

IT-SEP4C-S18 – Serious Game

Project report

# The Frangovers

## 

## Karolina Beliharova

## Marek Löwy

## (253810)

## (253652)

# Supervisors

## Kasper Knop Rasmussen

## Jakob Knop Rasmussen

Table of Contents

[Introduction 5](#_Toc515473116)

[Requirements 6](#_Toc515473117)

[Analysis 7](#_Toc515473118)

[Use case diagram 7](#_Toc515473119)

[Use case description 8](#_Toc515473120)

[Domain model 9](#_Toc515473121)

[Design 10](#_Toc515473122)

[Implementation 11](#_Toc515473130)

[Test 12](#_Toc515473131)

[Results and discussion 13](#_Toc515473132)

[Conclusions 14](#_Toc515473133)

[Project future 15](#_Toc515473134)

[Source of information 16](#_Toc515473135)

# Abstract

*The customer is the Ensight Games, a Danish company that focuses on developing fun and educational games for use in companies as well as schools and universities. The company required a serious game for memorization training that will be developed for both mobile and pc platforms. The system is simple and can be used by anyone who wants to play the simple lightweight game. A user can choose between playing a random level or the specific one. After the user starts the game, he runs around the maze and finds statues and gets access to different minigames for memory training. After the user finishes the minigame he will get a fire point and can continue in the maze until he collects 5 point and can finish the level.*

# Introduction

A game which can be used for educational purposes or spare time enrichment, has been known since around 2600 BC. To achieve the meaningful game experience, games needed to be defined by rules, to be understood clearly by the players.

French sociologist Roger Caillois (Caillois, 1953) defined game as an activity that needs to have the following characteristics: fun, separate, uncertainty, non-productive, governed and fictious.

The first video games can be dated to the early 50s, when the technology became advanced enough for scientist to design simple games and simulations using electronic circuits. The discovery of CRT lead to tremendous rise of game development.

Video games have great educational potential in addition to their entertainment value. Games designed for specific problem, or to teach a specific skill have been very successful, since they are motivating, engaging, and provide rewards and chance to improve.

Learning by playing games encourage students to learn outside of class. There are no consequences, it is only a game that means if players lose, they can simply start the game over, try it again and learn from previous mistakes. This is not possible with grades at school, so it is not possible to correct mistakes. Games make learning more fun, and student will be more motivated to study and learn something.

The project presented is based on a requirement from the Ensight Games company which came with a demand for a serious game for memorization thinking. They focus and develop fun and educational games for use in companies as well as schools and universities. The purpose is to create a user-friendly game that would help people with training their short-term memory. It should be in a form that is both entertaining and appealing to young people.

Main features of the game must include simple controls and clear rules. The game must be able to run on both PC and mobile platform.

# Requirements

1. A user should be able to use application on both PC and mobile platform.
2. The system should contain procedurally generated mazes.
3. The system should have different types of memory games for brain training.
4. The system should contain statues.
5. A user should be able to move by using game controllers.
6. The user should be able to access minigame by statues.
7. The system should have pexeso minigame.
8. The system should have Simon says minigame.
9. The system should have hidden object in room minigame.
10. The system should have Sailor puzzle minigame.
11. A user should be able to launch the application through the menu.
12. The user should be able to exit the labyrinth through the exit in the centre.
13. The system should have smooth animations to change scenes.
14. The system should have a least one character.
15. The user should have view from third person.
16. The system should be optimized.
17. A user should be able to easily quit from the game.
18. The system should have a story.
19. The system should have nice skybox.
20. The system should have more levels.
21. The system should give instructions to the player.
22. The system should have sound effects.

# Analysis

## Use case diagram

The following figure represents the Use Case Diagram for the system based on the above mentioned important requirements.

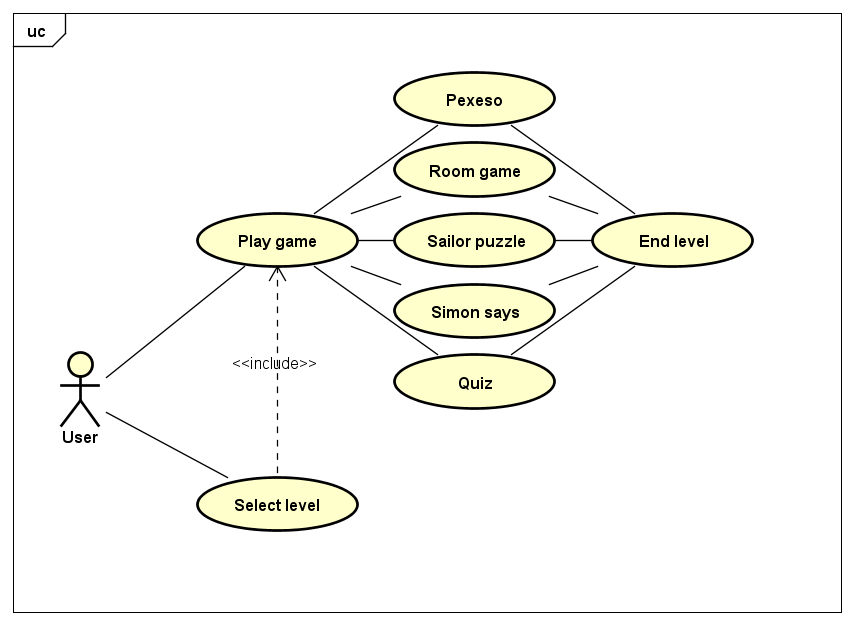


Figure 1: USE CASE DIAGRAM

When a user decides to play the game, the first level is loaded. In the game, the player navigates the labyrinth. While searching through the maze, the player encounters various statues, which lead to one of five minigames. One fire point is gained for completing each of the minigames. The player cannot end level until 5 fire points were collected.

## Use case description

From the use case description for Play Game, we can see that the user must first choose to play New Game in the main menu. When the New Game is selected, the first level is loaded. Afterward, the user plays the game, navigates through the maze and finds statues to launch the minigames. When the user collects five fire point, he can access the next level via temple.

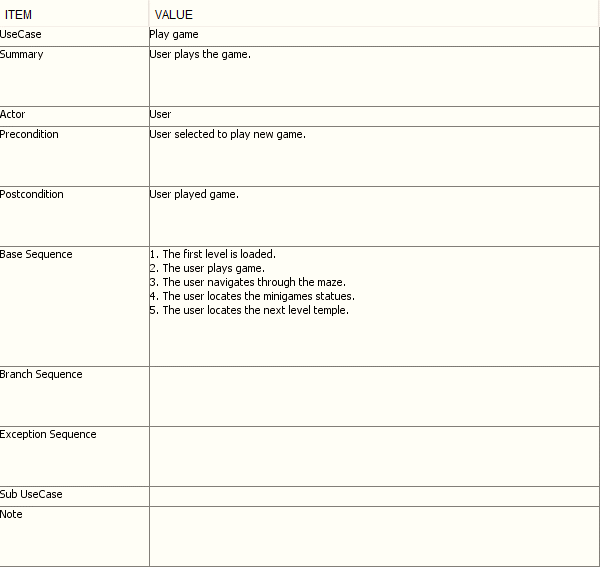


Figure 2: Use case description

## Domain model

The game will contain player, which will be controlled by the user and will contain the camera. The player can interact with statues, through which access to minigame is granted. Labyrinths will be generated using Maze Generator. The level contains Player, Maze, and the audio controls.

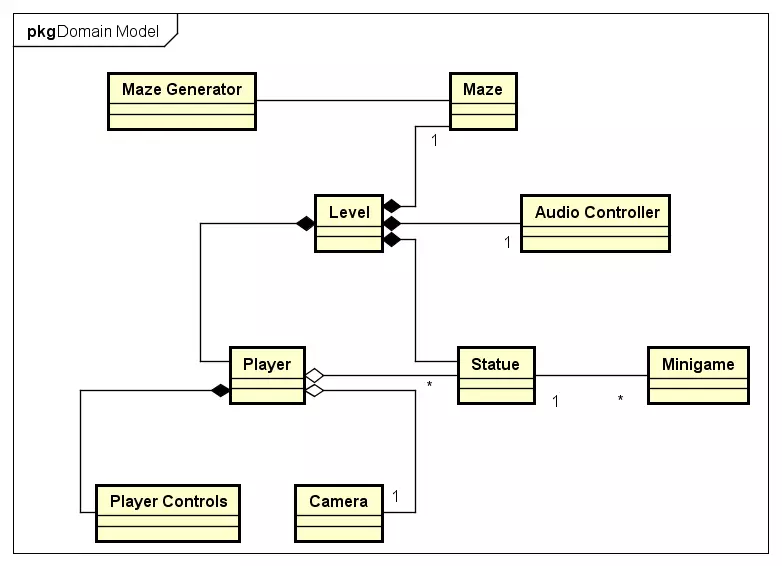


Figure 3: Domain model

# Design

We have decided to make a game, which will be appealing to young people and will convince them to train their memory. We will use Unity 5 to implement it since it is game engine we are most familiar with and will suffice our needs. We will make a 3D game with third-person controls. We have decided to do so because this type of games is quite popular in young generations and will help us in engaging them to play.

## Class diagrams

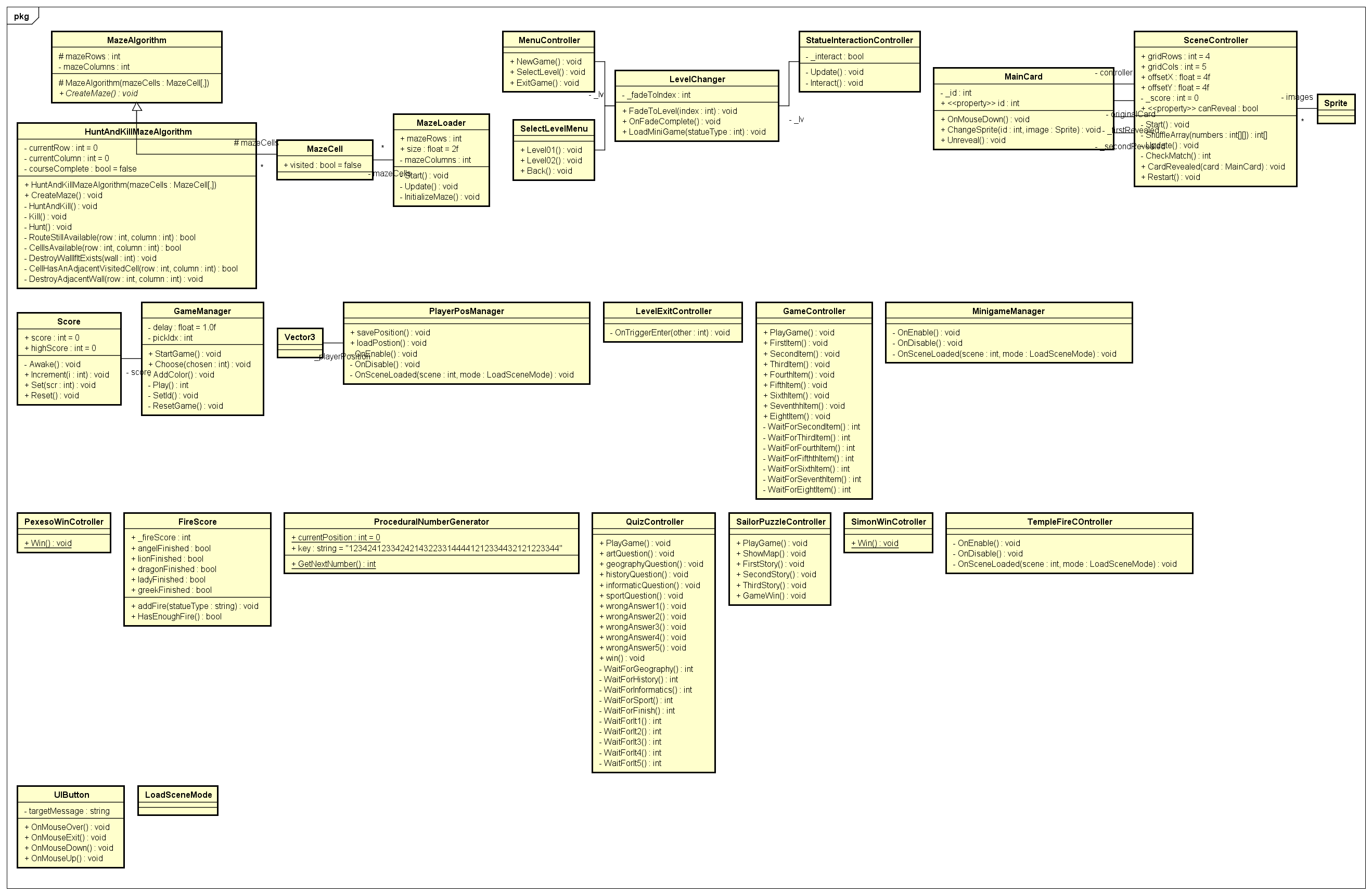


Figure 4: CLASS DIAGRAM

## Interaction diagrams

## Activity diagram

From the activity diagram for Play Game use case we can see that after navigating labyrinth and finding the statue, a user can interact with the statue which will lead to playing the mini-game. When mini-game is finished, fire point is received, and the user is back in the labyrinth. If the user has collected 5 fire points, the level can be finished and next one is loaded.

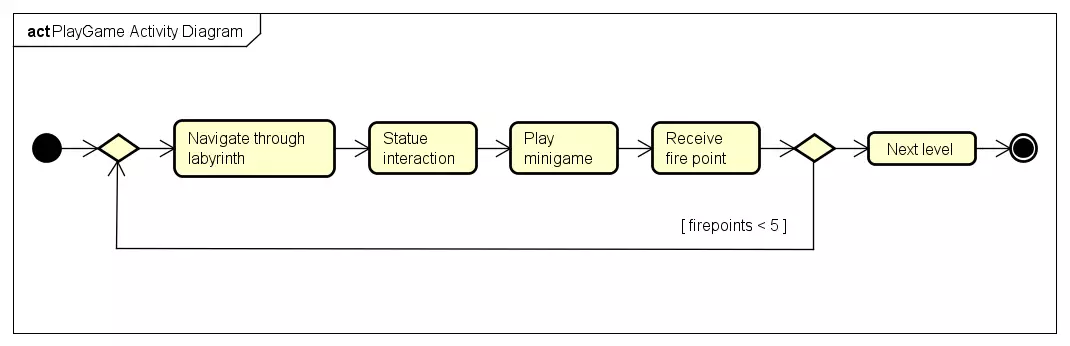


Figure 5: ACTIVity diagram

## Sequence diagram

From the sequence diagram for playing mini-game we can see that when game registers user input for interacting with the statue, a method to load mini-game is called on the level changer. The method takes a string with statues name as a parameter, using which it determines which mini-game to load. Before the new scene is loaded, player position is saved. When the game is finished, method Win() from win manager for a particular mini-game, which gives player one fire point and loads back the level. When the level scene is loaded, a previously saved player position is loaded.

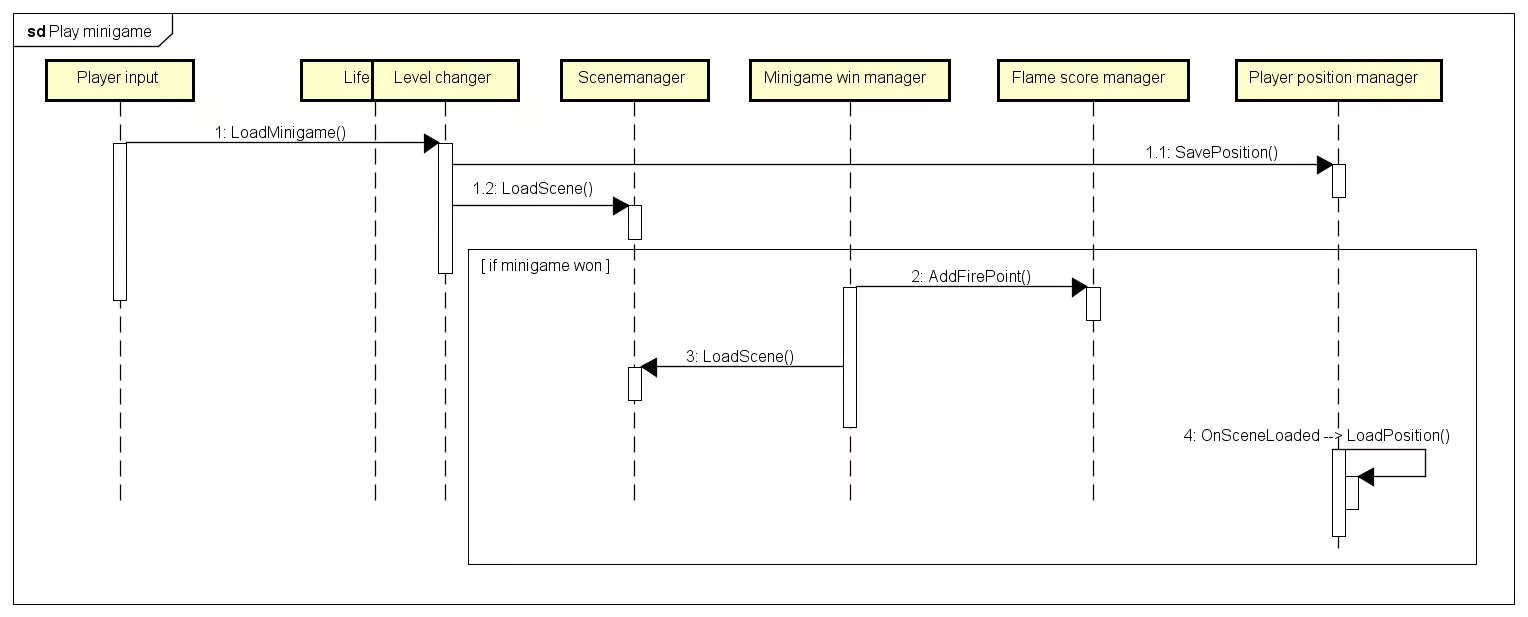


Figure 6: sequence diagram

## UI design choices

# Implementation

The game was implemented using Unity 5 and C# scripts. Most of scripts derive from MonoBehavior, which allows them to be parts of GameObjects.

Example 01: *StatueInteractionController*

* This component is responsible for interacting with statue when player input is given
* Method Interact first gets all objects with tag *Statue*
* If a statue is in range, method from LevelChanger is called to load the minigame

public class StatueInteractionController : MonoBehaviour {

[SerializeField] private LevelChanger \_lv;

private Transform \_player;

private bool \_interact;

void Update() {

if (Input.GetKeyDown(KeyCode.F))

Interact();

}

private void Interact() {

GameObject[] objs = GameObject.FindGameObjectsWithTag("Statue");

\_player = gameObject.transform;

foreach (var obj in objs) {

if (Vector3.Distance(\_player.position, obj.transform.position) < 4) {

\_lv.LoadMiniGame(obj.name);

}

}

}

}}

Example 02: *LevelChanger*

* This class is responsible for loading levels and minigames
* Method FadeToLevel takes build index of scene to load as argument
* Before loading next scene, animation to fadeout is triggered
* Minigame to load is determined by the statue name

public class LevelChanger : MonoBehaviour {

[SerializeField] private Animator \_animator;

private int \_fadeToIndex;

public void FadeToLevel(int index) {

\_fadeToIndex = index;

\_animator.SetTrigger("FadeOut");

GameObject go = GameObject.FindGameObjectWithTag("PosMan");

go.GetComponent<PlayerPosManager>().savePosition();

}

public void OnFadeComplete() {

SceneManager.LoadScene(\_fadeToIndex);

}

public void LoadMiniGame(String statueType) {

switch (statueType) {

case "Angel": {

FadeToLevel(SceneManager.GetActiveScene().buildIndex + 1);

break;

}

case "Lion": {

FadeToLevel(SceneManager.GetActiveScene().buildIndex + 2);

break;

}

case "Greek Man": {

FadeToLevel(SceneManager.GetActiveScene().buildIndex + 3);

break;

}

case "Screaming Lady": {

FadeToLevel(SceneManager.GetActiveScene().buildIndex + 4);

break;

}

case "Dragon": {

FadeToLevel(SceneManager.GetActiveScene().buildIndex + 5);

break;

}

default:

Debug.Log("Error");

break;

}

}

}

Example 03: *PlayerPositionManager*

* This class is responsible for saving and loading players position when minigames are loaded
* Position is stored as Vector3 and saved from LevelChanger when next scene is loaded
* When Level scene is loaded, method to load position is triggered

public class PlayerPosManager : MonoBehaviour {

private GameObject \_player;

private Vector3 \_playerPosition = new Vector3(0,0,0);

private Quaternion \_playerRotation;

public void savePosition() {

\_player = GameObject.FindGameObjectWithTag("Player");

if (\_player != null) {

\_playerPosition = \_player.transform.position;

\_playerRotation = \_player.transform.rotation;

\_player = null;

}

}

public void loadPostion() {

\_player = GameObject.FindGameObjectWithTag("Player");

if (\_player != null) {

if (\_playerPosition.Equals(new Vector3(0,0,0))) {

\_playerPosition =transform.position;

\_playerRotation = transform.rotation;

}

\_player.transform.position = \_playerPosition;

\_player.transform.rotation = \_playerRotation;

\_player = null;

}

}

void OnEnable() {

SceneManager.sceneLoaded += OnSceneLoaded;

}

void OnDisable() {

SceneManager.sceneLoaded -= OnSceneLoaded;

}

private void OnSceneLoaded(Scene scene, LoadSceneMode mode) {

loadPostion();

}

}

Example 04: *MinigameWinController*

* Each minigame has either WinController, or contains Win method somewhere in its main script
* Method Win grants a fire point to the player and then loads level from which minigame has been loaded

public class SimonWinCotroller : MonoBehaviour {

public static void Win() {

GameObject go = GameObject.FindGameObjectWithTag("FireScore");

go.GetComponent<FireScore>().addFire("Lion");

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex - 2);

}

}

Example 05: *TempleFireController*

* This class is responsible for checking whether the player has collected 5, or more Fire points
* When sufficient amount of point is collected, the lights and a flame in next level temple are activated, which allows player to enter and finish the level

public class TempleFireCOntroller : MonoBehaviour {

void OnEnable() {

SceneManager.sceneLoaded += OnSceneLoaded;

}

void OnDisable() {

SceneManager.sceneLoaded -= OnSceneLoaded;

}

private void OnSceneLoaded(Scene scene, LoadSceneMode mode) {

GameObject go = GameObject.FindGameObjectWithTag("FireScore");

FireScore fs = go.GetComponent<FireScore>();

if (fs.\_fireScore >= 5) {

Transform[] trans = GameObject.Find("Temple").GetComponentsInChildren<Transform>(true);

foreach (Transform t in trans) {

if (t.gameObject.name == "Lights") {

t.gameObject.SetActive(true);

}

}

}

}

}

# Test

The final version of the game was tested by controlling if all the requirements were accomplished.

|  |  |
| --- | --- |
| REQUIRMENT | STATUS |
| A user should be able to use application on both PC and mobile platform. | Zavřít |
| The system should contain procedurally generated mazes. |  |
| The system should have different types of memory games for brain training. | Znak zaškrtnutí |
| The system should contain statues. |  |
| A user should be able to move by using game controllers. |  |
| The user should be able to access minigame by statues. |  |
| The system should have pexeso minigame. |  |
| The system should have Simon says minigame. |  |
| The system should have hidden object in room minigame. |  |
| The system should have Sailor puzzle minigame. |  |
| A user should be able to launch the application through the menu. |  |
| The user should be able to exit the labyrinth through the exit in the centre. |  |
| The system should have smooth animations to change scenes. |  |
| The system should have a least one character. |  |
| The user should have view from third person. |  |
| The system should be optimized. |  |
| A user should be able to easily quit from the game. |  |
| The system should have a story. |  |
| The system should have nice skybox. |  |
| The system should have more levels. |  |
| The system should give instructions to the player. |  |
| The system should have sound effects. |  |

# Results and discussion

# Conclusions

# Project future

# Source of information

Why Use Games to Teach. *SERC* [ONLINE].

Available at: <https://serc.carleton.edu/introgeo/games/whygames.html>

Caillois, R. (1953). *Les jeux et les hommes.* Paris: Gallimard. Retrieved from Wikipedia

Available at: [https://en.wikipedia.org/wiki/Game#Roger\_Caillois](https://en.wikipedia.org/wiki/Game%23Roger_Caillois)

Playing video games is good for your brain – here’s how. *The Conversation* [ONLINE].

Available at: <http://theconversation.com/playing-video-games-is-good-for-your-brain-heres-how-34034>

Wikipedia. *History of video games*. [ONLINE]

Available at: <https://en.wikipedia.org/wiki/History_of_video_games>

Wikipedia. *Game*. [ONLINE]

Available at: <https://en.wikipedia.org/wiki/Game>

69 Trick Questions and Answers: How Many Can You Answer Correctly?. *Prisoner Of Class* [ONLINE].

Available at: <https://www.prisonerofclass.com/trick-questions-and-answers/>

Woody Woodpecker. In: *Wikipedia* [ONLINE].

Available at: <https://www.google.dk/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjP34HpyLzbAhXHfiwKHf1MDwAQjRx6BAgBEAU&url=https%3A%2F%2Fg.sina.re%2Findex.php%3Fq%3DoKipp7eAc2SWr3HmtNTZ4JzJqsRgwuHce-fP2bV-w-Cmp9G5jdja07esqdibpA&psig=AOvVaw0tj8-6LIJP8-Qk2ckf18lb&ust=1528289655515766>

*Ensight Games* [ONLINE].

Available at: <http://www.ensightgames.com>

Busina. 2015. Angel Statue. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/environments/fantasy/angel-statue-27594>

NVizzutti. 2017. Simon Says Minigame. [ONLINE]

Available at: <https://github.com/NVizzutti/SimonSays>

ChamferBox Studio. 2018. Greek Man Statue. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/props/discobolus-statue-107544>

VSIFY. 2016. Screaming Lady Statue. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/environments/dungeons/screaming-statue-60499>

Viacheslav Titenko. 2016. Dragon Statue. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/dragon-statue-59053>

organicpolygons. 2015. Lion Statue. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/props/exterior/lion-statue-34247>

Nobiax / Yughues. 2015. Pavement Material. [ONLINE]

Available at: <https://assetstore.unity.com/packages/2d/textures-materials/roads/yughues-free-pavement-materials-12952>

Thunderent's Assets. 2014. Temple Prefab. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/props/exterior/temple-props-19555>

OV. 2017. Skull Platform. [ONLINE]

Available at: <https://assetstore.unity.com/packages/3d/props/skull-platform-105664>

ProAssets. 2016. Skybox. [ONLINE]

Available at: <https://assetstore.unity.com/packages/2d/textures-materials/sky/free-hdr-sky-61217>

Invector. 2017. Third Person Controller. [ONLINE]

Available at: <https://assetstore.unity.com/packages/templates/systems/third-person-controller-basic-locomotion-free-82048>

Gameshard. 2017. Quiz Birdz. [ONLINE]

Available at: <https://assetstore.unity.com/packages/2d/characters/cute-birds-89649>

Unity Technologies. 2018. TextMesh Pro. [ONLINE]

Available at: <https://assetstore.unity.com/packages/essentials/beta-projects/textmesh-pro-84126>

Google Maps. 2018. Sailors Puzzle Map. [ONLINE]

Available at: <https://www.google.com/maps/@35.9470164,-5.5265056,145601m/data=!3m1!1e3>

Quiz Time. In: *Removing the Mask* [ONLINE]

Available at: <https://www.fromtpm.com/blog/2016/12/08/quiz-time-open-book-examination/>

Start. In: *Sell simple* [ONLINE]

Available at: <http://www.sellsimpleestateagency.com/wp-content/uploads/2016/09/start.png>

Wrong answer. In: *Clker.com* [ONLINE]

Available at: <http://www.clker.com/cliparts/7/e/1/6/1206557186603844069mcol_cross.svg.hi.png>

Colorful cartoon nature. In: *Storyblocks* [ONLINE]

Available at: <https://d2v9y0dukr6mq2.cloudfront.net/video/thumbnail/S15GBCm/colorful-cartoon-nature-background-with-space-for-your-text-or-logo-nice-sunny-day-with-some-clouds_ehseqyaig__F0000.png>

ICT Engineering, 2017 Project Report – VIA Engineering Guidelines

# APPENDENCIES

A – USE CASE DESCRIPTIONS

