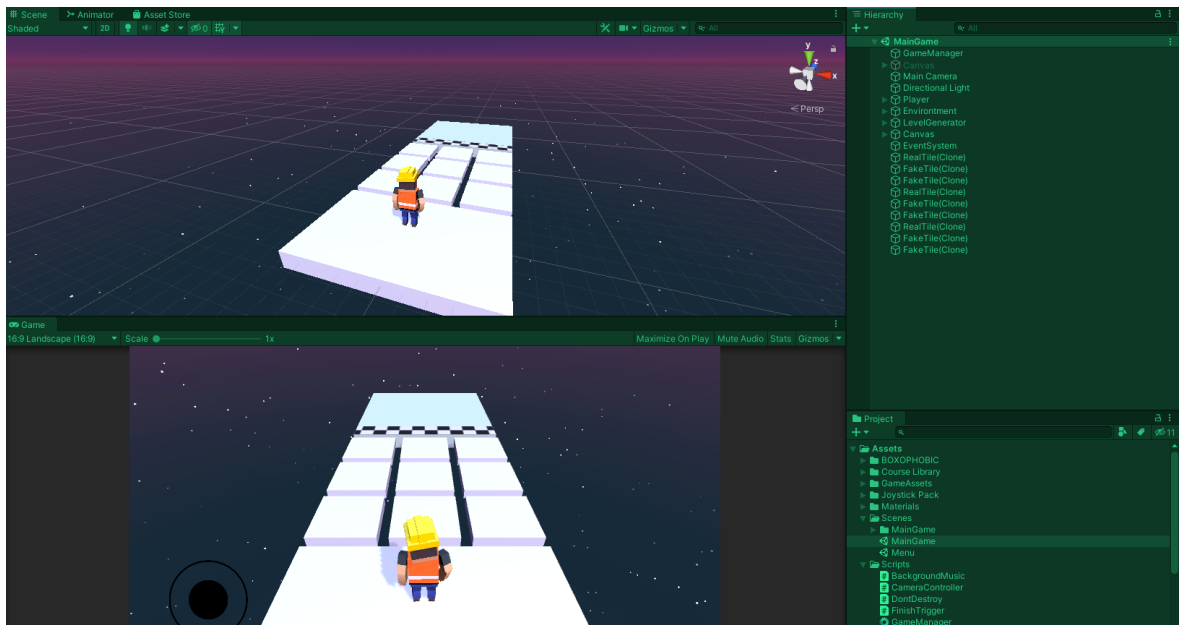


Figure 16 – Game development environment. Source: author

Menu scene was finished, core game controlling functionality was implemented.

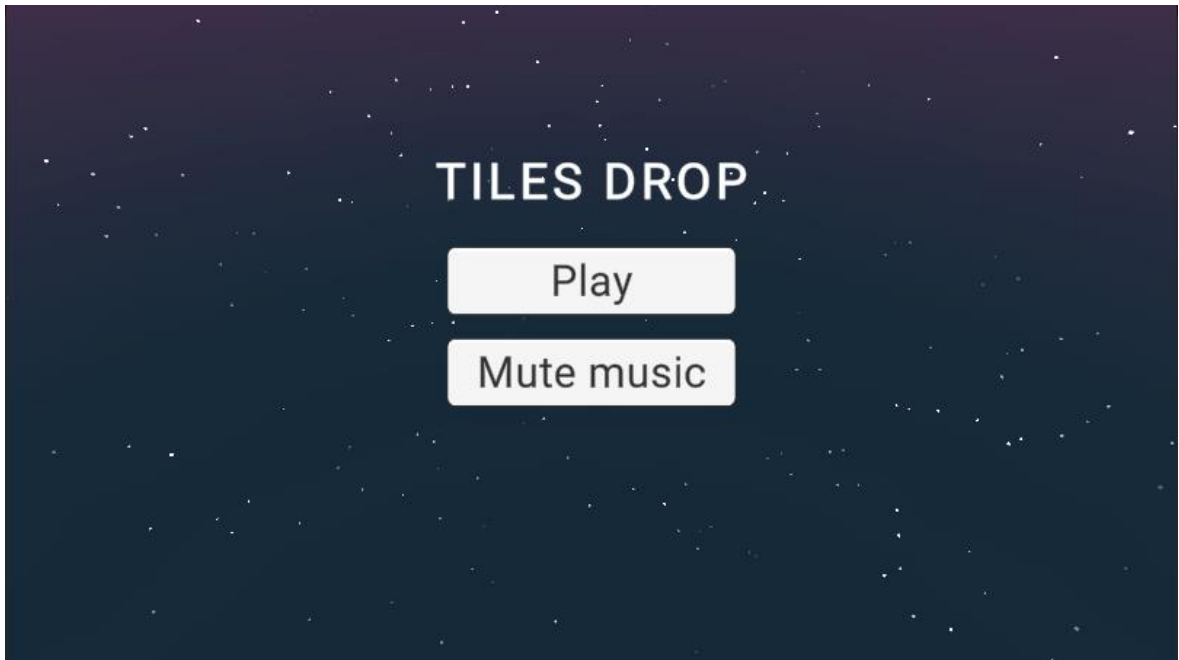


Figure 17 – Tiles Drop main menu. Source: author

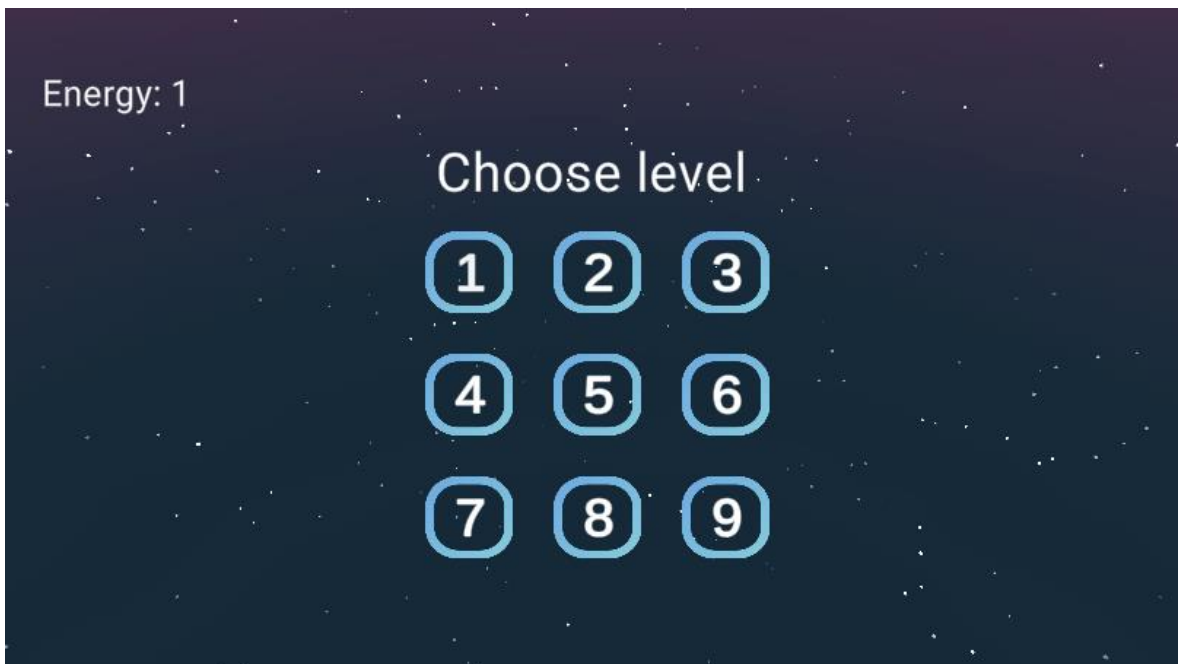


Figure 18 – Tiles Drop levels screen. Source: author



Figure 19 – Tiles Drop gameplay. Source: author

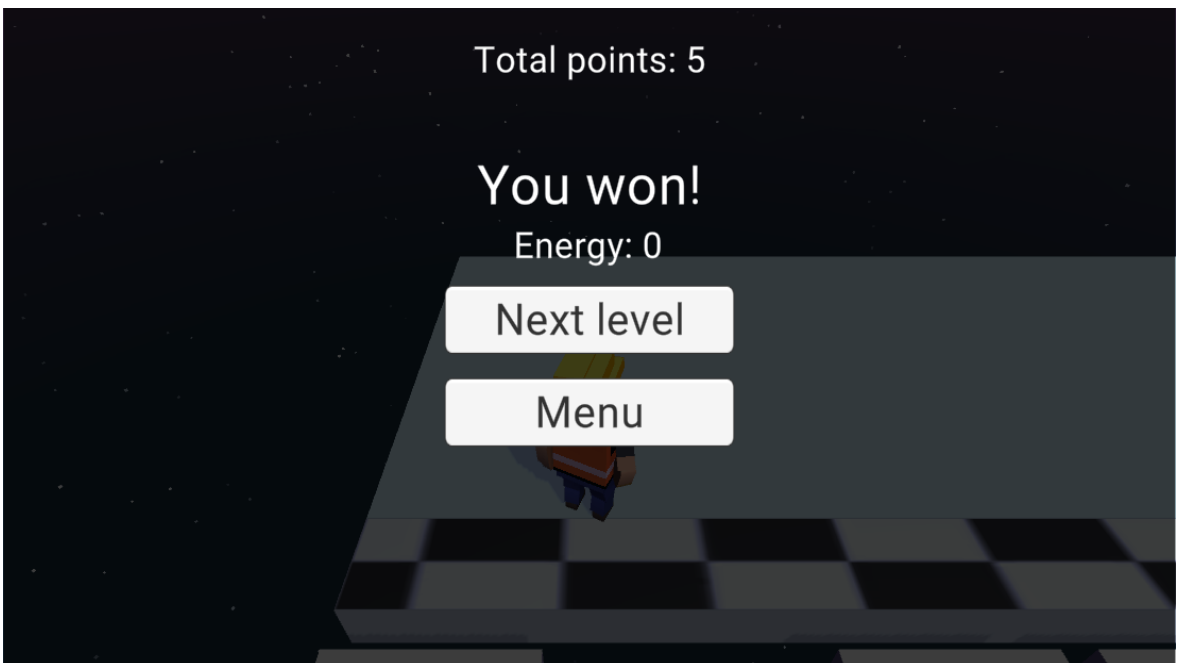


Figure 20 – Tiles Drop win screen. Source: author

At this point game development was finished, prototype application was ready.

### 4.2.3.3 Building game

To satisfy Google Play releasing requirements Target API Level was set to API level 30 and Scripting Backend was set to IL2CPP (ARM64 checked).

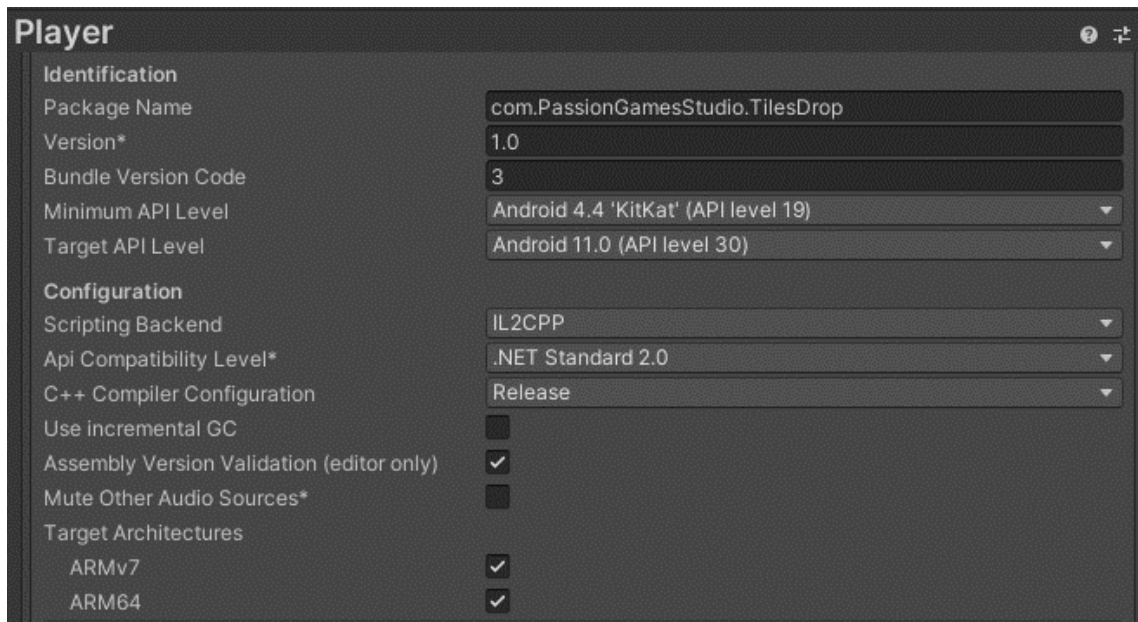


Figure 21 – Project player build settings. Source: author

Both scenes were selected, Build App bundle (Google Play) option was checked. The game was ready to be built.

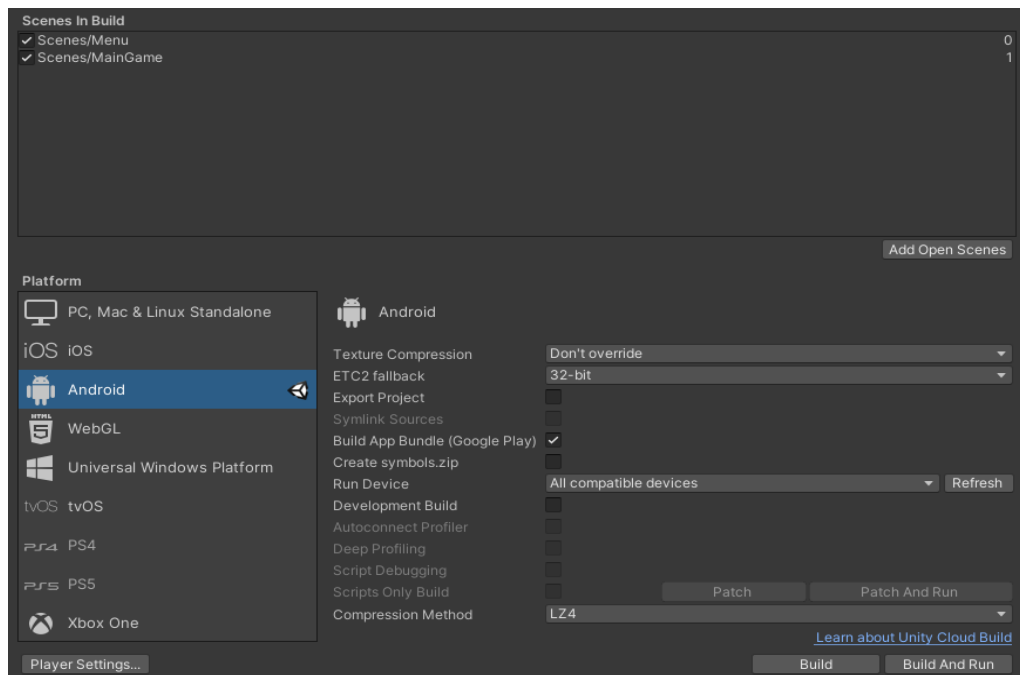


Figure 22 – Project build options. Source: author

## 4.3 Application release

### 4.3.1 Creating Google Play developer account

To create Google Play developer account following link was used:  
[play.google.com/console/about](https://play.google.com/console/about)

Google Play Console provides two developer account types:

- For personal use. Suits for students, hobbyists, semi-professional developers
- For organization or business. Suits for commercial, industrial, professional, or governmental activities

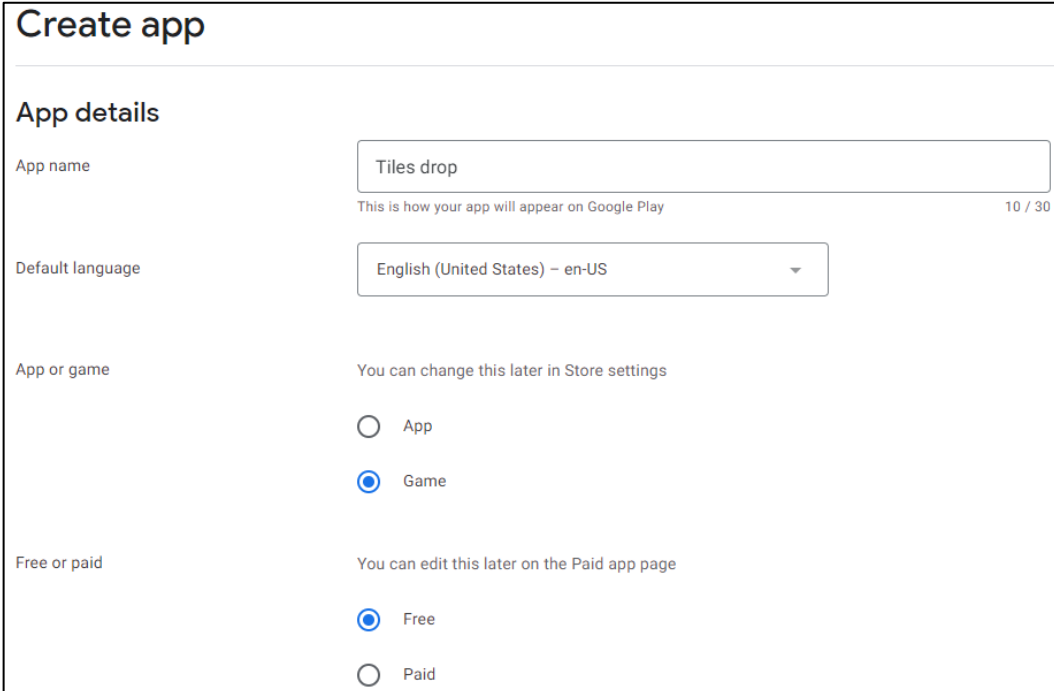
Personal developer account was selected.

After a few fill-in information steps and payment procedure developer account was created.

### 4.3.2 Creating an application

To create an application following information was filled in or selected:

- App name
- Default language
- App or game
- Free or paid application type



The screenshot shows the 'Create app' form in Google Play Console. The form is titled 'Create app' and is divided into several sections. The first section is 'App details', which includes a text input field for 'App name' containing 'Tiles drop', a dropdown menu for 'Default language' set to 'English (United States) - en-US', and radio buttons for 'App or game' with 'Game' selected. Below this, there are radio buttons for 'Free or paid' with 'Free' selected. The form also includes helpful text: 'This is how your app will appear on Google Play' with a character count '10 / 30', and 'You can change this later in Store settings' and 'You can edit this later on the Paid app page'.

Figure 23 – Creating an application on Google Play. Source: [play.google.com](https://play.google.com)

Following tasks were completed to set up the game before release:

- Content of the app
  - App access
  - Content rating
  - Target audience
  - News apps
  - COVID-19 contact tracing and status apps
  - Data safety
- How the app is organized and presented
  - Select an app category and provide contact details
  - Set up your store listing

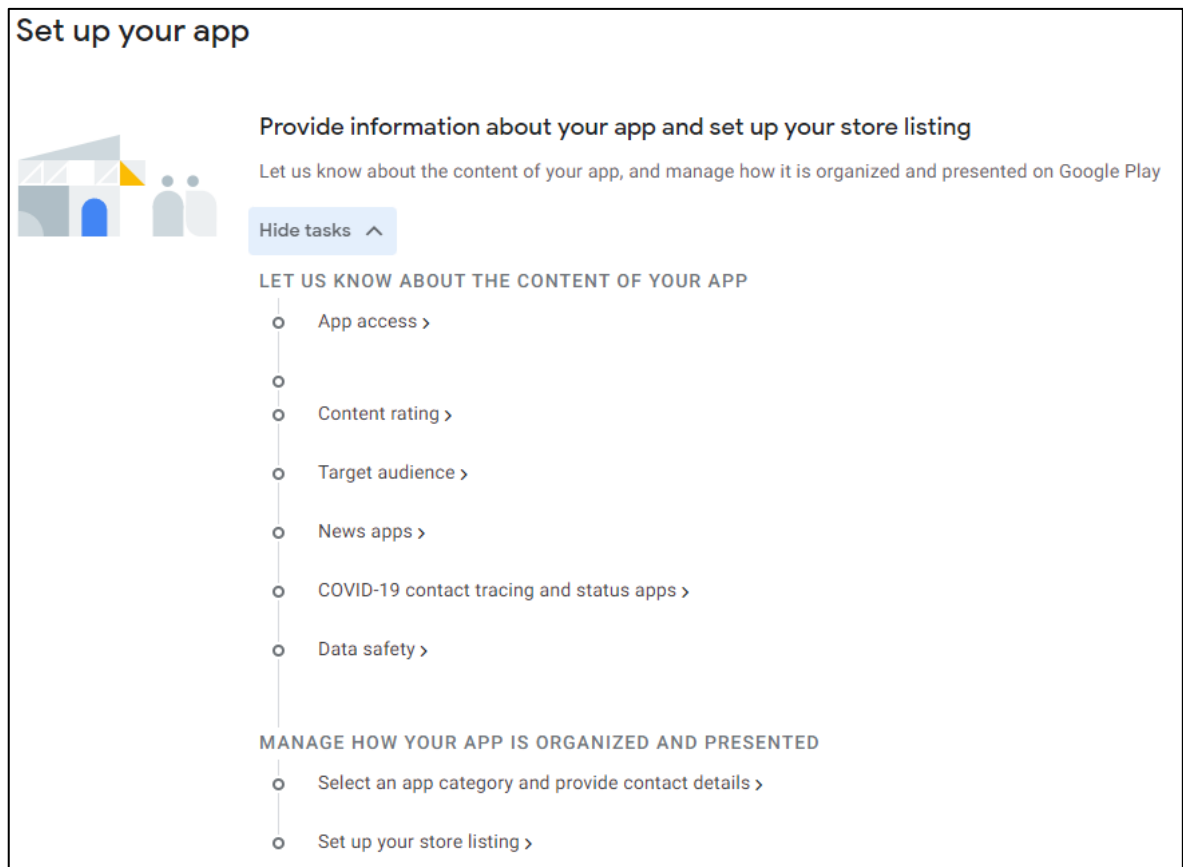
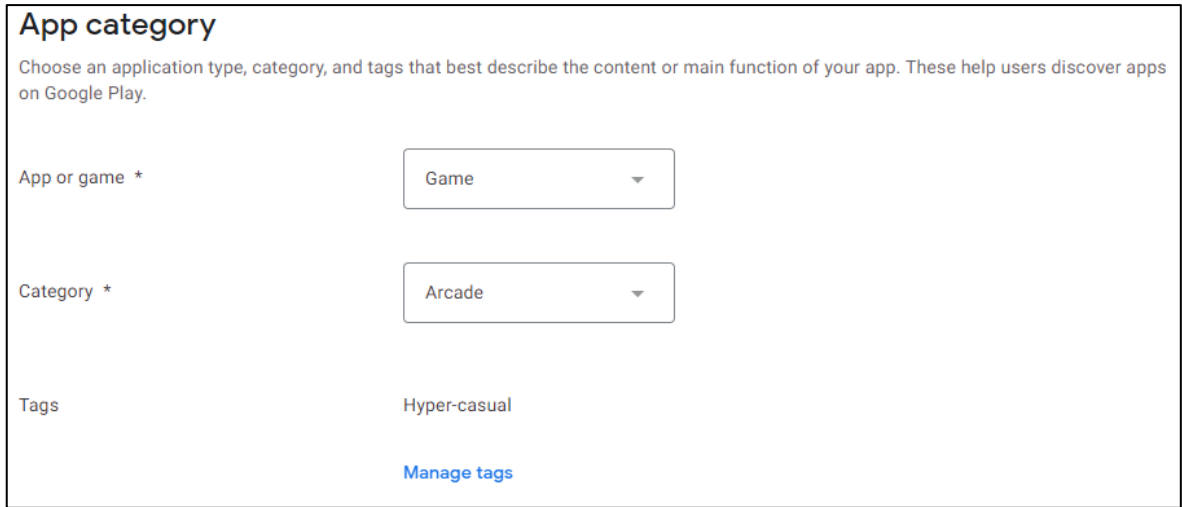


Figure 24 – Tasks for setting up the app. Source: play.google.com



Game category was chosen, and tags were added



**App category**

Choose an application type, category, and tags that best describe the content or main function of your app. These help users discover apps on Google Play.

App or game \*

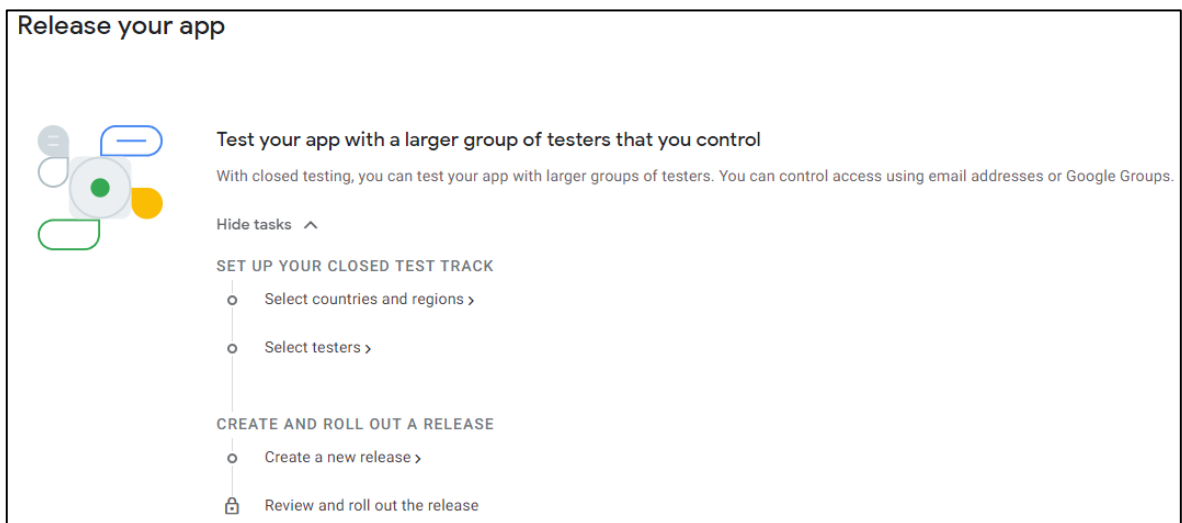
Category \*

Tags


[Manage tags](#)

Figure 25 – App category and tags. Source: play.google.com

Next steps were countries and regions selection and release creation. All 176 proposed countries were selected.



**Release your app**



**Test your app with a larger group of testers that you control**

With closed testing, you can test your app with larger groups of testers. You can control access using email addresses or Google Groups.

Hide tasks ^

**SET UP YOUR CLOSED TEST TRACK**

- Select countries and regions >
- Select testers >

**CREATE AND ROLL OUT A RELEASE**

- Create a new release >
- Review and roll out the release

Figure 26 – Application test countries and release creation. Source: play.google.com

App Bundle which was built in 4.2.2.3 was uploaded.

To be published the game must be approved by Google review. It took 3 business days to be reviewed.

The screenshot shows a notification titled "Changes in review" with a question mark icon. A tooltip explains: "These changes are being reviewed. Changes must be approved by Google before you can publish them." Below the notification is a table with the following content:

Item changed	
Production	
Tiles Drop	Full rollout started
Countries / regions	176 countries / regions added: Albania, Algeria, and 174 more
Countries / regions	Rest of world added

Figure 27 – Production release review state. Source: play.google.com

Finally, the game was released and now can be downloaded from this link:  
<https://play.google.com/store/apps/details?id=com.PassionGamesStudio.TilesDrop>

The screenshot shows the Google Play Store page for the game "Tiles drop" by Passion Games Studio. The page includes the game's icon, a character wearing a yellow hard hat and an orange safety vest, and the following details: "Passion Games Studio Arcade", "PEGI 3", "Contains Ads", and an "Add to Wishlist" button. A green "Install" button is visible in the top right. Below the main information is a preview image of the game's title screen, which features the text "TILES DROP" and two buttons: "Play" and "Mute music". On the right side of the preview, the text "Energy:" is visible.

Figure 28 – Tiles Drop page in the Google Play Store. Source: play.google.com

## 5 Results and Discussion

Based on the reviewed literature development tools and release platforms were analyzed. In practical part according to selected requirements and criteria preferred development tool and release platform were selected. To determine best alternatives Simple additive method of Multiple-criteria decision analysis was used.

To determine the best development tool alternative among seven chosen two following criteria were the most important: License price, Editor complexity. Unity Editor was selected as the best development tool alternative. The license for Unity usage was free and the editor turned out to be easy enough to make prototype application.

To determine the best release platform alternative among eight chosen criteria developer account cost was the most important criterion. Google Play was selected as the best release platform alternative. \$25 was spent on the developer account cost as expected in the MCDA part. Number of users and Money spend rate criteria were chosen as Important level factors with weights = 0.15. The application didn't succeed and didn't get many downloads. It turns out that Number of users and Money spend rate factors were not that important. In predefined scenario it was mentioned that the main goal of the application development is to gain practical experience, not profit. So, these two factors should be evaluated as Neutral level criteria with weights = 0.1.

Hardware cost was also evaluated as Important level criteria. The actual price of the computer used is approximately \$750, comparing to average PC price \$1000 defined in the 4.1.3 MCDA. Moreover, this money was not spent on the computer as I already had it before writing the thesis. The price of the mobile phone is approximately \$200, comparing to average android device defined in the 4.1.3 MCDA. Moreover, this money was not spent on the mobile phone as I already had it before writing the thesis. To sum up, the developer predefined in 4.1.1 Scenario is most likely to have a computer and a mobile phone for testing beforehand as well. So, the Hardware cost criterion level should be lowered to Neutral level criterion with weigh = 0.1. All others criteria importance levels were evaluated reasonably and should not be changed.

The prototype game was developed and released successfully. It is a simple game where the player's mission is to find the real path and reach the finish platform. It took 3 business days for the application to be reviewed and released by Google Play team. Now the developed game can be found on the pages of Google Play Store. The name of the game is "Tiles drop".

## **6 Conclusion**

As mentioned before, nowadays mobile applications industry is growing at a steady rate. It is very important to make reasonable, efficient, and practical decisions during application development lifecycle as it could save a lot of resources.

The main and partials goals of the thesis have been met. Professional and scientific information sources have been studied and analyzed. Different development tools and release platforms were analyzed and compared using multiple-criteria decision analysis based on selected requirements and criteria. In general, developed MCDA properly predicted the situation for the scenario defined in 4.1.1. Some minor improvements for the criteria importance level can be done further, as mentioned in 5 Results and Discussion section.

An experimental prototype application has been developed and launched utilizing the most suitable variant. Now the developed game can be found and downloaded from the Google Play Store.

## 7 References

1. Anastasiya Marchuk. Native Vs Cross-Platform Development: Pros & Cons Revealed. 2021. <https://www.uptech.team/blog/native-vs-cross-platform-app-development>
2. Bridget Poetker. What Are the Different Types of Mobile Apps? 2019. <https://learn.g2.com/types-of-mobile-apps>
3. Buttfeld–Addison, Paris; Manning, Jonathon; Nugent, Tim. Unity Game Development Cookbook: Essentials for Every Game. O'Reilly; 1st edition, 2019. ISBN 978-1491999158
4. Dustin Tyler. How to Choose the Best Video Game Engine. 2022. <https://www.gamedesigning.org/career/video-game-engines/>
5. Evelyn Trainor-Fogleman. Unity vs Unreal Engine: Game engine comparison guide for 2021. 2021. <https://www.evercast.us/blog/unity-vs-unreal-engine>
6. Heather Wellington. Native Apps vs Hybrid Apps Comparison. 2021. <https://saucelabs.com/resources/articles/native-apps-vs-hybrid-apps-comparison>
7. L. CECI. Number of apps available in leading app stores as of 2021. 2021. <https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/>
8. Lewis, Shaun; Dunn, Mike. Native Mobile Development. O'Reilly Media, Inc.; 1st edition, 2019. ISBN 9781492052876
9. Lionel Valdellon. What Are the Different Types of Mobile Apps? And How Do You Choose? 2020. <https://clevertap.com/blog/types-of-mobile-apps/>
10. Mansoor Iqbal. App Revenue Data. 2022. <https://www.businessofapps.com/data/app-revenues/>
11. Mantra Malhotra. Unreal Engine vs Unity 3D Games Development: What to Choose? 2021. <https://www.valuecoders.com/blog/technology-and-apps/unreal-engine-vs-unity-3d-games-development/>
12. Marko Strizic. Top reasons for app store rejection. 2021. <https://decode.agency/article/app-store-rejection/>

13. McWherter, Jeff; Gowel, Scott. Professional Mobile Application Development. Wrox; 1st edition, 2012. ISBN 978-1118203903
14. Mediakix. THE MOBILE GAMING INDUSTRY: STATISTICS, REVENUE, DEMOGRAPHICS, MORE [INFOGRAPHIC]. 2020.  
<https://mediakix.com/blog/mobile-gaming-industry-statistics-market-revenue/>
15. Mukherjea Sougata. Mobile Application Development, Usability, and Security. IGI Global; 1st edition, 2016. ISBN 978-1522509455
16. Priya Viswanathan. iOS App Store vs. Google Play Store. 2020.  
<https://www.lifewire.com/ios-app-store-vs-google-play-store-for-app-developers-2373130>
17. Renana Dar. Top 7 Gaming Engines You Should Consider for 2021. 2021.  
<https://www.incredibuild.com/blog/top-7-gaming-engines-you-should-consider-for-2020>
18. Sarah Perez. Smartphone owners are using 9 apps per day, 30 per month. 2017.  
<https://techcrunch.com/2017/05/04/report-smartphone-owners-are-using-9-apps-per-day-30-per-month/>
19. Satinder Singh. Native vs Hybrid vs Cross Platform – What to Choose in 2022? 2021. <https://www.netsolutions.com/insights/native-vs-hybrid-vs-cross-platform/>
20. Statista Research Department. Revenue of mobile apps worldwide 2017-2025. 2021. <https://www.statista.com/forecasts/1262892/mobile-app-revenue-worldwide-by-segment>
21. Unity. Unity Announces Fourth Quarter and Full Year 2020 Financial Results. 2021.  
<https://investors.unity.com/news/news-details/2021/Unity-Announces-Fourth-Quarter-and-Full-Year-2020-Financial-Results/default.aspx>
22. Figure 1 - App Store Revenue Share by Category.  
Source: <https://www.blog.udonis.co>
23. Figure 2 - Market share of mobile operating systems in Czech Republic from 2010 to 2020. Source: <https://www.statista.com>
24. Figure 3 - App Store vs Google Play Store page layouts. Source:  
<https://yalantis.com>
25. Figure 4 - Number of applications available in leading app stores as of 1st quarter 2021. Source: <https://www.statista.com>

26. Figure 5 - Soft Development Life cycle principles. Source: <https://sumatosoft.com>
27. Figure 6 - Mobile Internet: average daily time spent in the US, App vs Browser.  
Source: <https://www.emarketer.com>
28. Figure 7 – Unity plans and pricing. Source: <https://store.unity.com>
29. Figure 23 – Creating an application on Google Play.  
Source: <https://play.google.com/console>
30. Figure 24 – Tasks for setting up the app. Source: <https://play.google.com/console>
31. Figure 25 – App category and tags. Source: <https://play.google.com/console>
32. Figure 26 – Application test countries and release creation.  
Source: <https://play.google.com/console>
33. Figure 27 – Production release review state.  
Source: <https://play.google.com/console>
34. Figure 28 – Tiles Drop page in the Google Play Store.  
Source: <https://play.google.com/console>