UNIVERSITY OF TARTU

FACULTY OF SOCIAL SCIENCES

NARVA COLLEGE

INFORMATION TECHNOLOGY SYSTEMS DEVELOPMENT

Vitali Bassov

DEVELOPMENT OF AN INTERACTIVE ENGLISH LEARNING MOBILE GAME FOR RUSSIAN SPEAKING USERS

Diploma Thesis

Supervisors: Assistant Andre Sääsk

Nadežda Furs-Nižnikova

NARVA 2018

Olen koostanud töö iseseisvalt. Kõik töö koostamisel kasutatud teiste autorite tööd, põhimõttelised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.
………………………………….. /töö autori allkiri/

CONTENTS

TERMS AND ABBREVIATIONS

Gamification – the practice of making activities more like games in order to make them more interesting or enjoyable. (Definition of “gamification” ... 2018)

E-learning – E-learning or "electronic learning" is an umbrella term that describes education using electronic devices and digital media. (E-learning 2015)

Rendering – the process involved in the generation of a two-dimensional or three-dimensional image from a model by means of application programs. (Rendering 2018)

Play Store – The market to download applications on the Android platform.

Asset – game assets are described as being any piece of data which: is in a format that can be used by the game engine; will be presented to the user. (Davis 2009) The asset can be a sound, texture, text file, etc.

NPC – A non-player character (NPC) describes characters the player interacts with throughout the game. (Non-Player Character... 2018)

HUD – Head-Up Display. In a computer game, the part of the screen that shows information such as the player's score, how much time they have left, etc. (Meaning of “HUD” ... 2018)

Spawn Point – is a point where a character appears.

Z-Order – an ordering of overlapping two-dimensional objects (Z-order 2017: 870)

CPU – The central processing unit (CPU) is the unit, which performs most of the processing inside a computer. (Central Processing Unit... 2018)

JSON – **J**ava**S**cript **O**bject **N**otation. JSON is a format for storing and transporting data. (JavaScript... 2018)

GUI – A graphical user interface (GUI) is an interface through which a user interacts. This interface uses icons, menus and other visual indicator (graphics) representations to display information and related user controls. (Graphical User Interface… 2018)

UI – User Interface. Most of the time it has essentially the same meaning as the GUI.

Key handler – a part of an application, which handles keyboard typing.

RAM - Random Access Memory (RAM) is the hardware in a computing device where the operating system (OS), application programs and data in current use are kept so they can be quickly reached by the device's processor. (Rouse 2018)

INTRODUCTION

One of the growing problems facing all formal learning, whether classroom, online, distance, or “e-,” is keeping students motivated – motivated enough to stick with the learning process to the end of a class, lesson, session, course, semester, or degree. Motivation is important because learning requires putting out effort. However, the things that were effective in motivating learners in past do not motivate the learners of today. We need something new. (Prensky 2005: 1)

The obvious candidate for this “something new” is technology. The tremendous technological advancement of the last decades provides us with many new possible ways and means of teaching.

Today most schools still mostly follow traditional ways of teaching, not fully implementing most of the modern technological opportunities or even mostly ignoring them. It even happens sometimes that a school installs some new technological equipment or implements an e-learning environment, but teachers do not actually use it.

At the same time, every day more and more aspects of life turn digital and school education should not be exception. E-learning is a new trend nowadays and it becomes increasingly popular, as e-learning is available in any place in the world where the connection to the global web is available. Unfortunately, our schools are mostly trailing behind this process yet.

One of the major uncharted and currently mostly overlooked potential e-learning opportunities is gamification of educational activities. People, especially children, like playing games (including video games), but the education system currently almost do not use that potential resource of motivation boost and engaging power. As Diane Ackerman once perfectly coined it “Play is our brain’s favourite way of learning things” (Ackerman 2000: 11).

The problem

One of the crucial problems of current education is that traditional methods of teaching are rapidly losing their effectiveness. Students refuse to study the old ways, but schools are too conservative to change quickly enough to reflect and embrace this new reality. Unsurprisingly, the level of motivation for studying is dramatically low among students, which leads to very low education efficiency.

The solution

We now have a generation that when growing up deeply experienced, for the first time in history, a radically new form of play – computer and video games. As this new form of entertainment has radically shaped their preferences and abilities, it has absorbed their time and effort to an extent never before seen. The typical student has now played thousands of hours of video games before graduating from college. The engagement power of electronic games for this generation (and those to come) may be, if used correctly, the biggest learning motivator we have ever seen. (Prensky 2005: 1-2)

Therefore, one of the obvious solutions to the problem stated above would be "Learn by playing". Video games can boost interest in the subject (studying a language in our case) and enhance motivation.

Aim

The aim of this diploma thesis is to develop a language learning game, which would increase a student's interest and motivation through the following features:

- Ability to complete levels, which should boost a player's confidence;
- Ability to talk to the game characters (text dialogues);
- Ability to have a dictionary available all the time;
- Ability to move freely around the game world map;
- Ability to follow the game storyline.

Because of the extensive volume of the described above aim, the work on this project was divided between two authors. The second author's tasks were:

- Creating main and secondary game characters with their corresponding animations;
- Modelling maps and tiles;
- Compose educational texts for the dialogues.

Tasks

To achieve the aim of this work the author has to complete the following tasks:

1. Check, which similar solutions already exist and analyse them.
2. Choose appropriate technologies for the development of the game.
3. Development of the game core, which includes game world map rendering, main character walking, objects collision etc.

4. Development of the game user interface, which includes a main menu, loading screen, pause menu overlay etc.
5. Development of the game dialogue system, the in-game dictionary and final score overlay after a level completion.

Outline

Section 1 describes all the variety of technologies chosen and used by the author.

Section 2 describes available similar solutions, their strengths and limitations.

Section 3 describes the main elements of the game architecture developed by the author.

Section 4 describes the results of the development in details.

Section 5 describes the overall rating of the game based on the testing with children.

1 CHOSEN TOOLS AND TECHNOLOGIES

In this section, the author describes the technologies, he is going to use in the development of the game and explains his choices.

1.1 Android

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touchscreen mobile devices such as smartphones and tablets. (Android... 2018)

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2017, it has over two billion monthly active users, the largest installed base of any operating system, and as of 2017, the Google Play store features over 3.5 million apps. (Android... 2018)

Considering the overwhelming domination of Android operating system on mobile devices, it is an obvious and logical choice to pick it for the development of the mobile game.

1.2 Java

As Java[1] is the main programming language for Android, the author chose Java for the development of the mobile game.

1.3 Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Google created Android Studio based on IntelliJ IDEA specifically as an environment for the Android applications development. (Google 2018) That is why the author chose Android studio for the development of the mobile game.

1.4 Gradle

Gradle[2] is an open-source build automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language (DSL) instead of the XML form used by Apache Maven for declaring the project configuration. Gradle uses a directed acyclic graph ("DAG") to determine the order in which tasks can be run. (Gradle 2018)

[1] https://www.java.com /
[2] https://gradle.org/

The Android Studio is using the Gradle by default, already adapted and configured, which makes it the logical choice for this work as a build tool.

1.5 LibGDX

LibGDX is a relatively low level, free, open source cross-platform game development framework. The goal of the project is to assist developers in creating games/applications and deploy to desktop and mobile platforms without getting in the way and letting them design however they like. (Zechner 2013)

This framework is fast and very lightweight. The main alternative technology for LibGDX would be Unity. Unity requires more performance and disk spaces (for instance, an empty Unity project is about 8 megabytes, while an empty LibGDX project is about 200 kilobytes). (LibGDX vs Unity 2017)

The author chose LibGDX because of its performance, absolute manual customization and own project structure. Some developers might consider manual customization a disadvantage, because comparing to Unity it has no GUI and, therefore, developers need to write code only, but the author thinks that benefits of LibGDX higher performance by far outweigh this consideration.

1.6 Ashley and ECS

Entity–component–system (ECS) is an architectural pattern, which is mostly used in game development. An ECS follows the Composition over inheritance principle that allows greater flexibility in defining entities where every object in a game's scene is an entity (e.g. enemies, bullets, vehicles, etc.). Every Entity consists of one or more components, which add additional behaviour or functionality. Therefore, the behaviour of an entity can be changed at runtime by adding or removing components. This eliminates the ambiguity problems of deep and wide inheritance hierarchies that are difficult to understand, maintain and extend. Common ECS approaches are highly compatible and often combined with data-oriented design techniques. (Entity-component-system 2018)

Ashley is a tiny entity framework written in Java. It is inspired by frameworks like Ash (hence the name) and Artemis. Ashley tries to be a high-performance entity framework without the use of black magic and thus making the API easy and transparent to use. Ashley lives under the LibGDX family. (Saltares 2018)

The LibGDX contains the Ashley framework, which is based on Entity-Component-System pattern; therefore, the author chose this approach for greater flexibility during the development.

1.7 Scene 2D

Scene 2D is a 2D scene graph for building applications and UIs using a hierarchy of actors. Scene 2D is well equipped for laying out, drawing, and handling input for game menus, HUD overlays, tools, and other UIs. (Villegas 2018)

It is theoretically possible to create controls and menus with just the means of LibGDX, but Scene 2D has a tight compatibility with LibGDX and it is much easier to build UIs using Scene 2D, therefore, the author chose this framework.

1.8 Tiled

Tiled is a 2D level editor that helps you develop the content of your game. Its primary feature is to edit tile maps of various forms, but it also supports free image placement as well as powerful ways to annotate your level with extra information used by the game. Tiled focuses on general flexibility while trying to stay intuitive. (Tiled Documentation Writers 2017)

It is very easy to extract layers (for instance collision objects) using LibGDX and maps files created by Tiled, therefore the Tiled was chosen for building and reading game maps.

2 PREVIOUS AND SIMILAR SOLUTIONS

To find similar existing solutions the author decided to explore the Google Play Store. The Google Play Store was chosen because of the choice of Android. It is impossible to consider all the applications because of their great number.

The author used "language learning" search query to find applications in the area under consideration and found out that existing applications for language learning (mobile games for such purpose were not found during the research) are very similar. Most of them only offer different kinds of quizzes, which do not include the entire wealth of engaging possibilities, which modern information technologies offer.

The **Table 1** compares features of the author game and another 10 most popular applications in the Google Play Store.

Table 1. *Top 10 applications and their features compared to this project*

App title	Quizzes	Dictionary	Final Score	Chatbot or online session	Texts with translations	Storyline	Open world game
The author's solution	✓	✓	✓			✓	✓
Duolingo	✓	✓	✓				
Memrise	✓		✓	✓			
Busuu	✓		✓				
Mondly	✓		✓				
Learn 50 languages	✓	✓	✓				
Babbel	✓		✓				
Beelinguapp		✓			✓		
Drops	✓	✓	✓				
Rosetta Stone	✓		✓	✓			
HelloTalk				✓			

Data: created by the author

The author provides two the most popular apps from this research. These are Duolingo and Memrise. The screenshots in this chapter were taken with a Samsung Galaxy S7 Edge with Android 7.0.

2.1 Duolingo

Duolingo[3] offers a structured solution, it is divided by subjects beginning with basic rules learning and finishing with more advanced topics. Every lesson is quiz with listening, speaking, choosing a right answer, and writing (**Figure 1**). Duolingo is also available in web version.

In the official website Duolingo says that it is in a gaming form with earning points etc., but Duolingo doesn't have a story and especially gaming features (such as walking in an open world), it is based on testing style only.

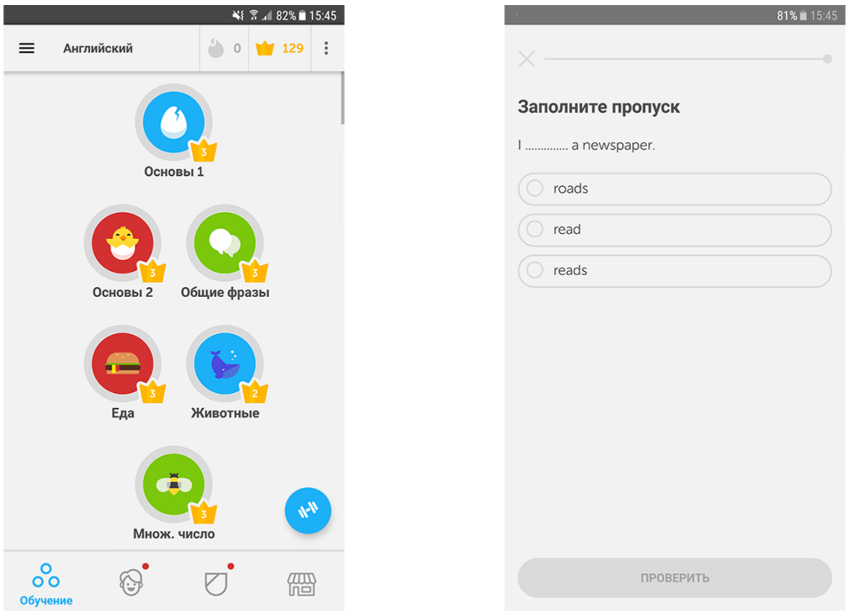


Figure 1. *Duolingo main menu on the left and Duolingo lesson on the right (Source: author)*

[3] https://www.duolingo.com/

2.2 Memrise

Another very popular application, which offers a simple way to learn vocabulary is Memrise[4].

The main disadvantage of the platform is the ability to learn words and phrases in the testing way only (it has a chatbot in the Web version) (**Figure 2**). It does not have any gaming features or storyline.

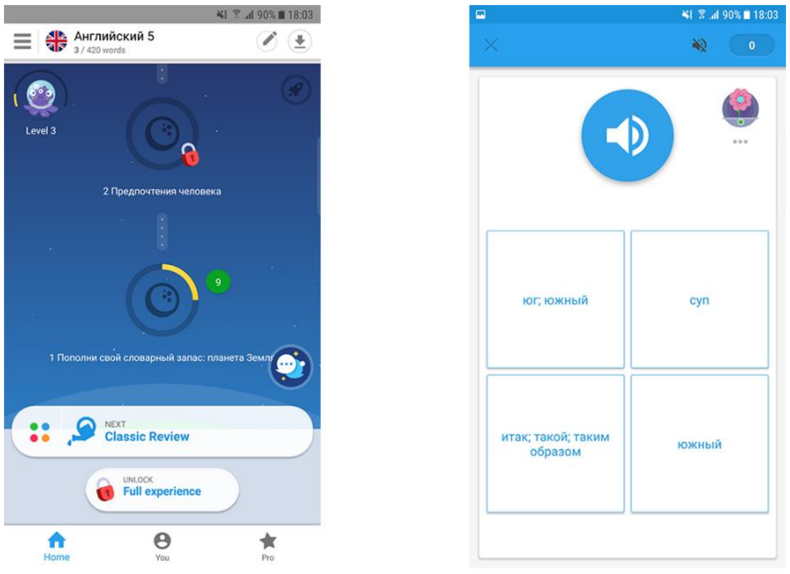


Figure 2. *Memrise main menu on the left and lesson on the right (Source: author)*

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

3 TECHNICAL OVERVIEW OF THE GAME

In the following section, the author describes the architecture of the game. Given part consists of seven chapters:

- Chapter 3.1 – gives definitions for mainly used terms.
- Chapter 3.2 – describes the development approach.
- Chapter 3.3 – describes small pieces for building entities.
- Chapter 3.4 – describes developed entities for the game.
- Chapter 3.5 – describes all general logic units.
- Chapter 3.6 – describes used techniques for a greater performance.
- Chapter 3.7 – describes the ease of new levels development.

This section does not include UI and transition effects implementation.

3.1 Terms

Component – an element that has no methods and is only capable of storing data. It used to compose an entity with several different components. (Entity-component-system 2018)

Entity – an existing or real thing. The word root is from the Latin, ens, or being, and makes a distinction between a thing's existence and its qualities. An entity exists and that is all it needs to do to be an entity. In some usages, an entity is close in meaning to object as it is used in object-oriented programming. (Rouse 2006) In this case, entity is an object that was composed from components and each component is a concrete characteristic of an entity.

System – a unit of logic that processes entities, which possess components of the same aspect as that System. (Entity-component-system 2018)

3.2 About the development pattern

As already mentioned the author uses Entity-Component-System pattern with Ashley (1.6) framework. In this game, everything is an Entity (characters, game world, controls, dialogues, collision objects etc.). Every entity is created by a set of Components and every Component is some specific characteristic of an entity. Any system is supposed to process entities, which match to specific Family (any family describes which components the entity should have to be processed.) (**Figure 3**).

Some systems can do tasks that are not associated with any entity at all.

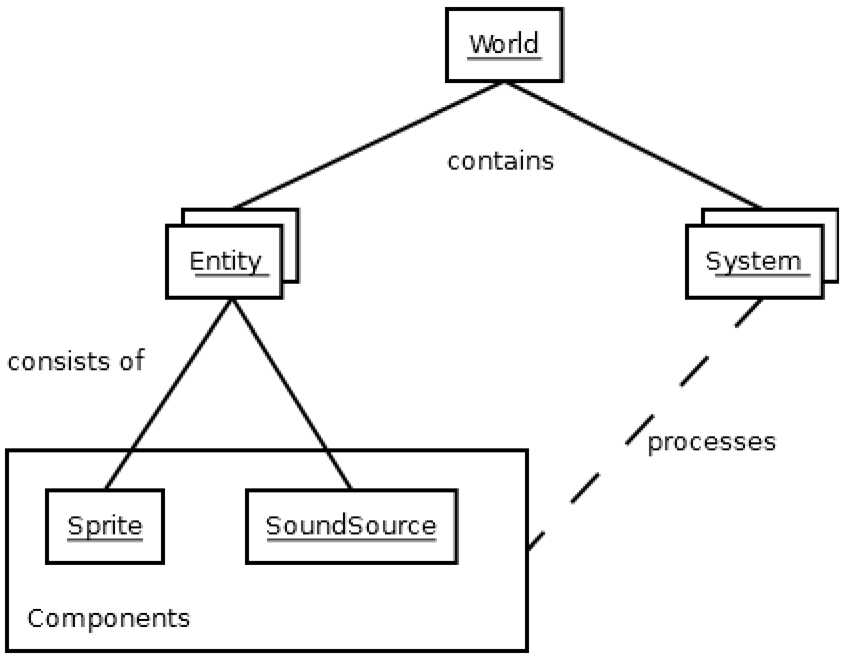


Figure 3. *Entity-component-system diagram (Source: Marcus von Appen 2017)*

One of the advantages of this development approach used by the author is the ability to disable processing of any system at runtime and delete any component from entities as well.

3.3 Components

There are two kinds of components: the first contains any information, which can be used during processing in some system, the second is used like Tag (does not contain any information, it is used to differentiate entities with the same set of components).

3.3.1 Components containing data

The components containing data are listed below.

The components, which are holding data:

- **Position Component –** contains X and Y coordinates.
- **Dimension Component –** contains Width and Height to define sizes.
- **Bounds Component** – contains polygon to define physical bounds (Used for collision detection).
- **Velocity Component** – contains velocity (X and Y velocities)
- **Particles Component** – Holds an effect object of particles and a processing flag (true or false to process the particles or not).
- **Direction Component** – Defines direction for an entity (used for appropriate animation while movement).

- **Animation Component –** contains a list of animations, animation index and animation time. Index is used to get a required animation object from the list (index is set under a certain condition). Animation time is used to get needed frame.
- **Sound Component** – contains a sound that plays under a certain condition.
- **Music Component –** contains a music track that plays during the game.
- **Texture Component** – contains a texture region (region is a piece of a texture atlas 3.6.2) for further rendering.
- **Z Order Component** – contains a number that defines a rendering priority.
- **Dictionary Component** – contains all available words and favourite words.
- **Controls Component** – contains four polygons for each movement direction (Up-Left, Up-Right, Down-Left, Down-Right) to detect that a point of a screen touch is within one of the polygons.
- **Stage Component** – contains a stage object, which is used for dialogues and as a User Interface.
- **Story Component** – contains a queue of conversations (each conversation contains game character texts and answers to choose), words of the author and main character monologues.
- **NPC Component** – contains Character type (Character type defines what kind of character it is supposed to be).
- **Portal Sensor Component** – contains a name of a sensor (Sensor is a polygon object, which is used to detect, for example, a collision with a door to enter a building).
- **Portal Sensor Spawn Component** – contains a name of spawn point that is near to a sensor. Used to get a spawn point to place a player at the correct place.
- **Tiled Map Component** – contains a game world map object.
- **Tiled Map Renderer Component** – contains the renderer for a game world map.

3.3.2 Tag components

The tag component does not have any information.

The tag components:

- **World Object Component** – defines a “world entity” (Sensors, spawn points, collision objects, a map).
- **HUD Component** – defines an entity as an object for displaying a data to the screen and for other controlling elements.

- **World Collision Object Component**- used to differentiate collision objects from other world objects.
- **Player Component** – defines an entity as a Player.

3.4 Entities

Every entity is produced by composing the components. The author had developed 11 entity types in total.

The entities:

- **Player** entity consists of the following components: position, dimension, bounds, velocity, particles, direction, animation, texture, sound, z order and player. This entity represents the main character of the game.
- **NPC** entity consists of the following components: position, dimension, bounds, sound, animation, texture, z order and NPC. This entity represents a character with whom a user can have a dialogue. NPC can represent different characters (NPC Component has a Character type data that defines a character).
- **Map** entity includes Tiled Map Component, Tiled Map Renderer Component, Bounds Component and World Object Component. The map entity is the game world on which a player can walk.
- **Portal Sensor** entity: bounds, position, sound, world object, and portal sensor components. This entity used to detect collisions with the player entity and switch map files (and delete all entities and create new ones for new map). For example, if the player collides a house door, the map will be changed from outside to the room of the house.
- **Portal Sensor Spawn** entity, it contains only position, portal sensor spawn, and world object components. It used to place the player to correct coordinates after the map change.
- **Collision objects**: bounds, position, world object, and world collision object components. It is like invisible borders, which are used for collision detection with the player to not allow the player to pass through.
- **Controls** entity: texture, HUD, position, dimension, controls components. The controls entity represents a cross arrow to move the player.
- **HUD** Entity: stage, sound, and HUD components. This entity is responsible to hold the game Graphical User Interface (dictionary tab, pause button and score number) to interact with a user.

- **Dictionary** entity: dictionary component. It holds two sorted maps (in this case, map is a list of key and value pairs) of all available words with translation and a user's favourite list of words.
- **Story** entity: story and sound components. Holds the story queue (All the text of the characters and all the answers, the author speech and the main game character monologues) and the finishing melody.
- **Music** entity: music component. It just holds a music object.

3.5 Systems

A system in the game represents a logic unit, which processes entity components if the entity matches for a specific family. A family is a set of component types to which an entity can match or not. If an entity matches to a family in a system, then its components (the components defined in the family) are going to be processed.

Some systems do not process any entity.

The tables, which describe all the systems, are located in the Appendices.

3.5.1 Active Systems

This type of systems executes every frame (60 frames per second). Some of them are using passive systems inside (Passive Systems 3.5.2).

The active systems are described in the **Table 2.** Active systems.

3.5.2 Passive Systems

This type of systems executes only when some condition becomes true inside other active systems and only once. Such systems are used when it is excessive to run it every frame or for some other reason.

The passive systems are described in the **Table 3.** Passive systems.

World Objects Collision System, World Wrap Up System, NPC Collision System and Sensor Collision System are widely used in the Movement System. The Movement System checks for collision to prevent any further movement through objects, which are not supposed to be passed through.

Character Render System and Particles System are used inside the World Render System: initially it renders the background layers of the map textures, then updates the Character Render System with the Particles System and finally it renders the foreground layers. It gives a 3D effect, when, for example, the main character is overlapped by a tree.

3.5.3 Debug Systems

This type of systems is used as a utility (Grid on a screen or auxiliary key handlers for a convenient development). Such systems are not supposed to be accessed by other users. These systems are useful while the development.

The debug systems are described in the **Table 4.** Debug systems.

3.6 Performance

3.6.1 Assets Management

An Asset Manager in LibGDX is a class that is used to load and unload resources (Textures, sounds, music, fonts etc.) for a developer's project. It will store a number of references an asset has and keep it loaded until it is no longer required. (Day 2017)

By using an Asset Manager a developer will load items asynchronously, keep all the assets together, and stop loading duplicate resources and keep references of used assets until no other assets reference it. This will help reduce memory usage in a developer's project as well as make loading assets easy. (Day 2017)

The author uses Asset Manager to centralize and use effectively all the resources in the project. Trying to load resources at runtime could lead to dramatically slowing down the game.

3.6.2 Texture atlases

To draw a Game Object on the screen, the engine has to issue a draw call to the graphics API (such as OpenGL or Direct3D). Draw calls are often resource-intensive, with the graphics API doing significant work for every draw call, causing performance overhead on the CPU side. This is mostly caused by the state changes done between the draw calls (such as switching to a different Material), which causes resource-intensive validation and translation steps in the graphics driver. (Unity Technologies 2018)

Texture atlas is just an image that contains multiple other images (textures). Such image is associated with a file descriptor which contains X and Y coordinates with sizes for every texture. If every element or entity has a separate texture, it is drawn with a separated "draw call." Each draw call takes some time to complete, making the rendering process longer and longer. (Sciutteri 2016)

The author had decided to pack all the textures to texture atlases. The game has several atlases, because of the large number of frames for the animations. The game would be very slow if it did draw calls for every separate image of any animation instead of getting regions from atlases. The **Figure 4** shows the Texture Atlas of the player that contains all frames of his animations.

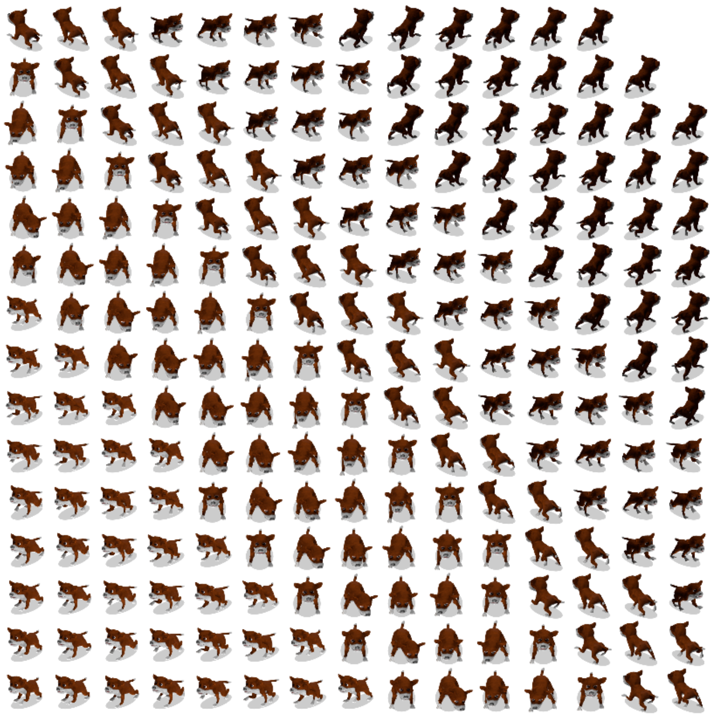


Figure 4. *Atlas example (This atlas contains all the walking and standing animation frames in all needed directions) (Source: author)*

3.7 Ease of future development

The structure of the game project developed by the author is very flexible for future levels. The developed game architecture allows to create any sequence of dialogues (4.4.3), monologues (4.4.2) or author speech (instructions) (4.4.1) for future levels. Currently it has only one level, which starts directly from the Main Menu Screen (4.5.2). A menu with a level selection is already planned. The game was developed in such a way that it does not require to code anymore or recompile the project to integrate new levels to the game. It is enough to just import new story sequence to the stories directory (with name level_2.json for example), create a directory (with name level_2) and populate it with new map files.

4 THE RESULT OF THE DEVELOPMENT

In the following section, the author describes the game as a whole (what kind of features it has and how to use them):

- Chapter 4.1 – lists functional and non-functional requirements.
- Chapter 4.2 – describes the requirements for a successful installation.
- Chapter 4.3 – describes the general gaming capabilities.
- Chapter 4.4 – describes the learning gaming capabilities.
- Chapter 4.5 – describes the user interface and HUD.
- Chapter 4.6 – describes the transition effects the game has.
- Chapter 4.7 – lists all the sounds and music in the game.

4.1 Requirements

4.1.1 Functional requirements

The functional requirements describe the main functionality of the game:

- Players must have an ability to walk and explore a game world.
- Players must have an ability to enter houses.
- Players must be able to find words translations and save them as favourites.
- In-game instructions must notify and help players during the game.
- The game must have branching dialogues as quizzes and ability to train player’s language skills.
- Players must be able to see final score after a level completion.
- Players must be able to pause the game.
- Players must be able to quit the game or resume using the pause menu.
- Players must be able to start or quit the game using the main menu screen.
- Players must see the loading screen as an explicit processing.
- The game must prevent a player leaving the main game boundaries.

4.1.2 Non-functional requirements

The non-functional requirements describe the game aspects, which do not affect to the functionality.

- The game must run on Android devices such as Smartphones and Tablets with Android 4.0 version or higher.
- The game is required to have transition effects.

- The game screen must have simple controls.
- The dictionary in the game must be available without Internet connection.
- The game must not require high performance devices.
- The game is required to have main character monologues.
- The game must have music and sounds.
- The game must not let a player to cheat.
- The game must be easy to scale for future levels.
- The game must be able to slide a player along an edge of a collision object if it has a complex shape and edges are not aligned to the player movement axis.

4.2 Installation

The game requires an Android phone or tablet with Android version 4.0 or higher.

Minimum installation requirements for Android devices:

- Android 4.0 or higher;
- 35 MB of disk space;
- 1024 MB of RAM;
- 1.2 GHz 32-bit processor;
- Any screen resolution.

The RAM and processor requirements were taken testing the game on the Samsung galaxy core LTE (SM-G386F), these hardware requirements are considered as a minimum because the author did not have access to any smartphone with lower performance hardware.

The appendices section contains all the necessary links to download the game.

The game also can be installed on desktop devices, which run on Windows OS. The installation file link is also included in the appendices section.

4.3 Main gaming features

4.3.1 Open world

Video games with open or free-roaming worlds typically lack the invisible walls and loading screens common in linear level designs. (Open world 2018)

The key gaming feature in the game developed by the author is the ability to walk in the game world. A player can walk wherever he likes (within boundaries), explore environment to make learning more enjoyable and interesting.

4.3.2 Ability to enter houses

The author developed a feature to enter houses. To enter a house, it is enough to move closer to a door and the game will automatically replace the world map to the room map. Such feature makes the game more interesting and the game world more expressive.

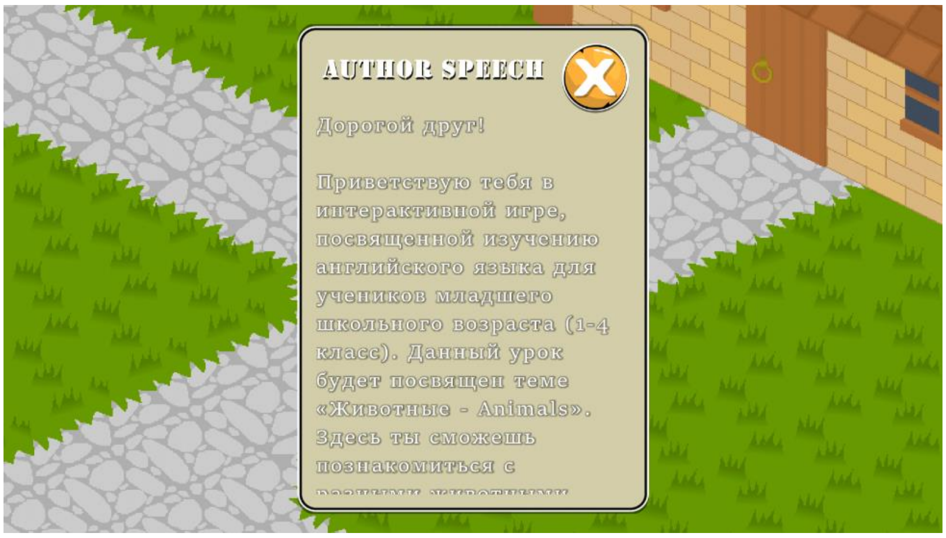
4.4 Learning elements

The author developed several features to support the language learning:

- Instructions while playing.
- Main character monologues.
- Structured dialogues.
- Available dictionary while playing.
- Final score to asses a user knowledge.

4.4.1 Instructions while playing

The game is helping a user with in-game instructions. When it is necessary, the game will show the scrollable window with next instructions to support player in potentially confusing situations (**Figure 5**). The close button appears with a little delay, so the user does not close it accidently.



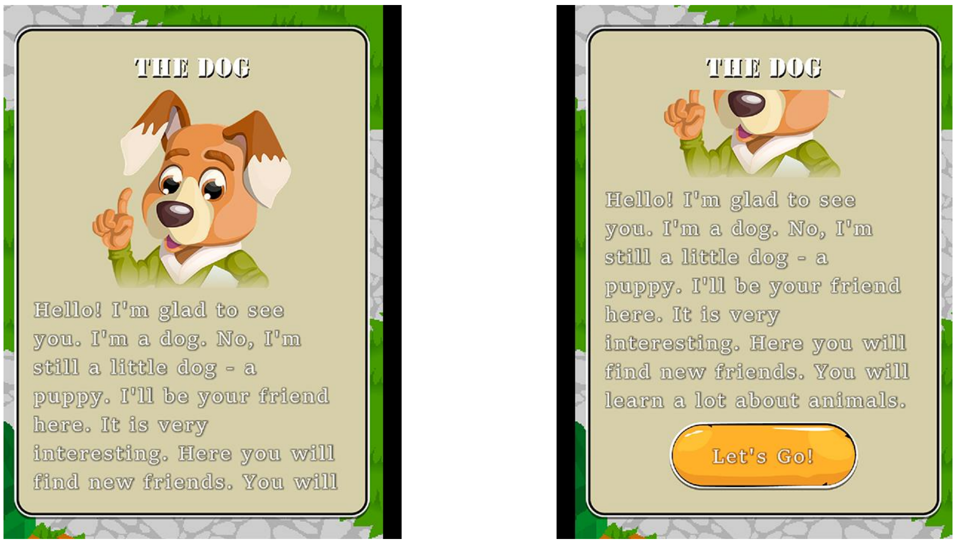
Дорогой друг!

Приветствую тебя в интерактивной игре, посвященной изучению английского языка для учеников младшего школьного возраста (1-4 класс). Данный урок будет посвящен теме «Животные - Animals». Здесь ты сможешь познакомиться с

Figure 5. *Author speech to give instructions to a player (Source: author)*

4.4.2 Main character monologues

To make the game more interesting the author developed the monologues of the main character (**Figure 6**). The monologue window contains the main character portrait and the scrollable text. A user can find the button by scrolling to the bottom. The button will appear identically as with the instructions (4.4.1).


THE DOG

Hello! I'm glad to see you. I'm a dog. No, I'm still a little dog - a puppy. I'll be your friend here. It is very interesting. Here you will find new friends. You will

THE DOG

Hello! I'm glad to see you. I'm a dog. No, I'm still a little dog - a puppy. I'll be your friend here. It is very interesting. Here you will find new friends. You will learn a lot about animals.

Figure 6. *Main character monologue (Source: author)*

4.4.3 Structured dialogues

One of the most important learning features in the game is dialogues (**Figure 7**). The dialogues allow a user to talk to a character answering questions etc. Every answer may lead to different character answers, for example, in case of incorrect answer to a question, a character may say about your mistake and suggest another set of answers and so on. Every answer gives different amount of points depending on correctness.

The dialogue system prevents a possibility to cheat the game: the game saves a state of a dialogue and a collected score preventing memorization of incorrect answers trying to talk the character again with correct answers.

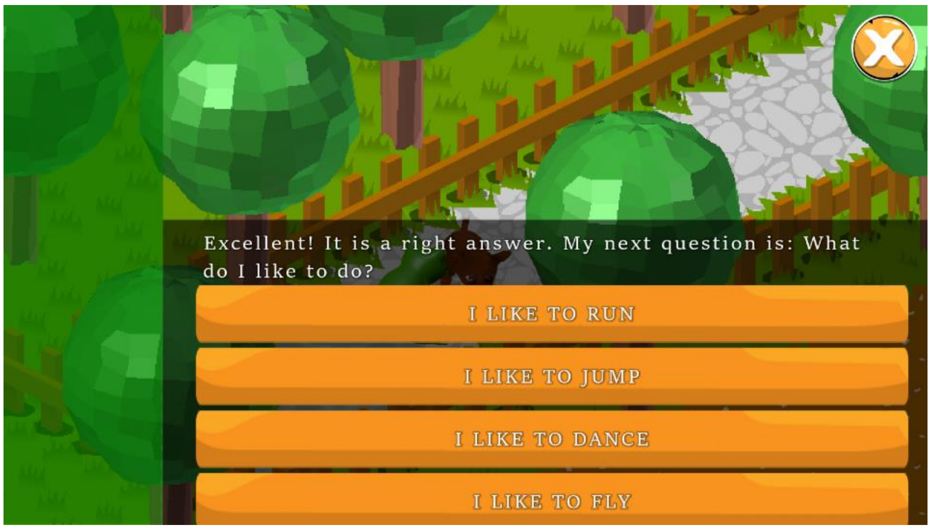

Excellent! It is a right answer. My next question is: What do I like to do?

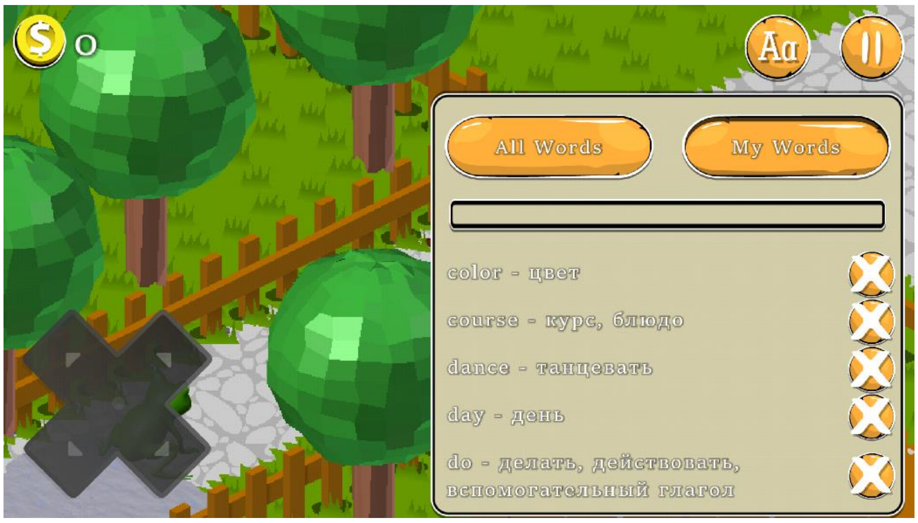
Figure 7. *Dialogue window (Source: author)*

4.4.4 Available dictionary while playing

The dictionary helps to prepare before a conversation, because it is available during free walking (**Figure 8**). The dictionary is going to be populated with new words before a next conversation with a next character. The dictionary is divided into two tabs:

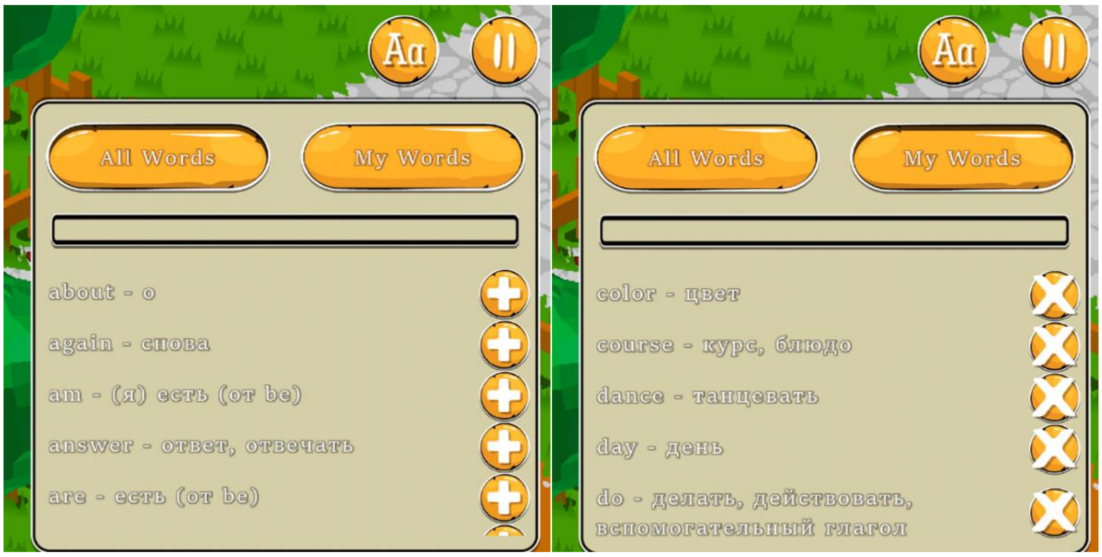
- All words (**Figure 9** on the left side) – this tab shows all the available words at the moment. There is the add button (plus sign) next to every word to add a word to a user favourite list (my words). The plus sign disappears if the word is already added.
- My words (**Figure 9** on the right side) – this tab contains all the added words by a user. Every word in this tab contains the delete button to delete a word from a user favourite list.

Both tabs have the searching feature.



color - цвет
course - курс, блюдо
dance - танцевать
day - день
do - делать, действовать, вспомогательный глагол

Figure 8. *Example of the opened dictionary (Source: author)*



about - о
again - снова
am - (я) есть (от be)
answer - ответ, отвечать
are - есть (от be)

color - цвет
course - курс, блюдо
dance - танцевать
day - день
do - делать, действовать, вспомогательный глагол

Figure 9. *All words tab on the left side and my words on the right (Source: author)*

4.4.5 Final score to assess a user knowledge

The game gives some amount of score according to answers in the dialogues. At the end of the game level, it gives an assessment based on the maximum amount of score (90% or higher – 5 stars; 75% – 90% gives 4 stars; 50% – 75% gives 3 stars; 30% – 50% gives 2 stars; and below 30% gives only one star). The stars appear one by one with a nice transition and vibration effects. The menu contains only one button "quit", which redirects to the Main menu screen (4.5.2). (**Figure 10**)

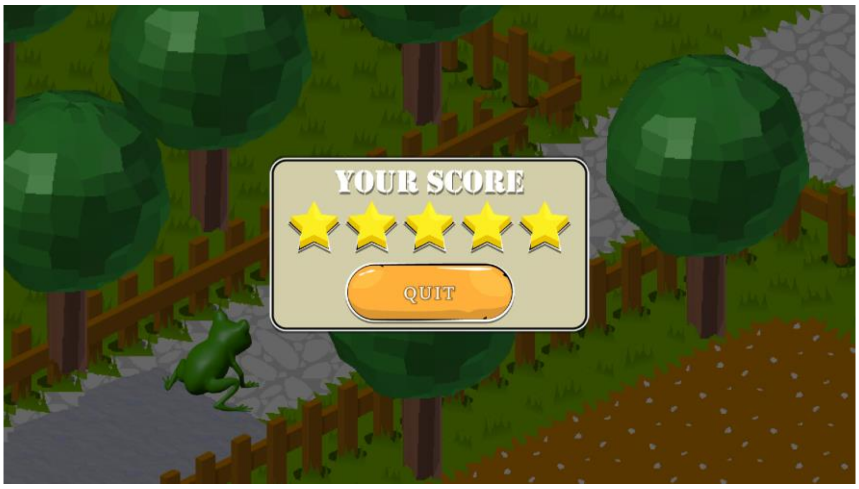


Figure 10. *Final assessment of the level (Source: author)*

4.5 Head-up display and user interface

4.5.1 Loading screen

Starting the game, it needs at first to load all the necessary resources (assets). The author developed the loading screen to show a user the loading resources processing explicitly (**Figure 11**). The loading bar starts from the centre and moves in both directions.

The loading time depends on a device hardware; however, the game loading can load resources not longer than 10 seconds. The author's phone (Samsung Galaxy S7 Edge) loads all the resources within 4 seconds.

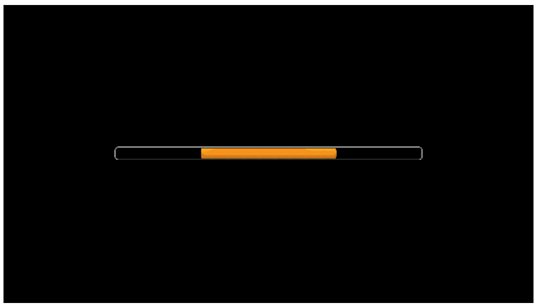


Figure 11. *Loading screen (Source: author)*

4.5.2 Main menu screen

The main menu screen (**Figure 12**) is very simple. It consists of the background image and two buttons:

- Play button – starts the gaming process by opening the Game screen (4.5.3).
- Quit button – closes the game.

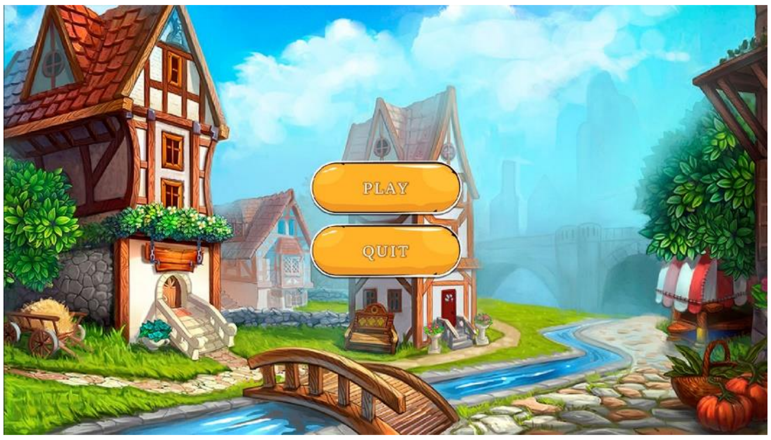


Figure 12. *Main menu screen (Source: author)*

4.5.3 Game screen

The game screen contains (**Figure 13**):

- Current score – it shows how many points a player has collected as a result of the dialogues during playing.
- Controls – the cross-arrow to move the main character diagonally.
- Dictionary button – to open the dictionary (4.4.4).
- Pause button – to open the pause menu (4.5.4).

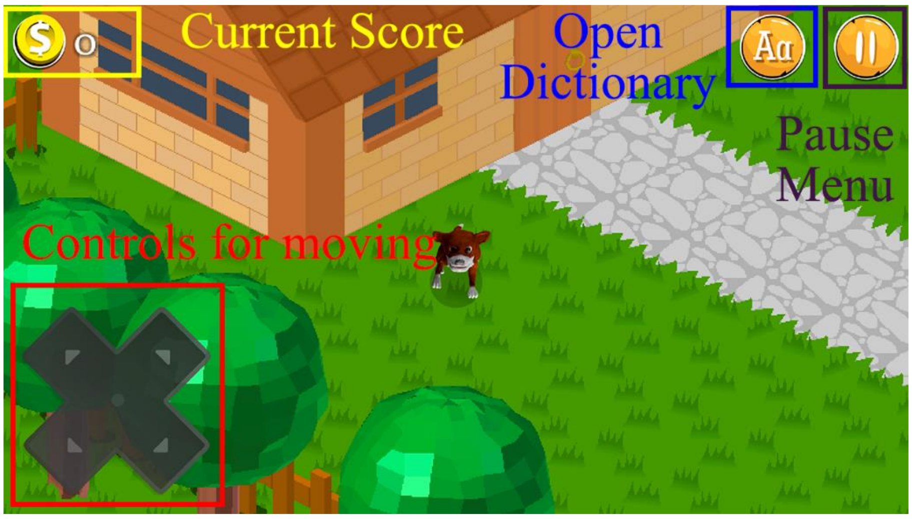


Figure 13. *Game screen buttons, controls, and score (Source: author)*

4.5.4 Pause menu

Opening the pause menu, the game hides all other controls from the screen. The pause (**Figure 14**) menu contains a couple of buttons:

- Quit button – this button returns to the main menu.
- Resume button – closes the pause menu and resumes the game.

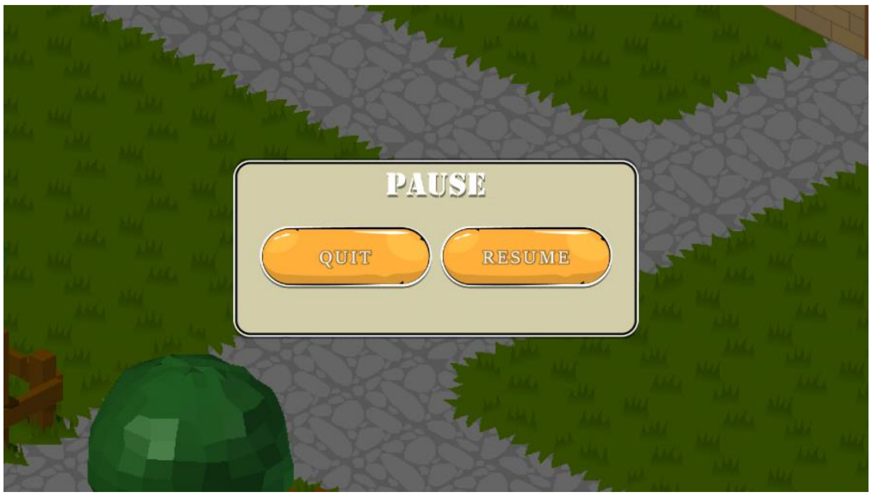


Figure 14. *Pause menu (Source: author)*

4.6 Transition effects

The author decided to develop transition effects to make the game smoother during playing. The game has transition effects between every screen (from the loading screen to the main menu screen, for example): loading screen (4.5.1), main menu screen (4.5.2) and game screen (4.5.3). Additionally, the game has effects opening the dictionary (it slides from right to left, for example) (4.4.4), pause menu (4.5.4), final score (4.4.5), instructions (4.4.1) and main character monologues (4.4.2).

4.7 Sounds and music

For the greater atmosphere and gaming experience the game contains sounds and music.

The game contains the following sounds:

- Walking sounds.
- Door sound entering any house.
- Achievement sound effect earning new points.
- Greeting sound for every character.
- Finishing sound finishing a level.

The music starts immediately after pressing the play button. Finishing a level, the game will reduce the music volume to make the finishing sound more hearable.

5 APPROBATION

5.1 The method

The game was tested in the "Narva Paju Kool" school with Olga Epinatieva (class teacher).

To measure the quality, ease of use and interest of the game the author decided to evaluate a satisfaction of a group of young students (third grade).

A group of eleven participants (aged 9-10) was chosen for the testing. The testing was individual for every participant (one on one). The participants were required to play from start to end and answer the following questions:

- Have they encountered any difficulties;
- Do they have any complains;
- What were their impressions during the game and how would they rate the game from 1 to 5;
- Do they prefer to study languages playing such games or following traditional methods as they do right now;
- Do they have any further suggestions.

5.2 The results

The test participants found following problems:

- Sometimes it is hard to find the next game character for a new conversation.
- The storyline is not clear enough, sometimes a participant could have stuck with a question like "Why I need to go to this character?"
- The characters are without animations and some other graphical complains like black background in the houses.
- The children do not like to read long monologues or in-game instructions.
- Not enough levels.

The suggestions to improve the game experience were as following:

- To add labels to items, for example, "tree" near a tree when the main character comes close enough.
- The game could have a navigation arrow, which shows the right direction to find the character if a user spent too much time and did not find the needed character.

The ratings were high with average 4.73, as visualized in **Figure 15**. Some of the children even tried to find the game in the Google Play Market as a sign of their interest. All of them answered that they definitely would prefer gaming education instead of traditional methods as it is shown in **Figure 16**.

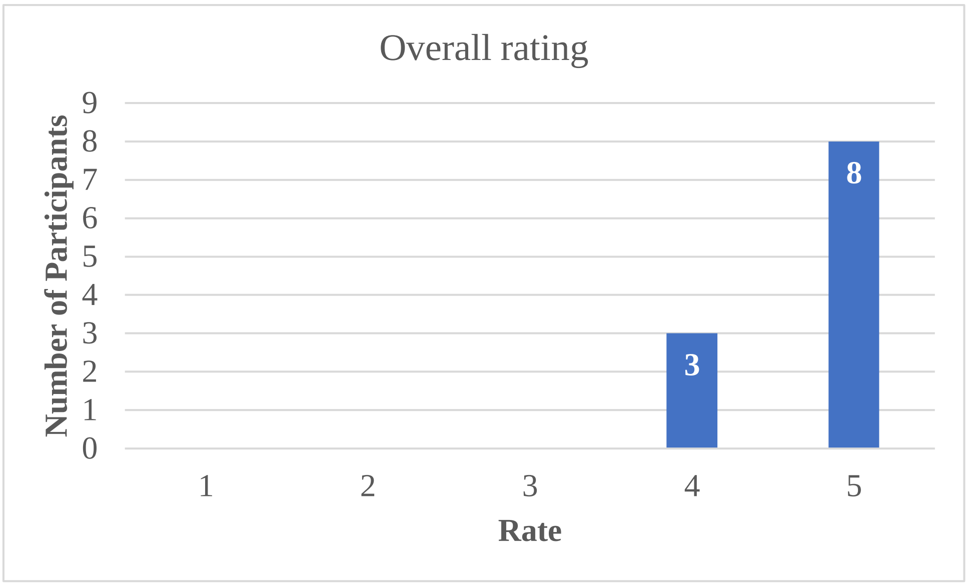


Figure 15. *Overall rating results*

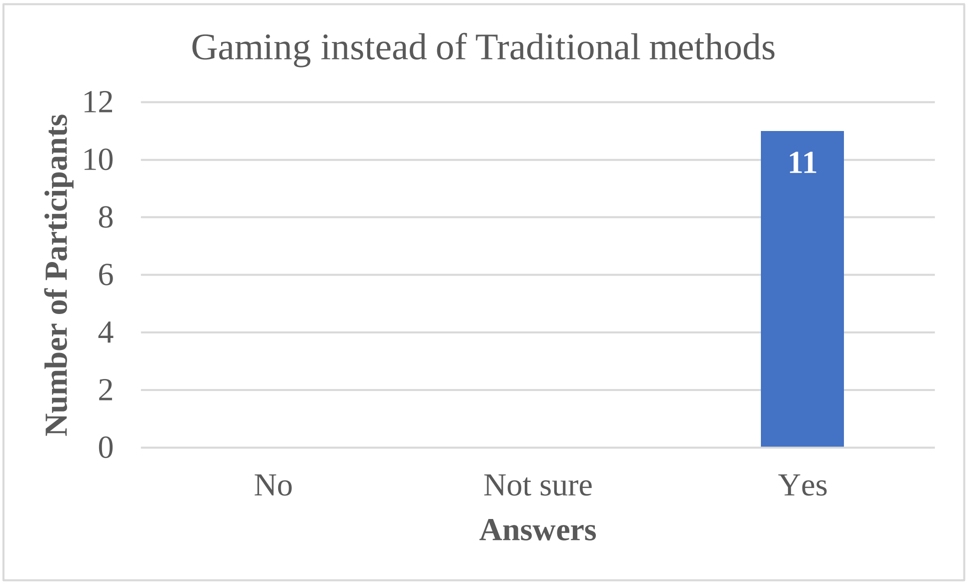


Figure 16. *Preferences results*

CONCLUSION

E-learning is our future, which develops quite actively, however e-learning has the same problem as formal learning – a problematic level of motivation. One of the major ways to meet this challenge is bringing entertainment into education. Gamification of education is one of the branches of this vast process.

The author of this work intended to participate in this important field through the development of an educative game, which could motivate students to learn languages by playing.

The first task of this work was to explore the market for similar solutions. The research results and comparison with the author solution are described in the section 2.

The second task of this work was to choose appropriate technologies for the development of the game. The author has solved this task by using the following stack of technologies for the game development:

- Android as a target operating system;
- Android Studio as a development environment;
- Java as a main programming language;
- Gradle as a build tool;
- LibGDX as a core game engine;
- Entity-Component-System with Ashley as a flexible way to structure the project;
- Scene 2D as a library for UI creation;
- Tiled as a map editor.

The main tasks consisted from the development of all game elements such as:

- Map rendering;
- Main character walking;
- Objects and characters collisions;
- Entering houses;
- Pause menu;
- Loading, main menu and game screens;
- Dialogue system including monologues and instructions;
- In-game dictionary;
- Showing final score.

The author described the completion of these steps in details in the section 3 and 4. Therefore, the author can conclude, that he has fully accomplished the aim of the project.

Currently the game has following features:

- Players are able to complete levels and get confirmations of their progress, which should boost their confidence and motivation;
- Players can talk to the game characters through the text dialogues;
- Players have a dictionary available when necessary;
- Players can move freely around the game world map and also enter game houses;
- Players have to go through the game storyline to complete the level.

In the future, the author intends to add following additional features to the game such as AI (Artificial Intelligence) to make the characters behaviour and dialogues more interesting; the ability to pick up items and bring them to characters as a task; and many more thematically filled levels.

RESÜMEE

Töö pealkiri on „Interaktiivse inglise keele õppe mobiilse mängu arendamine venekeelsetele kasutajatele“. Töö eesmärk oli arendada mobiilse keeleõppe mängu, mis tõstaks õpilase huvi ja motivatsiooni inglise keele õppimise vastu.

Töö koosneb viiest peatükist, igaüks millest kõneleb projekti erinevatest aspektidest. Peatükk 1 kirjeldab kõiki tehnoloogiaid, mida on töös kasutatud. Peatükk 2 kirjeldab olemasolevaid sarnaseid lahendusi, nende tugevusi ja piiranguid. Kahest kõige populaarsemast lahendusest räägitakse eriti põhjalikult. Peatükk 3 kirjeldab autori poolt arendatud mängu arhitektuuri põhielemente. Peatükk 4 kirjeldab arendustöö tulemusi üksikasjalikult. Peatükk 5 kirjeldab mängu testimise tulemusi 3. klassi õpilaste poolt.

Töö oli ajendatud tõsiasjast, et traditsioonilised õppemeetodid koolis märgatavalt kaotavad oma efektiivsust ja kooliõpilaste motivatsioon õppimiseks üldse on üldiselt väga madal. Kooliõpilased ei taha käia koolis ja on õnnelikud iga kord, kui kooli ei pea minema. Üheks võimalikuks lahenduseks sellele probleemile näeb autor mänguelementide sissetoomise haridusse, eriti algkoolis. Antud töö oli mõeldud kui autori panus sellesse protsessi.

Autori poolt arendatud mäng loodab tõsta õpilaste huvi inglise keele õppimise vastu läbi järgmiste elementide:

- Mängijad saavad lõpetada mängu tasemeid ja saada kinnitusi enda edasijõudmise kohta, mis peaks tõstma nende enesekindlust ja motivatsiooni;
- Mängijad saavad rääkida mängu tegelastega läbi tekstiliste dialoogide;
- Mängijad saavad kasutada mängu sisse ehitatud sõnaraamatut, kui vaja.
- Mängijad saavad vabalt liikuda üle kogu mängu maailma ja siseneda majadesse.
- Mängijad peavad minema läbi mängu loo selleks, et lõpetada mängu tasemeid.

Tulevikus autor plaanib lisada mängu järgmisi funktsioone: tehisintellekti (AI) selleks, et teha mängu tegelasi ja nende dialooge huvitavamaks ja mitmekesisemaks; võimaluse kasutada mängu esemeid ja tuua neid mängu tegelastele mänguülesannete täitmise käigus; ja palju-palju uusi temaatilisi mängu tasemeid.

REFERENCES

Ackerman, Diane 2000. *Deep play.* A division of random house, Inc. New York City, New York.

Sharpened Productions 2015. E-learning. Available at https://techterms.com/definition/e-learning, accessed May 1, 2018.

Davis, Ben 2009. What are game assets? Available at http://conceptdevelopmentbendavis.blogspot.com.ee/2009/02/what-are-game-assets.html, accessed May 2, 2018.

Foley, James; Andries van Dam; Feiner, Steven; Hughes, John 1987. *Computer Graphics: Principles and Practice.* Addison-Wesley Publishing Company. Reading, Massachusetts.

W3Schools 2018. JavaScript JSON. Available at https://www.w3schools.com/js/js_json.asp, accessed May 1, 2018.

Technopedia 2018. Non-Player Character (NPC). Available at https://www.techopedia.com/definition/1920/non-player-character-npc, accessed May 1, 2018.

Cambridge Dictionary 2018. Meaning of "HUD" in the English Dictionary. Cambridge University Press. Available at https://dictionary.cambridge.org/dictionary/english/hud, accessed May 1, 2018.

Cambridge Dictionary 2018. Meaning of "gamification" in the English Dictionary. Cambridge University Press. Available at https://dictionary.cambridge.org/us/dictionary/english/gamification, accessed May 1, 2018.

Technopedia 2018. Rendering. Available at https://www.techopedia.com/definition/9163/rendering, accessed May 16, 2018.

Technopedia 2018. Central Processing Unit (CPU). Available at https://www.techopedia.com/definition/2851/central-processing-unit-cpu, accessed May 16, 2018.

Technopedia 2018. Graphical User Interface (GUI). Available at https://www.techopedia.com/definition/5435/graphical-user-interface-gui, accessed May 16, 2018.

Prensky, Marc 2005. *Computer Games and Learning: Digital Game-based Learning.* Handbook of Computer Game Studies. 97-122.

Wikipedia 2018. Android (Operating System). Available at https://en.wikipedia.org/wiki/Android_(operating_system), accessed March 22, 2018.

Google. Meet Android Studio. Available at https://developer.android.com/studio/in-tro/index.html, accessed March 22, 2018.

Wikipedia 2018. Gradle. Available at https://en.wikipedia.org/wiki/Gradle, accessed March 22, 2018.

Wikipedia 2018. Entity-component-system. Available at https://en.wikipe-dia.org/wiki/Entity%E2%80%93component%E2%80%93system, accessed April 9, 2018.

Marcus von Appen 2017. A component-based entity system framework. Available at http://python-utilities.readthedocs.io/en/latest/ebs.html, accessed May 5, 2018.

Zechner, Mario 2013. Introduction to LibGDX. Available at https://libgdx.bad-logicgames.com/documentation/, accessed April 6, 2018.

LibGDX vs Unity – How to choose an Android game engine: libGDX vs Unity. 2017. Available at http://dariopenic.com/how-to-choose-an-android-game-engine-libgdx-vs-unity/, accessed April 6, 2018.

Villegas, Julien 2018. Scene2D. Available at https://github.com/libgdx/lib-gdx/wiki/Scene2d, accessed April 29, 2018.

Saltares, David 2018. Ashley. Available at https://github.com/libgdx/ashley, accessed April 1, 2018.

Tiled Documentation Writers 2017. Tiled Documentation. *Introduction.* Available at http://doc.mapeditor.org/en/latest/manual/introduction/, accessed April 1, 2018.

Rouse, Margaret 2006. Entity. Available at https://whatis.techtarget.com/definition/en-tity, accessed April 9, 2018.

Rouse, Margaret 2018. Random Access Memory (RAM). Available at https://searchstorage.techtarget.com/definition/RAM-random-access-memory, accessed May 19, 2018.

Day, John 2017. Game Development. Asset Manager – LibGDX Tutorial. Available at https://www.gamedevelopment.blog/asset-manager-libgdx-tutorial/, accessed April 8, 2018.

Sciutteri, Matteo 2016. Using a Texture Atlas to Optimize Your Game. Available at https://gamedevelopment.tutsplus.com/articles/using-texture-atlas-in-order-to-optimize-your-game--cms-26783, accessed April 8, 2018.

Unity Technologies 2018. Unity Manual. Draw call batching. Available at https://docs.unity3d.com/Manual/DrawCallBatching.html, accessed April 8, 2018.

Wikipedia 2018. Open world. Available at https://en.wikipedia.org/wiki/Open_world, accessed April 30, 2018.

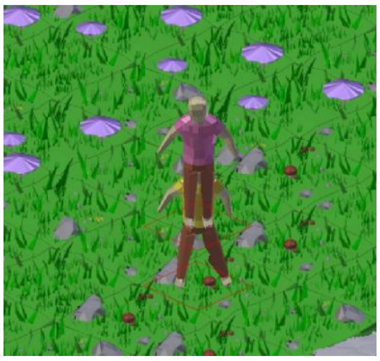
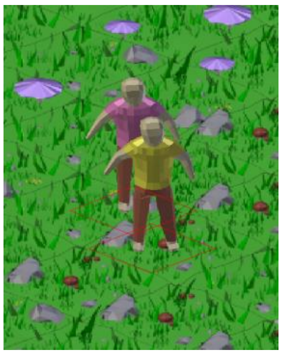
APPENDICES

Tables

***Table 2.** Active systems*

System Name	Family	Task
Bounds System	• Bounds Component • Position Component	Updates bounds position for every entity.
Movement System	• Position Component • Velocity Component • Particles Component	Updates position of an entity based on its velocity. Also checks for collision using passive collision systems (3.5.2) to prevent movement through objects. Sets the processing flag for the Particle Component if the entity is moving.
Animation System	• Animation Component • Texture Component	Retrieves next frame from the animation component and place it to the texture component for further rendering.
Camera Following Player System	• Player Component • Position Component	Retrieves a position from Player entity and set it to the Camera position.
HUD System	Controls Family: • HUD Component • Texture Component • Position Component • Dimension Component • Controls Component HUD Family: • HUD Component • Stage Component	Renders controls and HUD separately. Also listens any button pressing, but not controls.
Player Control System	Player Family: • Player Component • Velocity Component	Handles controls pressing. Checks which of the arrows where pressed, applies appropriate velocity to this direc-

System Name	Family	Task
	• Animation Component • Direction Component Controls Family: • Controls Component • Texture Component • HUD Component	tion and place appropriate animation index to the Animation Component of the player entity. Changes the texture of the controls to visualize in what direction the player moves.
Sound System	• Player Component • Sound Component • Direction Component	It makes a sound of walking if the direction is not Idle. It is an active system because of walking, but it is also partially passive because it is used sometimes to produce other sounds conditionally.
Music System	• Music Component	Simply plays a music. Restarts a track if it is finished.
World Render System	• Tiled Map Component • Tiled Map Renderer Component • World Object Component	Renders first two layers of the map (Background and Top background), then renders particle effects, then player with all characters and finally renders foreground. Such layered structure is needed to simulate 3D effect. It uses the Passive Systems (**Table 3**): for a player and characters it uses Character Render System, for particles effect it uses Particles System.
Z Order System	Player Family: • Position Component • Z Order Component • Player Component NPC Family: • Position Component • Z Order Component • NPC Component	This System sets the render priority. If a player is behind a character (Based on the Y coordinate), then the player gets higher render priority and will be rendered first (**Figure 18**) and vice versa. It gives the effect that a player can stay b

System Name	Family	Task
		hind or in front of a character. The **Figure 17** clearly demonstrates how it looks without the Z Order System.  ***Figure 17.*** *Without Z ordering (Source: author)*  ***Figure 18.*** *With Z ordering (Source: author)*
Conversation System	Story Family: • Story Component Dictionary Family: • Dictionary Component HUD Family: • HUD Component • Stage Component • Sound Component	It is a quite big system and a bit complex. Initially it is a passive system, but after a starting a dialogue with a character it becomes as an active system. It starts the conversation window with a character and loads appropriate dialogue image and dialogue sequence based on the character type. If a dialogue was started, then it plays the greeting sound based on a character. This system is simultaneously processing

System Name	Family	Task
		and rendering the dialogue. If the dialogue is over the system is shutting down itself and invokes Sound System to play a sound as an achievement (4.7).
Monologue System	• Story Component	Opens the author speech (4.4.1) window with the author text, if the author in the queue in the story component. In the same way, it starts the main game character monologue. (4.4.2)
Dictionary System	Story Family: • Story Component Dictionary Family: • Dictionary Component HUD Family: • Stage Component • HUD Component	The system checks for a next conversation to add new words to the dictionary before the actual conversation. The User Interface contains the dictionary window (4.4.4), and it needs to be updated.
Finish System	• Story Component • Sound Component	Checks with an interval of a couple of seconds the story queue for emptiness. If the story is empty, then the game state is going to be set as finished. Then the final score menu appears (4.4.5), and the Sound System is invoked to play the finishing sound (4.7).

Data: created by the author.

Table 3. *Passive systems*

System Name	Family	Task
Character Render System	• Position Component • Texture Component • Dimension Component • Z Order Component	Renders characters and a player by priority based on Z Order Component. It gets a texture from the Texture Component, place it to the Position from Position Component and applies sizes with Dimension Component. It is passive by

System Name	Family	Task
		itself but used in the World Render System every frame (**Table 2**).
Particles System	• Position Component • Particle Component	Gets and renders a particle effect and puts it in a proper position. For example, a dust while the player walking. It is passive by itself but used in the World Render System every frame (**Table 2**).
Clean Up System	NPC Family: • NPC Component World Object Family: • World Object Component	It removes all characters and all world objects. Often used while changing maps (while entering any building in the game).
Entity Factory System		This system creates entities. Creates every component for a specific entity and then composes them.
Start Up System		Uses Entity Factory System to populate the entire game world.
World Objects Collision System	Player Family: • Player Component • Position Component • Bounds Component World Object Family: • Bounds Component • World Object Component • World Collision Object Component	This system is responsible for detecting collision with world objects, such as houses or trees, any object that logically cannot be passed through. These boundaries are separate entities, because objects like trees and houses are textures of a map.
World Wrap Up System	Player Family: • Player Component • Position Component • Bounds Component World Family:	Checks that a player is inside of a big polygon and prevents its leaving.

System Name	Family	Task
	• Tiled Map Component • Bounds Component	
NPC Collision System	Player Family: • Player Component • Position Component • Bounds Component NPC Family: • NPC Component • Bounds Component	Checks any collision with NPCs. If there is any collision and this NPC is the next in the story queue for a dialogue, then it starts the Conversation System (**Table 2**).
Sensor Collision System	Player Family: • Player Component • Position Component • Bounds Component Sensor Family: • World Object Component • Portal Sensor Component • Bounds Component • Sound Component	Checks collision with Sensors. If a sensor was collided, then it cleans up the world with the Clean Up System. Sets new player position based on the Portal Sensor Spawn point, loads another map file based on the sensor name and then populates the world with new entities. Also uses Sound System to play the door sound.
Pause System		It has similar behaviour like the Conversation System, it is also active periodically. It starts when a user presses the pause button, then it shows the pause menu (4.5.4).

Data: created by the author.

Table 4. *Debug systems*

System Name	Family	Task
Debug Camera System		This system allows moving the camera manually and zooming in and out.
Debug Render System	• Bounds Component	Renders bounds of entities which have Bounds Component

System Name	Family	Task
Grid Render System		Just renders grid to check a proper layout.

Data: created by the author.

The source code

The source code can be found by following this link:

https://github.com/vitalibassov/Interactive_Language_Teacher_Game

Installation instructions

Android version

1. Download the apk file from: https://github.com/vitalibassov/Interactive_Language_Teacher_Game/blob/master/installation%20files/android/android.apk
2. Copy the apk file to your Android device.
3. Find the file using a file manager on your device and run it.

Windows version

1. Download the exe file from: https://github.com/vitalibassov/Interactive_Language_Teacher_Game/blob/master/installation%20files/windows/windows.exe
2. Run the file and follow the installing instructions.

Author rights of the used assets

- Cat, frog, fish, and chicken sounds are licensed under the Creative Commons 0.
- Bee sound – the Attribution license 3.0 Unported. Author is “CGEffex”.
- Achievement sound (Earning score) – the Attribution license 3.0 Unported. Author is “Little Robot Sound Factory”.
- Finishing sound – the Creative Commons 0.
- Door sound – the license: https://www.freesoundeffects.com/licence.php.
- Step sound – the Attribution license 3.0 Unported. Author is “Little Robot Sound Factory”.
- Main theme – YouTube Audio Library. The Author is “Dan Lebowits”. The audio – “Parkside”.
- Graphical User Interface (Buttons and panels) – Public Domain license.

- Main menu screen background – permission for use only in this diploma thesis. Author is “RetroStyle Games”.
- Tile sets for the maps – the Creative Commons 0.
- The furniture – the Creative Commons 0.
- Controls (Cross-arrow) – the Creative Commons 0.
- The dog image used for the app icon and Main Character Monologues – the Creative Commons 0.
- Water – the Creative Commons 0.
- Big dictionary used in the game was found here: http://osinavi.ru/my/8000.php .
- Ivan Švaiger created all the animations, tree, and dialogue texts.