The Fino Project - Game Mode

Janine D. Mayer[*]
S2010745015

Abstract

This paper deals with the design and development of an augmented reality (AR) application *"The Fino Project"* using the game development platform *Unity*. The developed application runs on mobile Android devices and allows the interaction with a small Dragon (also called "Fino"), where as the interaction possibilities are loosely based on the famous games *Tamagotchi* [3] and *Pokemon* [2].

1 Introduction

The resulting program can be divided into two parts. The first part, also called the "Tamagotchi" part, was made by my colleague Miss Helena Wilde and gives the user the opportunity to take care of a little dragon by feeding it, letting it sleep and similar actions.

This paper deals with the second part of the program, which is very slightly based on the 1996 Gameboy version of Pokemon, and allows the user to level up his dragon by interacting with other dragons and gems.
The dragon with which the player plays is also referred to as *Fino* in this paper.

2 Design

This section deals with the game design, which describes all the elements that occur in the game mode and interaction possibilities, as well as the assets used.

The idea of the game mode is to recreate the retro feeling of a Pokemon battle. In these battles, you could attack, weaken and ultimately defeat wild monsters with various attacks, or just plain defeat them. The Fino project takes a more casual approach to these battles. Instead of fighting physically, the player has the choice of calming down the opponent with jokes or tickling. If you manage to do this, you receive a gift.

A marker is placed on which either an opponent or a power stone (boosts experience, energy or health) is randomly generated which can be used for the player's adventure. The game mode is rather a gamble than a strategic game as the force of your attack is set randomly.

2.1 Gameplay

To switch from the main scene to the game mode you need the red button which is placed on a marker. When changing from and to the Tamagotchi scene, the energy value of the dragon is taken over. So Finos energy can be replenished by sleeping in the main scene.
The game mode uses the same target for the player's dragon as in the main scene (Tamagotchi scene). In addition, there is a marker which is used to generate an *adventure* (either opponent or gem).

[*]e-mail: S2010745015@students.fh-hagenberg.at

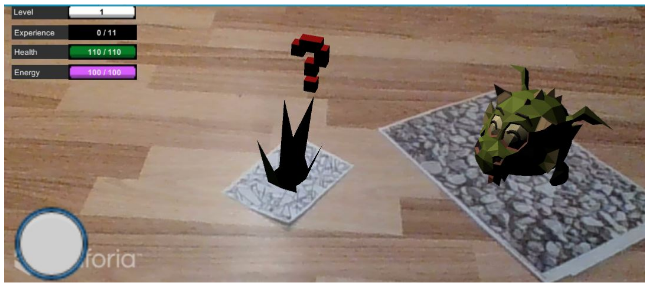
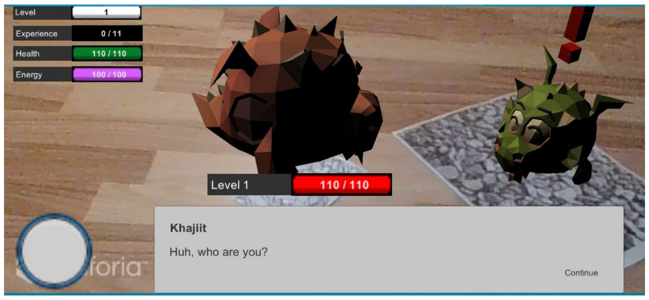


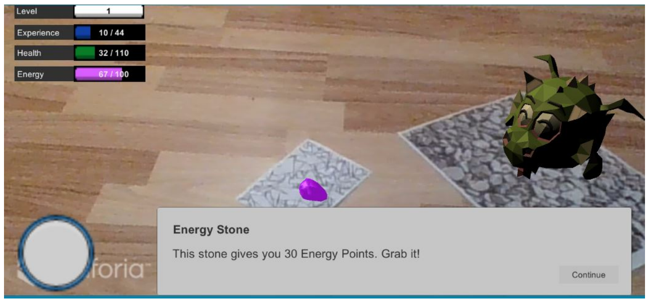
Figure 1: A questionmark above the vegetation shows that there is an action to make.



Khajiit

Huh, who are you?

Figure 2: Upon click on the vegetation a random object or enemy is generated.



Energy Stone

This stone gives you 30 Energy Points. Grab it!

Figure 3: After a won fight the player gets a reward. In this case an energy gem.

2.1.1 Player

Fino, the player's dragon, has three status bars that show his experience points, health and energy. In addition, he has a level, which is calculated on the basis of the experience points.

- **Experience Points** which are used to level up. They can be collected with every won fight and also via "Experience Gems" (blue).
- **Health Points** which are needed to survive. The Health points automatically regenerate during gameplay (but not during a

fight) but can also be collected through ”Health Gems” (green). The Fino loses health points when being attacked by an enemy.

- **Energy Points** which are needed to start counterattacks. Energy points automatically regenerate during gameplay (but not during a fight) but can also be collected through ”Energy Gems” (violet).

The Fino can be moved by means of a joystick in the lower left corner. A double tap allows the fino to make a loop.

2.1.2 Adventure

The adventure marker creates random vegetation in which various objects or enemies are hidden. If the Fino is within a certain distance of this vegetation, a question mark appears and the player can hear a rustling sound to indicate that something is hidden in this bush or tuft of grass. By clicking on the vegetation via the screen of the mobile device, either an opponent who wants to fight or a gemstone that boosts various status characteristics of the Fino appears.

A fight against an opponent can either be won or lost. If the player wins, a gem is spawned on the adventure marker. If he loses (health is 0), the opponent wins and the player is returned to the main scene to recover.

Enemies The generated enemies are also dragons which's level is within a range of 2 from the players level. The player can choose between the options ”tickle” and ”tell a joke” to appease the opponent's anger. The opponent, on the other hand, can attack the Fino physically.

Gems Gems are items that can be used to improve the status of the user. The colour of the stones matches that of the dragon's status bar to make them easier to identify. There are three different types as described below and seen in figure 4.

- **Experience Stone** used to boost the dragons experience points and make it easier to level up.
- **Health Stone** is used to heal the dragon.
- **Energy Points** is used to restore the dragons energy.

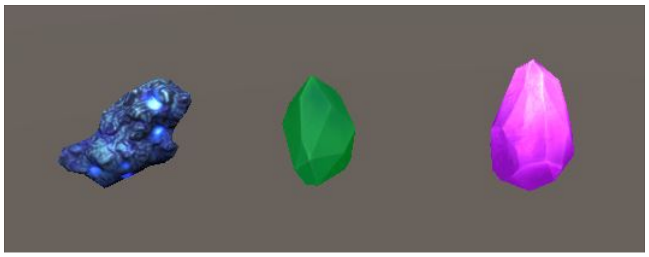


Figure 4: Gems, from left to right: Experience gem, health gem, energy gem.

2.2 Assets

Several assets were used for the design of the game mode. The dragons ”Finos” [4], the vegetation [6] and the gems [5] were downloaded from the asset store as models. The dragons and vegetation came with animations, with only the animations of the dragons being actively triggered in this program. For the choice of models, it was important that they are low poly so as not to throttle performance. In addition, all jokes [7] and sounds [1] were selected online and integrated into the game.

3 IMPLEMENTATION CONCEPT

This section outlines the implementation of the application and deals with the main components and the logic that runs in the background.

The application was implemented using the game development platform unity (version 2019.2.2.f1). In addition, the vuforia engine was used for marker tracking. The total project scope for the game mode consists of more than 1000 lines of code.

3.1 Classes

3.1.1 Character

CharacterBasic: This class is implemented by the Tamagotchi (project part Helena Wilde) and the Character class to ensure that methods for storing and retrieving data from the PlayerPrefs are implemented. This is important to ensure that information is not lost when switching between scenes (Tamagotchi scene and Game Mode scene).

Character: This class contains the most important information about the player's dragon. Here the status updates are made in form of a coroutine, which is stopped if you are in a fight. Many calculations happen, which are important for the progression of the level.

$$level = 0.3 * \sqrt{ExperiencePoints} \qquad (1)$$

This calculation is reversed to obtain the maximum number of experience points needed to reach the next level.

The maximum health is also calculated on the basis of the level (i.e. actually already achieved experience points) in order to let the maximum health increase aliquot to the level.

$$maxHealth = 10 * Level + 100; \qquad (2)$$

Enemy: The Enemy class serves primarily to facilitate interaction. It contains all the dialogues that the opponent can choose during a fight. It also has a maximum rage and a maximum and minimum attack strength. When instantiating an enemy object, dialogue options are randomly chosen for the new enemy (also the type of laugh). The level is chosen aliquot to that of the player, as well as the attack strength. Methods to generate and take damage are also given.

3.1.2 Items

GemStone: The GemStone class is used to manage the different gemstone types in a class instance. When instantiating, the dialogue matching the gemstone type is set as well as a random value for the gemstone.

3.2 Interactionlogic

3.2.1 Adventures

AdventureStates The AdventureStates is an enum which is used to manage the different states that the interaction with the Adventure-Marker can take. In Figure 5 the possible sequence of these states is displayed.

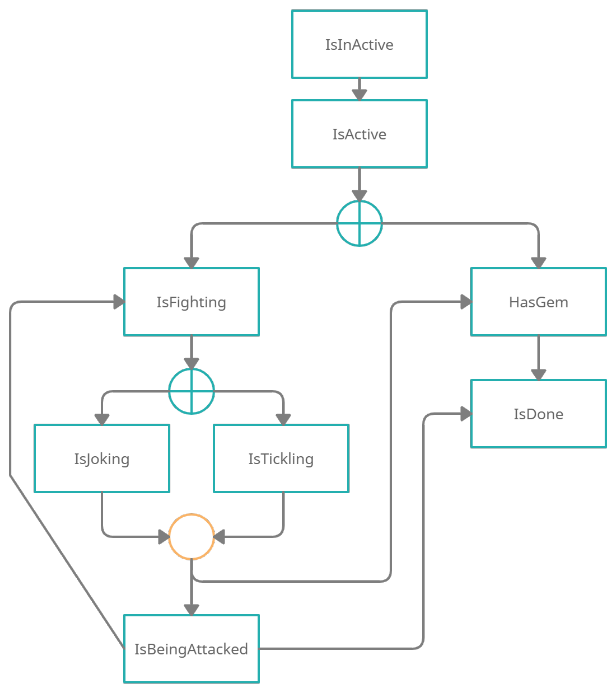


Figure 5: Possible sequences of the adventure states in a game

RandomAdventure This script contains most of the interaction between the player and the enemy or gem. It manages the timer, which acts as a countdown for the generation or unlocking of the next adventure. Here, either enemies or gems are randomly generated and the interaction with them is managed. In particular, the possibility of giving a dialogue a callback with a function to be executed is used here. At the end of an interaction, all used game objects are first destroyed and then a random vegetation for the marker is generated. Furthermore, the timer is set to show the player when the next interaction is possible.

AdventureAction This script is used by the VirtualButton on the Adventure Marker. It is used to track the interaction with the marker. This interaction takes the form of picking up gems or tickling the opponent.

3.2.2 Movement

Joystick This script is essential for controlling the Fino and is attached to the Joystick gameobject on the canvas. It implements the *IDragHandler* which implements *OnDrag*, *OnPointerDown* and *OnPointerUp*.

OnDrag clamps the position of the joystick on the pad so the stick cannot be dragged too far, while the position is used in a Coroutine to move the dragon.
OnPointerDown starts the Coroutine which moves the gameobject and is fired on pressing down the joystick.
OnPointerUp stops the Coroutine and setsback the joystick to its original position.

Movement This script was originally thought to contain the whole movement logic. It now only contains the logic to trigger the loop animation for Fino on a double tap and allow movement via a keyboard.

SwitchScene This script is attached to the VirtualButton which helps switching between the Tamagotchi scene and the Game mode scene. Before loading the next scene the script takes care of saving important data to the PlayerPrefs.

3.3 UI

3.3.1 Dialog Management

Dialog The Dialog class is a simple class to contain the name of the speaker as well as all sentences which can be displayed.

DialogManager The DialogManager acts as a service that starts dialogues and can switch from one sentence to the next. To write the sentence, a coroutine is used to make the dialogue appear letter by letter.

The *StartDialog* method also takes a callback, which makes it possible to execute a given method after the end of the dialogue. This is an important part of the code, as these callbacks enable interaction between the player and opponents.

Animations The animations for the appearance and disappearance of a dialogue were created with the help of the animator itself.

3.3.2 Other

Timer The timer script creates a timer from a prefab, which attaches itself to a given object. The timer can be started and stopped externally and is attached to the canvas.

There are further scripts (MarkerRotation, Statusbar) to help rotate or keep track of the gameObjects they are attached to.

4 User Guide

To start the game, you have to switch to the game mode scene (Extended) with the red VirtualButton. Here you can use the same marker as in the main scene to represent Fino. Another marker is used for random interaction. With the markers, you have to make sure that they can always be captured well. It is best to have them in a foil or laminated to avoid bending them. If a target cannot be perceived immediately, it can be picked up and used closer to the camera. Interaction requires both the screen of the mobile device (to move the dragon, click on actions, click on dialogue, tap on vegetation or double tap to loop the dragon) and interaction via virtual buttons (tickle the dragon, pick up a gem, switch from/to the main scene).

The game can be played indefinitely, as it is only about levelling up your Fino. Interaction with an opponent is won when their Anger (red bar) is empty. It is lost if the Fino's health points are empty. In this case, you are sent back to the main scene.

5 Conclusion

This project is a small presentation of how (Augmented Reality) interactions can be realised in Unity with the help of Vuforia. The use of virtual buttons help provide a wide range of interaction possibilities, while markers help ”bringing games to life”.

Over 40 hours of design and development went into this project. The selection of features to be implemented is huge and holds out the prospect of further development of the project.

Acknowledgments

In the course of the Augemented Reality course at the *University of Applied Sciences Upper Austria*, we were given the opportunity to develop this AR application for mobile devices.

The author thanks FH-Prof Dr. Anthes for the introduction to Augmented Reality, Unity and Vuforia and also Unity for not crashing too much

References

[1] V. Artists. Various sounds. FreeSound.
[2] V. Authors. Pokémon. Wikipedia.
[3] V. Authors. Tamagotchi. Wikipedia.
[4] BitGem. Micro dragon fino - faceted style. Unity Asset Store.
[5] ClayManStudio. Toon crystals pack. Unity Asset Store.
[6] Elcanetay. Low poly nature - free vegetation. Unity Asset Store.
[7] Various. 101 short jokes anyone can remember. Reader's Digest.