SAVONIA

Game Development: Sci-Fi Endless Runner

Dmitry Boronin

Bachelor's Thesis

___. ___. _____ _____

Bachelor's degree (UAS)

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Dmitry Boronin			
Title of Thesis Game Development: Sci-Fi Endless Runner			
Date	3 February 2017	Pages/Appendices	37
Supervisor(s) Mr. Mikko Pääkkönen, Software Engineer, Mr. Arto Toppinen, Principal Lecturer			
Client Organisation/Partners			

Abstract

Game development is the one of the most popular fields in IT, which is responsible for creating games. Many people like playing games on different platforms such as PC, consoles. Therefore, this point is very profitable for game developers.

The purposes of this project were to learn basic things about game development, game engines and to make a game with the original design.

The tutorials and instructions were used to get the necessary information about game development, for instance, Unreal Engine docs. Unreal Engine 4 was used as a game engine to implement the game.

As a result, the endless runner game was made – one of the popular genre of games nowadays. The game is not done completely; there are some features, which should be added to make the game more interesting and exciting.

Keywords
Game Development, Unreal Engine 4

CONTENTS

1 INTRODUCTION

1.1 Lead-in and history

The idea to create a game came up, when the main advantages of the game engines were read and learnt during the last summer holiday. On the one hand, the game development is one of the most interesting and exciting fields of IT, however, it is clear that the difficulties are expected in this area. Nevertheless, the game development was chosen as the topic of the thesis.

The history of game industry started in 1950. Approximately, in 1970 – 1980 video games became popular when arcade game machines, home video consoles, together with first computers were getting available to the average consumer. (The history of video games.)

Nowadays, video games are one of the most popular types of entertainment and they are a huge part of world culture. In addition, game industry is the one of profitable spheres in the world and it is developing incessantly and getting more and more people year by year. The Magnavox Odyssey is the first commercially successful home video game console, which was patented in 1968 and was released in 1972. Odyssey is a child of Ralph Henry Baer and it was developed together with engineers such as Bill Harrison and Bill Rusch. There are 350,000 units of this home video console sold. (The history of video games.)

The game PONG is one of the first arcade video games and a popular video game. Atari Inc. developed the original game. The company released this game in 1972. Allan Alcorn has created PONG for training skills, which were suggested to him by a founder of Atari Nolan Bushnell company. Comparing to previous released games, Pong was expected to get an incredible success and acceptance of gamers. (The history of video games.)

The “golden age” (1973-1986) of arcade video games is considered to be the peak of popularization and technical development of arcade video games. There were developers and engineers, who tried to present people more advanced games at that time. (The history of video games.)

1.2 Game genres

There is a variety of game genres and subgenres, which were coming out during development of game industry. The main genres are listed below: (The history of video games.)

- Action: it is a genre that challenges physical skills of the player such as reaction speed, eye coordination
- Action – adventure: it requires not only reaction from the player, as ordinary action game, also this genre consists of an interesting story, characters, inventory system and dialogs and other features of this genre
- Adventure: this genre focuses on story and suggests the player to pass the character through the enthralling and exciting story
- Role-playing game: this genre allows the player to control one or several heroes, which he will lead to victory, accomplishing the variety of quests, based around the main storyline
- Online games: online games allow to combine hundred players at the same time, each of them can focus on one's task and, in any case, to have influence on the progress of the game world
- Strategies: it is a special genre, requiring developed strategic thinking and possibility to plan one's tactic for several steps forward for achieving the aim
- Horror: horror games are focused on the survival of the player under continuous danger. The stress is achieved by the weakening of the player, compared to ordinary action game, and also by a major lack of resources, health and super powers
- Simulators: the genre describes a very wide category of video games, where the main feature is an attempt to simulate aspects of a reality via screen or other devices

1.3 Evolution of home game consoles

1.3.1 8 bit, 16 bit games

In 1985, Nintendo released an 8-bit game console called Famicom, known as Nintendo Entertainment System (NES) in Asia, thereby the American game console market got a new impetus in development. The console was defined as the entertainment system by Nintendo. (The history of video games.)

FIGURE 1. Sega Genesis, a 16-bit home game console by Sega (The history of video games)

Sega Genesis (FIGURE 1.), known as Mega Drive outside of North America, is a 16-bit game console, which was developed by Sega Enterprises, Ltd. It was the third game console in series of the company's consoles. Sega was released in 1988 in Japan. (Szczepaniak, 2006.)

In 1990, Nintendo in Japan released a 16-bit game console Super Nintendo Entertainment System or SNES (FIGURE 2.) It became one of the best-selling game consoles in the world, in spite of late start and fierce competition from the North American and Europe Sega's game consoles. (The history of video games.)

FIGURE 2. SNES, home game console by Nintendo (The history of video games)

Also, a technology development affected on game industry. During the times, developers had more opportunities for making games. However, it was just the beginning of graphics development. (FIGURE 3.)

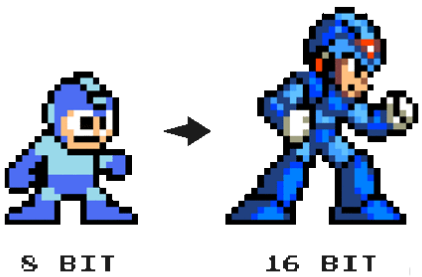

FIGURE 3. Technology development of games.
From 8 bit to 16 bit (The history of video games)

1.3.2 32 bits, 64 bit games

PlayStation or PS is 5th generation home game console, developed by Sony Computer Entertainment under direction of Ken Kutaragi. The console was released on December 3, 1994 in Japan. A more than 100 million of them were sold, so PS has become very popular. (The history of video games.)

Nintendo 64 – 64 bits' home game console. (FIGURE 4.) Nintendo developed it in collaboration with Silicon Graphics. It was released in 1996 and became a competitor between Sony PlayStation and Sega Saturn. (The history of video games; Drake, 2011.)

FIGURE 4. Nintendo 64, 64 - bit home game console
(The history of video games)

1.3.3 2000s

In 2000, PlayStation 2 (PS 2) was released and later in 2001, Rockstar Limited released their first 3D game, which is called Grand Theft Auto III (Gamespot Staff, 2000). It became available on PS 2 in the same year (Gamefaqs.com).

Grand Theft Auto III (GTA III) is an active-adventure genre game, which includes elements of the third person view and auto simulator in wide and open game world. Comparing with previous versions of the game, RenderWare 3D engine was used in developing GTA III. (Walton, 2015.)

1.4 Nowadays, Virtual Reality

Obviously, games are becoming more realistic, more powerful due to development of game technologies; more games appear year by year. Game industry has improved thus, that we can also play virtual reality games, using, for example, special glasses. (FIGURE 5.)

Virtual reality (VR) is a world created by technical resources (objects and subjects), transferring to human through the sensations such as vision, hearing, smell, touch, etc. (Merriam-Webster). Application of VR can be various simulators as an example.

FIGURE 5. PlayStation VR headset by Sony
(The history of video games)

2 GAME DEVELOPMENT

2.1 Description

Game development is a process of the making computer games. Generally, one person or the company develops the video game and the company consists of teams of developers. Usually, the commercial games are created by the developer teams and one company hires these teams. The companies can make the games for home game consoles, personal computers or tablets.

Frequently, the game development is sponsored by a larger company, which is called a publisher. The publisher is responsible for a game copy distribution and takes all costs related to it. There is a type of the games which a company distributes without a publisher, using, for instance, digital distribution resources, which is called an indie game.

2.2 Specializations

In the beginning of 1980s when the first home personal computer and home game console were made, one programmer could manage almost all tasks, related to game development. Modern game development is supposed to have a wide range of skills and support. Nowadays, a typical game development team usually consists of different members of various specializations such as:

- one or several producers for production monitoring
- designers
- artists
- programmers
- level designers
- sound engineers (composer, sound effects maker)
- tester

2.3 Project documentation

Since developer teams develop games, it is necessary to document the process of development. The main documents are described below:

- Concept document: it consists of 2-6 pages of text with illustrations, which show general features of the game. This document shows which resources are required to develop a game
- Game design document: it helps to organize efforts within a development team
- Proposal document: it is a short description of the game without detailed development, which should explain to the investor, why this game is profitable
- Technical design document: it describes technical requirements for the game. For example, database using, memory, fps, portability (Akhmedov).

2.4 Development process

Usually, the development process includes pre-production, production, milestones and post-production. Frequently, on the pre-production stage a design document should be written by the game designer. A concept and gameplay should be described in that document. On the production stage, a huge amount of work is done by programmers, artists, sound engineers, composers and level designers.

Testers start to work when there is something to test in the game, for example a part of game scene. Nowadays, testing is very important for the games, because due to complexity of most of them, even one single change can lead to catastrophic consequences. (Tielkes, 2016.)

2.5 Game Engines

A game engine is a set of systems, which simplifies the most frequently used functions of the game. The engine makes it easier to develop games and can make them portable. In addition, the code is more organized and managed. Moreover, the game engine allows working abstractly, i.e. the programmer does not need to deal with low-level things and to know how they work. There are game engines such as Unity, Unreal Engine, CryEngine, etc. Unreal Engine 4 was chosen to develop the project.

Comparing Unreal Engine with other engines, Luis Cataldi (Educational Evangelist for Epic Games) said, that this engine is a unique professional tool with open source, which is opened for public and allows reducing the development team up to two members. The engine is designed for huge studios and for small teams of indie developers such as Rocket League, for instance. (Mobio, 2016.)

Nowadays the engine can be used not only for first person shooters, but also for the other genres. People are developing different types of games; therefore, Unreal Engine can be applied in various fields. For example, Unreal Engine can be used by programmers, who do architectural visualization, develop virtual study courses for medics, also for people, who make scenes or even whole movies. (Mobio, 2016.)

3 UNREAL ENGINE 4

3.1 Overview

Unreal Engine is a game engine developed and maintained by Epic Games. The first game, which was created on this engine, is Unreal in 1998. (Moddb.com.)

This engine is made with the C++ programming language. Unreal Engine allows creating games for the most operating systems and platforms: (Unrealengine.com.)

- Microsoft Windows
- Linux
- Mac OS and Mac OS X;
- Xbox, Xbox 360, Xbox One
- PlayStation 2, PlayStation 3, PlayStation 4
- PSP, PS Vita
- Wii
- Dreamcast
- GameCube
- For various portable devices such as Apple devices (IPad, IPhone), Android

Unreal Engine supports different rendering systems (Direct3D, OpenGL, Pixomatic), sound playback (EAX, OpenAL, DirectSound3D), network and input modules. Rendering is a 3D object formation in the form of a bitmap which is done by the special software.

For the online games Windows Live, Xbox Live, GameSpy technologies and others are supported, including 64 players (clients) at the same time. So, the engine is adopted for the MMORPG games (LineAge II).

Unreal Engine games are made in various genres, however, first projects were created, generally, as shooter or action genres; flexibility of technologies allows creating strategies, quests, simulators, etc. Moreover, the engine can be applied for working with graphics in cinematography, for instance, creating special effects and for educational purposes.

3.2 Unreal Engine Versions

3.2.1 Unreal Engine 1

Unreal Engine 1 was developed in 1998. It was one of the first unique game engines; it included a graphics engine, physics engine, artificial intelligence (AI), file and network system management, and a ready development environment for games – UnrealEd. Several engine's features were extremely new for that time, for example, the using of dynamic scene graph. (Zero_Cool.)

A scene graph is a data structure, which is used mainly in vector graphics editors and computer games. Examples of such programs are Acrobat 3D, Adobe Illustrator, AutoCAD, CorelDRAW, OpenSceneGraph, VRML97 and X3D. This technology allowed using a number of effects to overlay the surface:

- Partly or completely mirror surfaces
- Warp technology: it is an ability to change an image of one surface image projection to another, i.e. parallel projection during the drawing
- Skybox: it is a projection rendering on the surface from another point, which usually placed to small "box" with overlaid sky texture

The engine has become one of the first engines, where there was halo effect around light sources. These light sources were smoothly fading and when the player moved, they were overlapped by wall edges. (Zero_Cool.)

Initially, the engine has been released for Windows and Macintosh. Due to the module engine system, there was the ability to port the engine to the "new generation" consoles of that time. Later, it has been successfully used on the platforms such as GameCube, PlayStation 2 and Xbox. (Zero_Cool.)

3.2.2 Unreal Engine 1.5

In 1999, an updated version of Unreal Engine was released for modern (in those times) computers and game consoles Dreamcast and PlayStation 2. There were significant updates: (Zero_Cool.)

- characters' facial animation support
- a new texture resolution (1024x1024)
- added S3TC technology
- 2nd version of UnrealEd was integrated

S3TC (S3 Texture Compression) is a method of DirectX texture compression. This engine version was used in games such as “Unreal Tournament” and “Harry Potter and Philosopher’s Stone”. (Zero_Cool.)

3.3 Unreal Engine 2

The second version of Unreal Engine was released in 2002 together with Unreal Tournament 2003, America’s Army: Operations and Unreal Championship games. The core and rendering mechanism were almost completely rewritten. In addition, 3rd version of UnrealEd was integrated; a physics subsystem Karma, which is supporting ragdoll physics, was also integrated. (Zero_Cool.)

The new technologies added (Zero_Cool.):

- fluid surface: it is a flat object, which consists of large numbers of polygons, imitating liquid surface
- foliage: it is generating objects, which decorate landscape, for example, grass
- VoIP support: the possibility of chatting via microphone with other players during the game
- Speech recognition: it is a voice translation into the text and processing as a command, for example, an ability to give commands to bots. This technology was used by Microsoft Speech API and that’s why it was supported only in 32-bit version of Windows
- “Karma” technology: processing of the body movements as a ragdoll with the parts of the body
- Ragdoll physics
- Vehicles: ability to process events from sided actor. Vehicles divided onto types of realization – static (stationary defensive sets), composed of several objects (Karma vehicle), using a skeletal structure and vehicles attached to other vehicles (for example, machine gun on tank tower). Some vehicles can operate without “pilot”, controlling by its own artificial intelligence
- EAX 3.0 – 3D sound engine, developed by Creative Labs

3.3.1 Unreal Engine 2.5

In this version, the graphics engine was improved and optimized. There was a support for rendering technologies such as Direct3D 9, OpenGL 2 and Pixomatic. It was added support of 64-bit operating systems Windows NT and GNU/Linux. The most available texture resolution was upped to 4096x4096 pixels. Added a full support of Unicode (16-bit) that allows creating full localized games in Asian languages. (Zero_Cool.)

Moreover, the engine itself has been significantly optimized, it had a better performance for the same system requirements. There was the possibility of video playback in DivX and Bink formats. In later versions, the trees creating system was integrated (SpeedTree). This version of the engine has been used in such computer games as Unreal Tournament 2004, Pariah, Killing Floor, and many others. (Zero_Cool.)

3.3.2 Unreal Engine 2 Runtime

Unreal Engine 2 Runtime is a special version of Unreal Engine 2.0 with limited license. Later this version of engine was changed onto UDK (Unreal Development Kit) – a cheapened version of Unreal Engine 3. (Zero_Cool.)

A user could download Windows engine distribution from official website (it is required to buy license for others operating systems). There were map editor UnrealEd, ucc utility, test level and a small set of models and textures in the kit, demonstrating technology abilities. The engine was free for non-commercial projects and for educational purposes such as creating 3D presentations. It required buying a license for commercial purposes. (Zero_Cool.)

3.3.3 Unreal Engine 2X

This is a special engine version for Xbox game console based on Unreal Engine 2.0. Besides engine code optimization, it was added new visual effects such as depth of field, dynamical gamma-correction, bloom and other variations of blur. (Zero_Cool.)

Texture format has been changed for more realistic displaying of shades in a high resolution. Memory Tracking, together with voice chat support, Xbox Live service and split screen function were added. This version of engine was used only in Unreal Championship 2 game. (Zero_Cool.)

3.4 Unreal Engine 3

The version was developed considering personal computers, which are using modern rendering systems (DirectX 9/10 and OpenGL 2/3), and consoles of next generation (PlayStation 3 and Xbox 360). Due to wide spreading of multiprocessor systems, the engine uses two parallel main threads – general thread (responsible for game process) and rendering thread. In addition, secondary threads can be called for implementing single tasks. (Zero_Cool.)

The updated graphics engine supports many modern technologies, including HDR, per-pixel lighting, dynamic shades, and fourth version of shader model and geometry shaders. The physics subsystem Karma was replaced onto PhysX, developed by AGEIA.
The integrated animation engine FaceFX, developed by OC3 Entertainment, is responsible for face animation. The EAX version was updated to the fifth one. Added SpeedTree technology support for tree generation. There is new UnrealEd editor, which is rewritten using wxWidgets. (Zero_Cool.)

Despite the Unreal Engine 3 (UE3) is opened for modification creating; only license owners were able to sell games based on UE3. However, Epic Games released free version based on UE3 in 2009, which was called Unreal Development Kit (UDK) and was available for every novice developer. In December 2010, there was released UDK version, supporting game development on iOS platform. (Zero_Cool.)

In 3.5 version, there were next evolution of graphics engine. Ambient occlusion filter has been added, which is improving shades and lighting. A number of processed characters has been increased. In 2010, a preview of new engine possibilities has been published. (Zero_Cool.)

3.5 Unreal Engine 4

The vice president of Epic Games Mark Rein announced on 18th of August 2005, that Unreal Engine 4 is already 2 years in developing and its target platforms are personal computers and consoles of 8th generation and Tim Suini is the only one man who is working on engine. A technical demonstration of Unreal Engine 4 was held at E3 2012. (Zero_Cool.) E3 or Electronic Entertainment Expo is an annual exhibition of computer games industry, conducted by Entertainment Software Association (ESA) (The free dictionary by Farlex).

In 19th of March 2014, Unreal Engine 4 has started free distribution with subscription 19$ per month. Source codes also have been uploaded onto GitHub repository. In 2nd of March 2015, Epic Games announced that now engine is free for all developers with one condition, if profit from applications, created under this engine, exceed 3000$ per quarter, then developers have to pay 5% from profit to Epic Games. (Zero_Cool.)

4 UNREAL ENGINE TERMINOLOGY

4.1 Projects, Objects, Classes

A project is an autonomous block, which presents a single game by itself. From the viewpoint of the file system, it is a folder with subfolders and files for work. All information about the project is kept in file with the extension .uproject. At the same time any amount of projects can be managed. Objects are general “building blocks” of Unreal Engine. Whole functionality is made via objects (Unrealengine.com.)

Classes describes behavior and properties of particular Objects and Actors, which are used in the game. Classes are organized hierarchically, i.e. child classes inherit or override properties and behavior of parent class. Classes are described by either C++ code or Blueprints. (Unrealengine.com.) The Blueprints (FIGURE 6.) is a visual script language, which allows writing a game logic without program language application. It is a powerful tool, which can be applied for creating characters, levels, models, etc. (Blintsov, 2015).

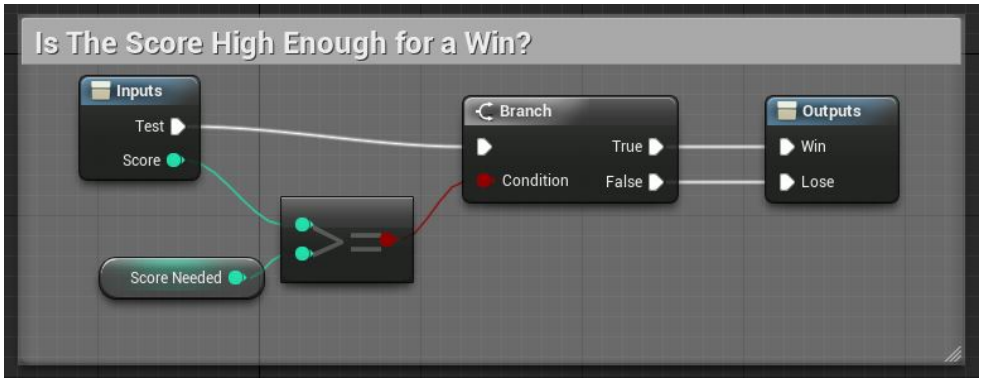

FIGURE 6. Example of the blueprint (Unreal Engine docs)

4.2 Actors, Components, Pawns

Actor (FIGURE 7.) is any object which can be located into 3D scene (level). It is common class which support 3D transformations such as relocation, rotation, scale. An actor can be created and deleted by code (C++ or Blueprints). In C++, all Actors’ base class is AActor. (Unrealengine.com.)

Components are some functional, which can be added to an Actor. They cannot be existing by themselves, without reference to Actor. (Unrealengine.com.)

Pawns (FIGURE.8) are subclass of Actors. They represent a game character and can be controlled either by the player or Artificial Intelligence (AI). In AI case they represent non-player characters (NPC). There are two types of pawns - possessed, which are controlled by player or AI, and unpossessed – no one controls them. (Unrealengine.com.)

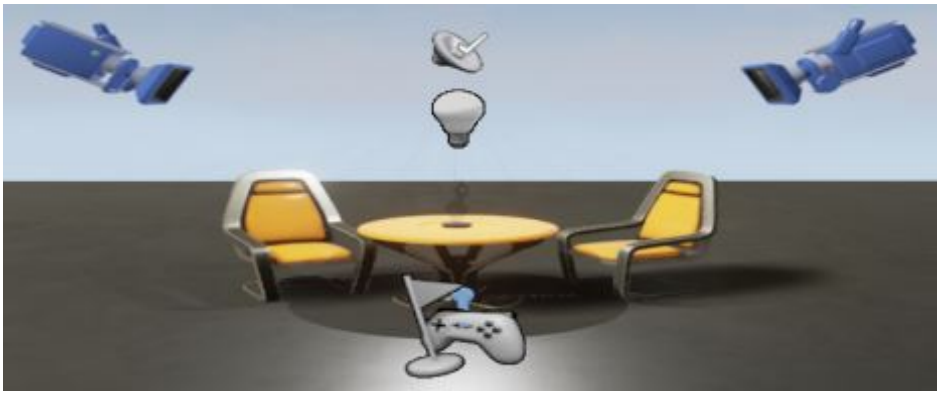

FIGURE 7. Actors (Unrealengine.com)

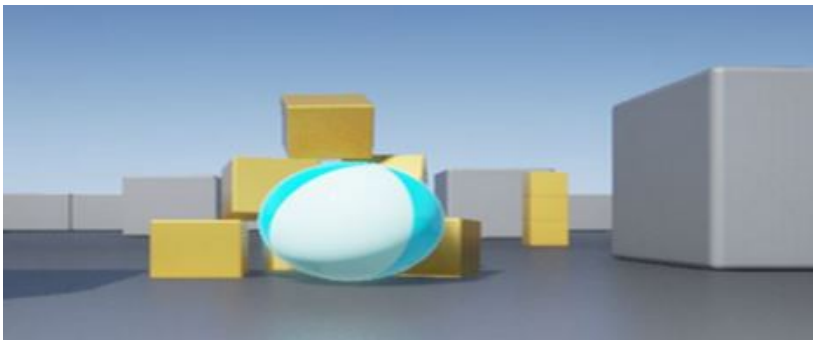

FIGURE 8. Pawns (Unrealengine.com)

4.3 Characters, Player Controller, AIController

Character (FIGURE 9.) is a subclass of Pawns. It is used as a person by a player. It has the same properties such as collision setup, settings for bipedal movements and additional code to manage movements, which are controlled by the player. (Unrealengine.com.)

The PlayerController is a class, which receives the input data from the player and transforms them to the game actions. Each game has at least one instance of this class. The PlayerController controls Pawn or Character often as a representation of a player in the game. This class has a network interaction point for multiplayer games. Moreover, during online games there is one controller for each player and every client has only one controller of the player, corresponding to it. (Unrealengine.com.)

AIController (FIGURE 10.) is a class, which controls the Pawn, representing NPC by itself. By default, all Pawns and Characters have basic AI controller. (Unrealengine.com.)

FIGURE 9. Character (Unrealengine.com)

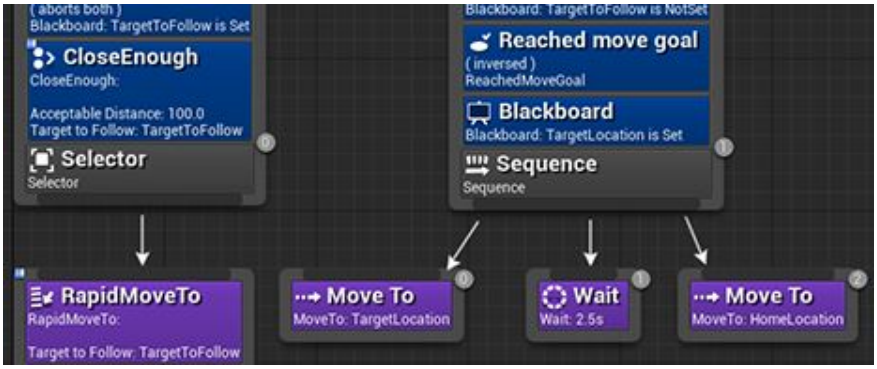

FIGURE 10. Full Tree (Unrealengine.com)

4.4 Brushes (BSP), Levels, World

Brush (FIGURE 11.) is an actor, which describes a 3D-zone of the space in the form of parallelepiped. It is used for prototyping levels, usually, for gameplay testing. The brush locates in the level space and defines the properties of its space. There are blocking volumes (invisible, which do not allow actors to go through them), pain causing volumes (any actor, who is caught to this volume, will get repeating damage) and trigger volumes (call some events, when character enters or leaves these volumes). (Unrealengine.com.)

FIGURE 11. Geometry Brush (Unrealengine.com)

Level (FIGURE 12.) is an area of gameplay, which is defined by the player. In general, it consists of Actors. Each level is saved in .umap file, so, sometimes they are called maps. (Unrealengine.com.)

FIGURE 12. Level (Unrealengine.com)

A World (FIGURE 13.) consists of Levels list that are downloaded into the game. It controls Level changing and dynamically created Actors. Direct interaction with World is not necessary, but it helps to define reference points. (Unrealengine.com.)

FIGURE 13. Levels Window (Unrealengine.com)

4.5 GameModes, GameStates, PlayerStates

GameMode (FIGURE 14.) is a class, defining rules of the current game, including how players are connecting to the game; order and rules of levels changing; victory conditions, etc. The game mode “by default” is setting in Project Settings, but it can be redefined for each specific Level. For each Level there is only one instance of GameMode. In a multiplayer game, GameMode is set only on the server, his rules are sent to all clients. (Unrealengine.com.)

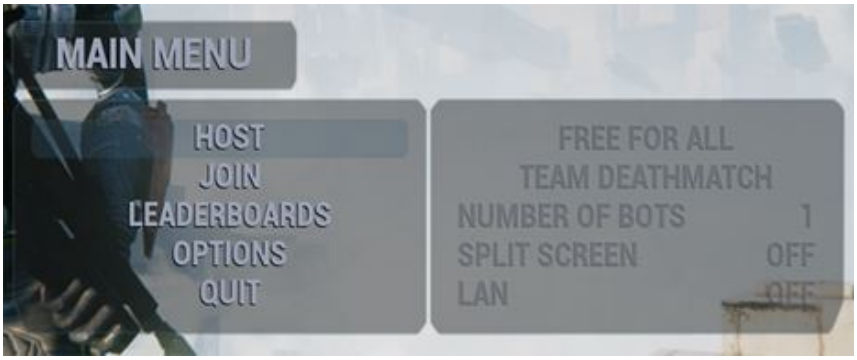

FIGURE 14. GameMode lander (Unrealengine.com)

GameStates (FIGURE 15.) contains currents settings of the game, which are being send to all connected clients, for example: information about score; is match started, how many AI actors are created, etc. For multiplayer games there is only one exemplar of GameStates on each client's machine. (Unrealengine.com.)

FIGURE 15. GameState (Unrealengine.com)

PlayerState (FIGURE 16.) is a state of human – player or bot, who is imitating real player, including such parameters as player name, number of points, health level, etc. AI character, representing an object of the world, does not have PlayerStates. In multiplayer games, all PlayerStates exist on every machine for all users and can duplicate server data for common synchronization. (Unrealengine.com.)

FIGURE 16. PlayerState (Unrealengine.com)

5 SCI-FI ENDLESS RUNNER

5.1 Game description

The game was made on Unreal Engine 4. It is a game for mobile platforms, such as Android, iOS and Windows Phone. The game has been done and tested only on PC. It is not finished yet, because there are some features that should be still added. At this moment, about 80% percent of the project has been done. Tutorials have been read and used from Unreal Engine's official website during the game development.

A type of the game is an endless runner. It means that there is a character, who is running somewhere continuously. For instance, there is a popular game "Subway Surf" (FIGURE 17.). The main idea of the game is that there is a person, who looks like Tron's (FIGURE 18.) movie character, and he is running on shiny cosmic road in outer space. He has to run as long as he can, overcoming obstacles such as corners, meteorites and precipices. The velocity of the runner is accelerating second by second, so it becomes more difficult to play. If the character falls down, hits a wall or meteorite, the game is over. The player can collect "perks" or in other words super powers during the game. Super powers are:

- Magnet for the rings: it allows to player collects the rings, because of magnetism. It works for 10 seconds
- Bomber: it allows to the player to run through the meteorites, so meteorites explode when the player bumps into them. It works for 5 seconds
- Time slow (not finished yet): it allows to the player to slow the time. It works for 10 seconds.

FIGURE 17. Screenshot of Subway Surf (Gametop.com)

FIGURE 18. Tron: Legacy movie (Juan Carlos Cuadra)

In addition, the player can collect cosmic rings while running and upgrade his runner. For example, to buy a new costume or extend time of each perk.

5.2 Process of the developing the game

Firstly, the Unreal Engine 4 software was downloaded, which is available from Epic Games official website. An account was created and a development environment was downloaded via this account (FIGURE 19.). The development was started with Unreal version 4.11; however, it was not updated during development, because the newer version of the engine can be incompatible with the project, which is done on the old engine version. At this moment, current version of engine is 4.14.

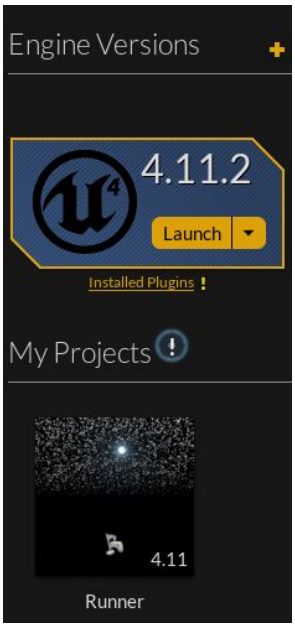

FIGURE 19. Engine version

5.2.1 Player Control

When the downloading was done, a new project was created with a Third Person blueprint, which is available for the developing. In this type of project there is just a character, which can run any direction at some location, walk, jump as in a standard third person shooter.

The Endless runner means that a character has to run continuously forward with the fixed camera, i.e. the player cannot change a view in the game, using mouse, for instance, and cannot just change the direction to wherever he wants. Therefore, a movement of the person should be changed.

Opening the character blueprint class, which is called RunnerCharacter, there are three tabs: Viewport, Construction Script and Event Graph. In Construction Script, the movements settings can be found, which are represented as the blueprints. All unnecessary control's blueprints were deleted such as a gamepad and mouse control, touch input – because the game was tested on the PC only and there is no need of mouse; and in Viewport, the camera is set a little bit higher. Next step, the character should run continuously without the keyboard input and turn 90 degrees when the person reaches the corner, following blueprints were used. (FIGURE 20, 21.):

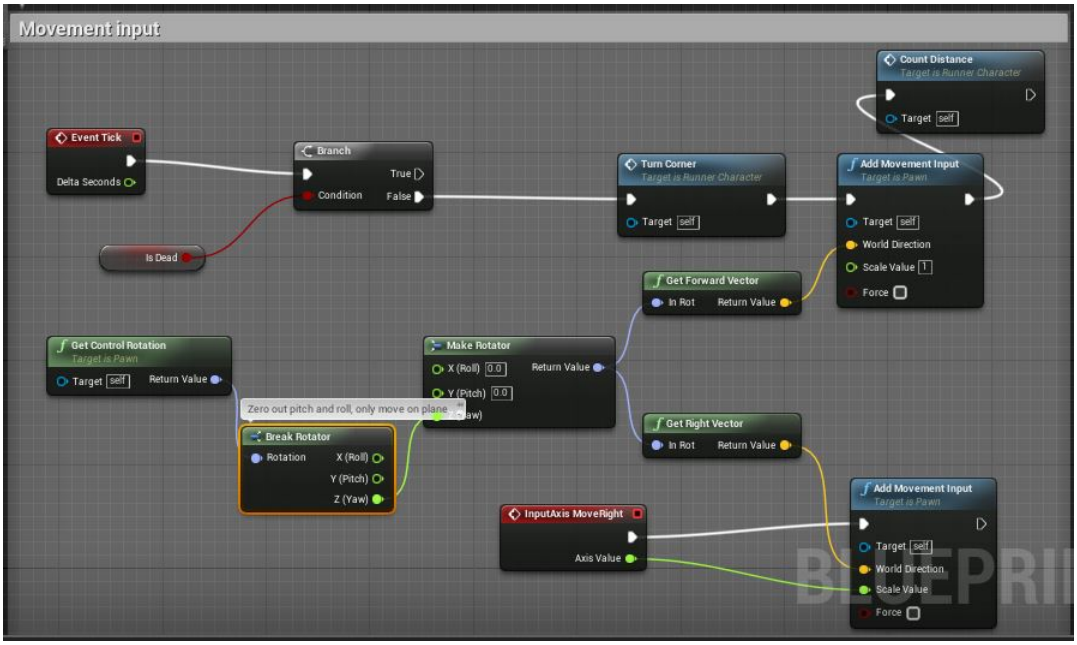

FIGURE 20. Blueprint for endless running

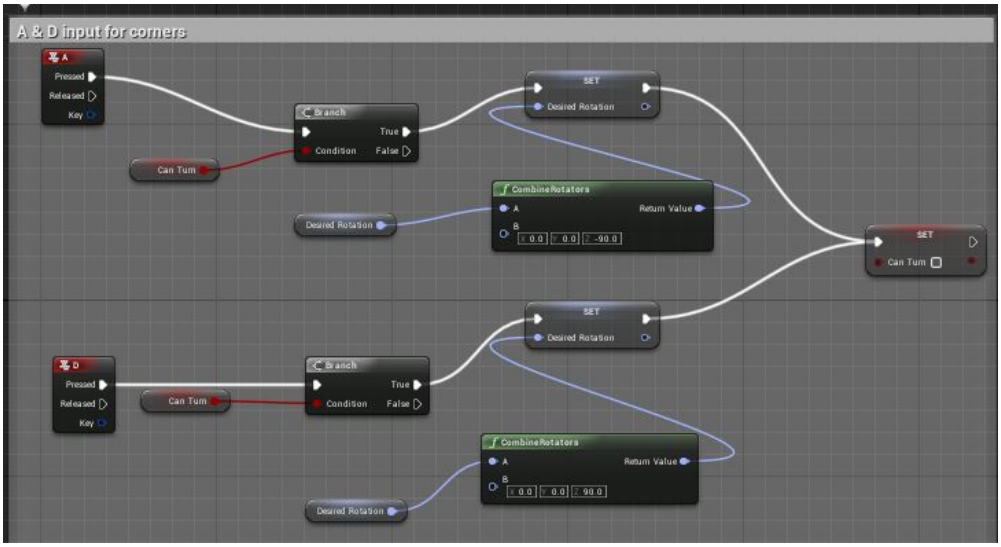

FIGURE 21. Blueprint for 90 degrees' rotation

5.2.2 Adding Floor Tiles

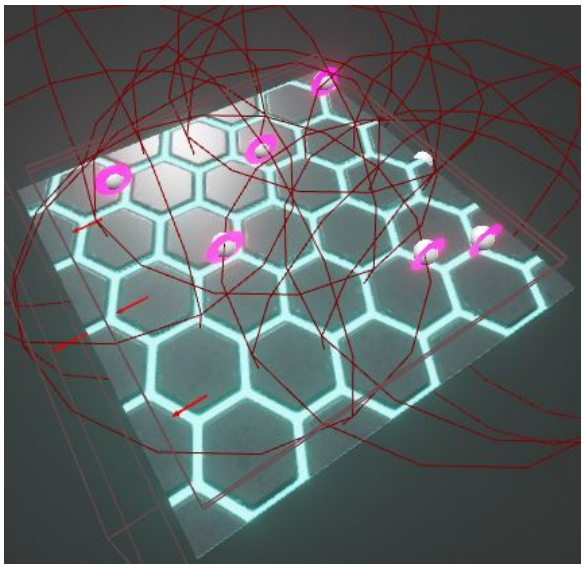

FIGURE 22. Floor tile model. Red lines are collision areas of each ring placed to the floor.

When movement setting was done, floor tiles should be made continuously, where character will run. A new blueprint class called BP_FloorTile was created, where three tabs can be found: Viewport, Construction Script and Event Graph. In Viewport, a model of floor tile was created and designed (FIGURE 22.) and then, a function was developed and applied to spawn those tiles (FIGURE 23.) and, finally, an event was made in Event Graph, which allows the function to be called:

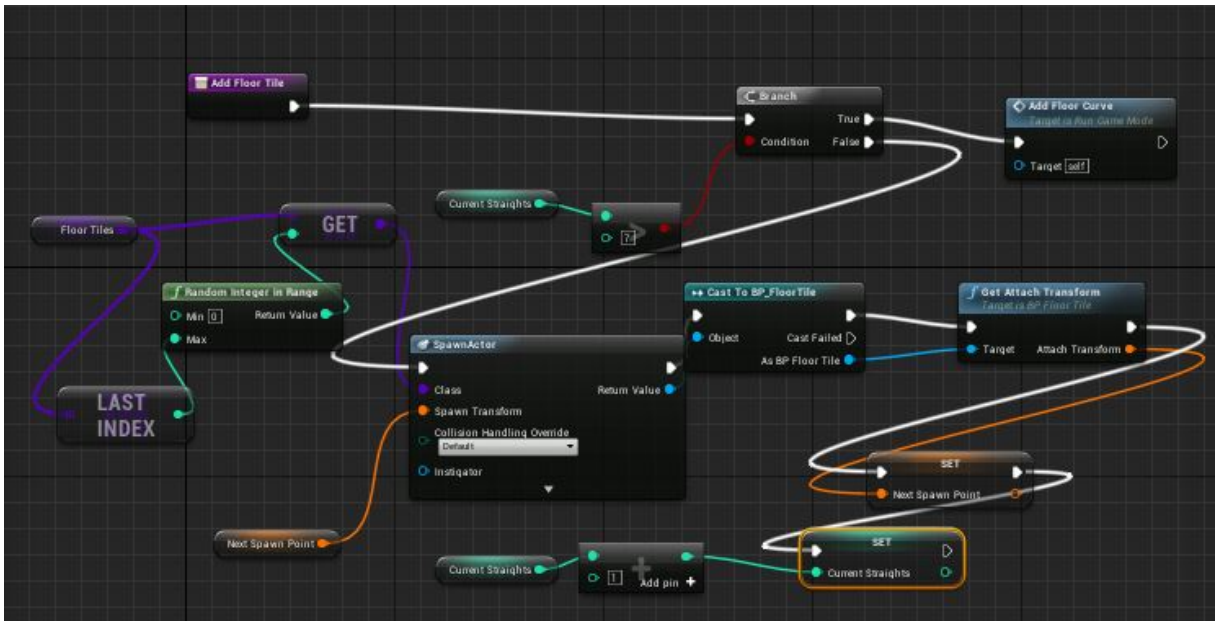

FIGURE 23. Blueprint of function for adding floor tiles

This function generates a couple of the floor tiles as the person runs ahead of himself. When he has run them the floor tiles are deleted behind him; they should be removed, otherwise, the game will crash due to memory leak.

5.2.3 Adding Obstacles and final result

As before, a new blueprint class was created, which is called BP_Blocker. In Viewport, a model of obstacle was made, which looks like a meteorite, but design is not finished yet (FIGURE 24.). In addition, a simple shader for stone was done and applied to it, so, meteorite is shining. Shaders are algorithms of drawing of shades, surfaces, etc.

After that, this meteorite was put on the floor and the function was made, which allows meteorite randomly appears on the left, middle or right side of the tile. In addition, it can be just a floor without meteorites.

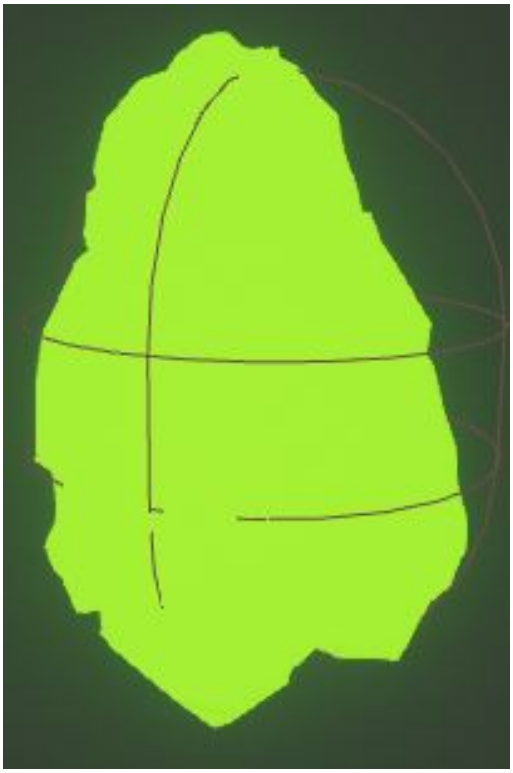

FIGURE 24. Meteorite's model

When the character bumps into obstacle, he can simply run through it. It is a problem, which should be solved. In the obstacle's settings a blocker was made on its meteorite that does not allow the runner go through the meteorite. Moreover, an event was created, where character dies and game is over, when there is collision between character and obstacle. Blueprints of collision (FIGURE 25.) and character's death (FIGURE 26.) are shown:

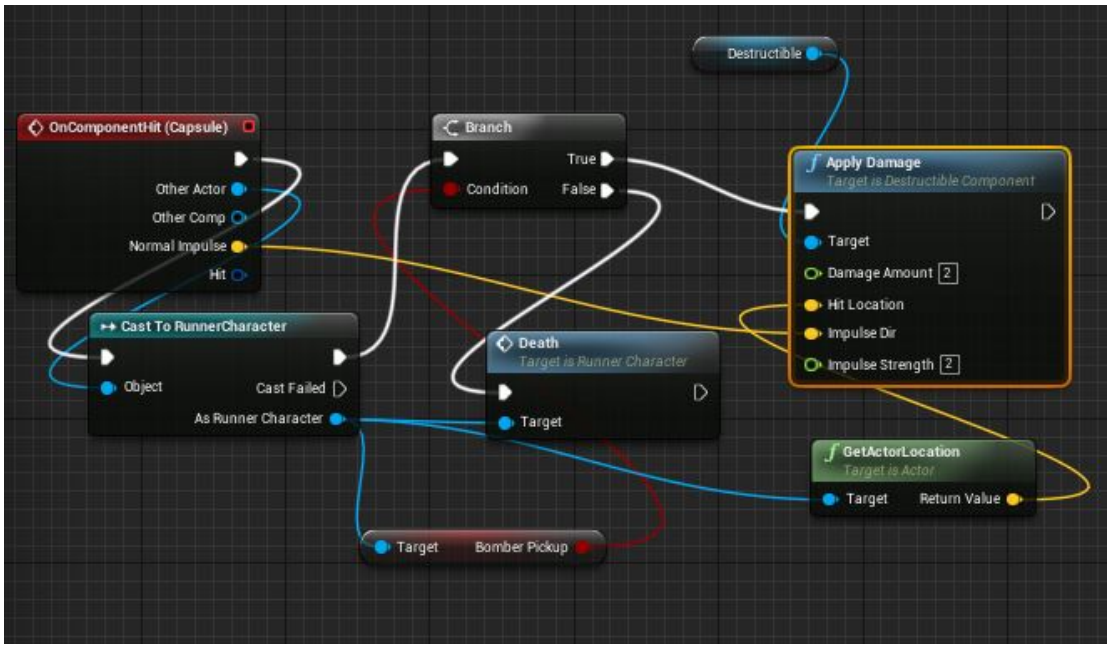

FIGURE 25. Blueprint for hitting the meteorite, which activates character death event

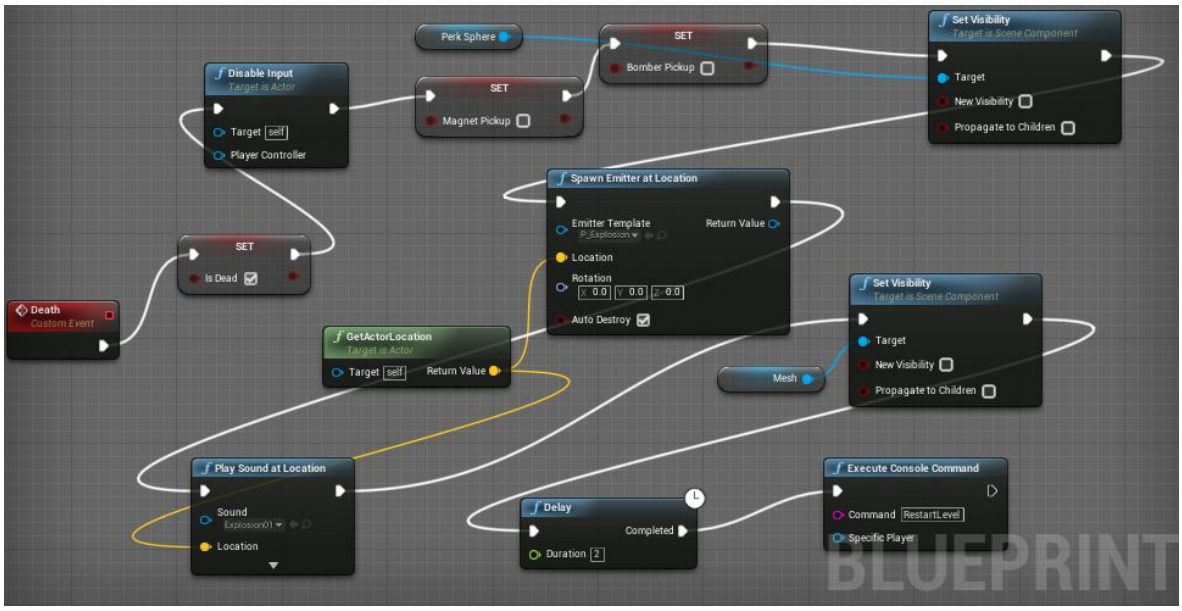

FIGURE 26. Blueprint for character death event

At this moment, the game looks like on the picture below (FIGURE 27.) To make game more interesting, a few features were added to the game such as turning corners (runner has to turn at the right moment either left or right), gaps (runner has to jump over them, otherwise, he will fall down and die), hills, distance and rings counters. Features that have not been done are background dynamics (such as comets, asteroids, etc.), force fields as

another obstacle, improving design and shader of meteorite, flat rings, time slower perk, more effects from perks.

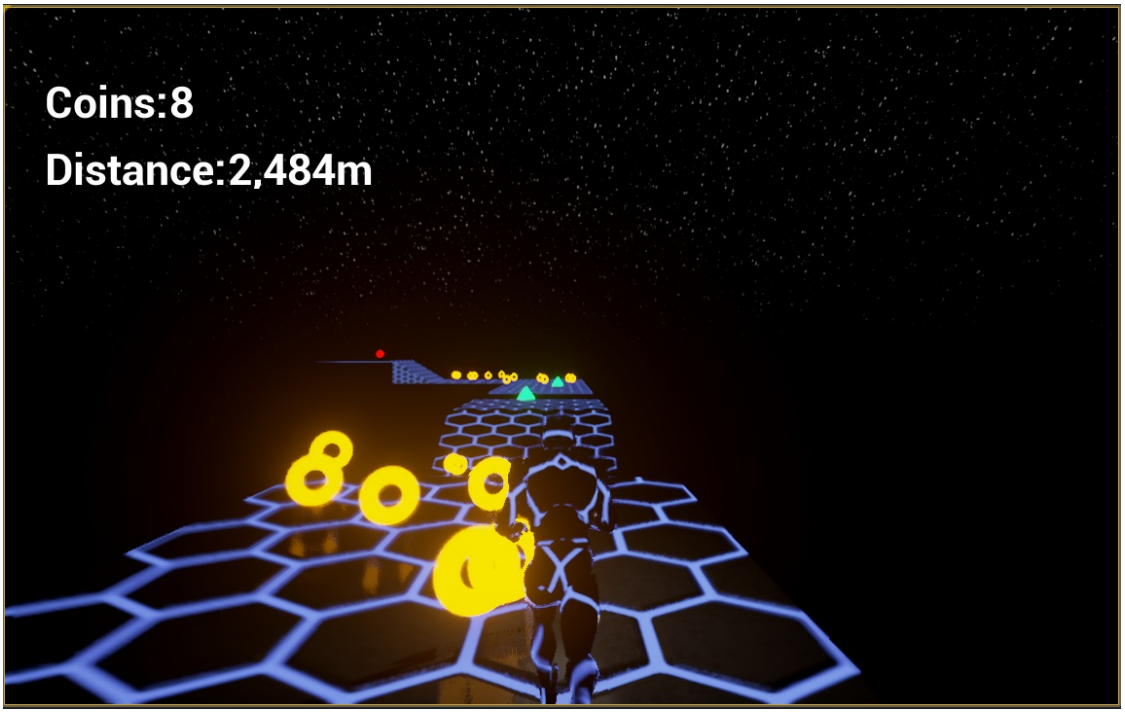

FIGURE 27. Project’s gameplay

6 CONCLUSION

As a result, an endless runner game was made under Unreal Engine 4. During game development, there were obstacles, which were overcome. The main problems were to make the game mode, i.e. continuously spawning and deleting the course, to make mortal collisions between the character and the course and to make meteorites spawn randomly on that course. Unreal Engine's documentation and tutorials were studied to solve the problems. In addition, there were obstacles such as making perks and shaders, which had to be applied to the objects.

However, there are features, such as flat rings, time slower perks, force fields that should be done soon to make the game more interesting and exciting. For instance, a force field is one more obstacle, which should be overcome by a character; the time slow perk allows the player to slow time down for 5 seconds, during the running.

Taking everything into account, a valuable experience and skills in game development were learnt especially in Unreal Engine management. I hope that the obtained knowledge will help me to find a suitable job in the IT sphere, where I will continue improving my skills and make more difficult, serious and exciting game projects.

REFERENCES

Akhmedov Anatoly, Technical Design Document: What, Why and How? [reference made: 31.01.2017]. Available at: http://www.dailytelefrag.com/articles/read.php?id=46398

Blintsov Alexander, Unreal Engine 4 for developing your abilities, 2015 [reference made: 20.01.2017]. Available at:
https://habrahabr.ru/post/249965

Cuadra Juan Carlos, Tron: Legacy on Behance's picture of a character [reference made: 30.01.2017]. Available at:
https://www.pinterest.com/pin/521643569316617254

Drake Jonathan, Nintendo 64: Launching a legacy, 2011 [reference made: 20.12.2016]. Available at:
http://www.ign.com/articles/2011/09/24/nintendo-64-launching-a-legacy

Gamefaqs.com, Grand Theft Auto III release date on PlayStation 2 [reference made: 20.12.2016]. Available at:
https://www.gamefaqs.com/ps2/466217-grand-theft-auto-iii/data

Gamespot Staff, Sony announces PS2 launch date and price, 2000 [reference made: 25.12.2016]. Available at:
http://www.gamespot.com/articles/sony-announces-ps2-launch-date-and-price/1100-2568701

Gametop.com, Subway Surf Screen screenshot [reference made: 30.01.2017]. Available at:
http://www.gametop.com/download-free/subway-surfers

History of video games' pictures [reference made: 30.01.2017]. Available at:
http://history-of-video-games.webflow.io

Merriam-Webster, Virtual Reality definition [reference made: 31.01.2017]. Available at:
https://www.merriam-webster.com/dictionary/virtual%20reality

Mobio, Interview with Luis Cataldi (Epic Games) about Unreal Engine, 2016 [reference made: 07.01.2017]. Available at:
https://habrahabr.ru/company/mobio/blog/316972

Moddb.com, Unreal Engine 1 [reference made: 21.12.2016]. Available at:
http://www.moddb.com/engines/unreal-engine-1

Szczepaniak John, Retroinspection: Mega Drive, 2006 [reference made: 12.12.2016]. Available at:
http://www.sega-16.com/2006/09/retroinspection-mega-drive

The free dictionary by Farlex, E3 Media and Business Submit [reference made: 20.01.2017]. Available at:
http://encyclopedia.thefreedictionary.com/Electronic+Entertainment+Expo

The History of video games [reference made: 10.12.2016]. Available at:
http://history-of-video-games.webflow.io

Tielkes Daniel, Why game testing is important, 2016 [reference made: 20.01.2017]. Available at:
https://www.spritecloud.com/2016/05/why-game-testing-is-important

Unreal Engine docs, Blueprint screenshot [reference made: 30.01.2017]. Available at:
https://docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/Macros

Unrealengine.com, Frequently asked questions (FAQ) [reference made: 21.12.2016]. Available at:
https://www.unrealengine.com/faq

Unrealengine.com, Unreal Engine Terminology [reference made: 20.01.2017]. Available at:
https://docs.unrealengine.com/latest/INT/GettingStarted/Terminology/index.html

Unrealengine.com, Unreal Engine Terminology's pictures [reference made: 30.01.2017]. Available at:
https://docs.unrealengine.com/latest/INT/GettingStarted/Terminology

Walton Steven, Grand Theft Auto V Benchmarked: Pushing PC Graphics to the Limit, 2015 [reference made: 27.01.2017]. Available at:
http://www.kotaku.com.au/2015/04/grand-theft-auto-vbenchmarked-pushing-pc-graphics-to-the-limit

Zero_Cool, History of technology - Unreal Engine [reference made: 21.12.2016]. Available at:
http://gamegpu.com/history/istoriya-tehnologii-unrealengine.html


SAVONIA