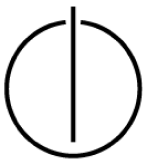
DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics: Games Engineering

Development of a Turn-Based Battle System for a Serious Game for Learning Japanese

Robin Fischer



DEPARTMENT OF INFORMATICS

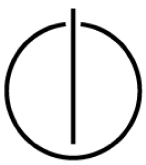
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics: Games Engineering

Development of a Turn-Based Battle System for a Serious Game for Learning Japanese

Entwicklung eines Runden-basierten Kampfsystems für ein Serious Game zum Erlernen der japanischen Sprache

Author:	Robin Fischer
Supervisor:	Prof. Gudrun Klinker, Ph.D.
Advisor:	Dr. David A. Plecher
Submission Date:	15.09.2021



I confirm that this bachelor's thesis in informatics: games engineering is my own work and I have documented all sources and material used.

Munich, 15.09.2021 Robin Fischer

Acknowledgments

I want to thank my advisor David Plecher for giving me this exciting topic for my thesis. He always provided me with useful feedback and advice.
I am also very grateful to my family and friends, who provided me with useful feedback.

Abstract

Japanese culture and products are getting more and more popular around the world. Therefore, the number of people interested in learning Japanese is increasing. But especially for people that start without any prior knowledge about Japanese kanji, learning the language can be tedious. To effectively learn kanji, they have to be reviewed many times over a long time. The game "Dragon Tale" is a Serious Game for learning Japanese. As a useful extension to the current game, a turn-based battle system is suitable for providing an entertaining way to review Japanese kanji and words. First, related work was analyzed, which was useful for finding general design guidelines. With those, a rough design of the battle system was created. After that, the implementation was started by first creating a class diagram of the whole application. After the development was completed a user study was conducted. Due to current circumstances, a proper formal user study could not be performed, but a design of such a study is proposed in this thesis. For getting feedback on the user interface and for gathering suggestions, an informal user study was conducted.

Contents

1. Introduction

Nowadays video games are a major part of the entertainment industry [Mam]. People use them to experience rich stories, seeking a challenge, or for relaxing. While playing video games people can enter a state where they forget about their surroundings and are fully immersed within the game. A topic that is getting more popular is Serious Games, where the benefits of this immersed state are tried to be utilized to accomplish a second goal. This could be the learning of a new language. Information on the language can be included within the story of the game. Then by letting the player recall this information in challenging game mechanics, developers of the Serious Game hope to ease the process of learning the new language for the player. A suitable language to learn with a Serious Game is Japanese. For people with no prior knowledge of Japanese kanji, the learning of the Japanese writing system is especially problematic. In modern Japanese, more than 2000 kanji have to be mastered [Con13]. This is the reason the learning process can be tedious, and a Serious Game could instead offer an entertaining way for learning the language. The Game "Dragon Tale" tries to accomplish this, and the turn-based battle system that will be developed for this bachelor thesis will extend "Dragon Tale".

1.1. Motivation

Even in Western countries, the Japanese language has a presence. Big international companies originated from Japan like Sony or Toyota [Son; Toy]. The company Nintendo also was found in Japan and played a crucial role in the history of the gaming industry. In 1889 beginning with selling playing cards the company was founded and since then Nintendo distributes games all over the world [Nin]. Many well-known games like the ones from the "Mario" franchise were developed by Nintendo. The company is also one of the biggest game console manufacturers, besides Sony with the "PlayStation" series and Microsoft with the "Xbox" series [Art].

Summing up the revenue the biggest gaming companies made in total and then compared by country, Japan ranks place two. USA's companies make the most revenue [PIL]. Besides the gaming industry, Japan has a significant presence in the automobile industry, the animation industry, and the electronics industry. Therefore learning the

Japanese language can yield benefits within the working market. Furthermore, fans of Japanese media can enjoy the works in the original language.

For Japanese learners there exist the JLPT (Japanese Language Proficiency Test). It ranks the Japanese proficiency of a person in five levels (N1 - N5), while N1 is the highest and N5 is the lowest level. Comparing the number of people taking the JLPT between 1984 and 2018, more and more people are taking the test [The]. In that time interval, most people have taken the test 2018, where the participants' number was over one million. Furthermore, the company Duolingo, developer of a popular language-learning platform states in their 2020 report that Japanese is the sixth most learned language on their platform [Duoa]. Considering all these facts, one could say the learning the Japanese language is popular and therefore a market exists for Serious Games that help in easing the learning process.

"Dragon Tale" is a game where the player can experience a rich story and while progressing through the game, the player learns different kanji and Japanese words. These have to be recalled to solve different puzzles in the game. To increase the frequency of reviewing the Japanese language knowledge, the game needs a different game mechanic. "Dragon Tale" features RPG-like (Role-Play-Game-like) elements and various RPG-like games that originated from Japan feature a turn-based battle system. Games from the "Pokemon", "Persona" and "Final Fantasy" franchise for example. These games have been well received in the gaming market and therefore it is assumed that a battle system fits into these kinds of games. Developers of other Serious Games for learning Japanese seem to have come to the same conclusion because their games frequently feature a turn-based battle system. In chapter 3 some of these games will be introduced.

1.2. Problem Statement

In the previous section, the motivation behind choosing a turn-based battle system for learning Japanese is explained. A conclusion about the effectiveness is tried to be made at the end of this bachelor thesis.
The main part of the bachelor thesis is the development of a suitable turn-based battle system. There are two main requirements; it should fit into the base game "Dragon Tale" and it also should feature the learning of the Japanese language. Figuring out how to include the learning of Japanese into the game is part of the bachelor thesis. Furthermore, the turn-based combat itself has to be designed and implemented. The following issues have to be addressed.

- How is the battle system integrated into Dragon Tale?
- How is a battle initiated?
- What choices does the player have in each turn?
- How is the Japanese language included in the turn-based combat mechanics?
- What influence does the player's Japanese knowledge have on the outcome of the battle?
- What kind of outcomes are possible after a fight, and how do these influence the player?

The implementation of the battle system will be done with the Unity3D engine. For coding the programming language C# will be used with the IDE Visual Studio 2017.

1.3. Outline

The following list describes the chapters of this bachelor thesis.

- In chapter 2 provides the theoretical background.
- The analysis of related work is done in chapter 3.
- The battle system is based on the game "Dragon Tale". A short introduction to this game is given in chapter 4.
- The main part of this bachelor thesis is the explanation of the developed battle system. In chapter 5 information about the design and the implementation of the battle system is provided.
- A informal user study was performed after the development of the battle system was finished. Chapter 6 covers this user study and also proposes a formal user study.
- In Chapter 7 some outlook is given regarding future work that can be done for the battle system to improve it.
- In the last chapter 8 a conclusion is given.

2. Theoretical Background

For the development of a Serious game for learning Japanese, some theoretical background has to be understood. In this chapter in the first section, some information about language learning is given. Afterward, the second section teaches basic knowledge about the Japanese language, focusing on the writing system. In the last section, the term Serious Game will be explained.

2.1. Language Learning

Learning a language consists of various skills that have to be mastered. The first step is to learn basic vocabulary and grammar, which then can be used to read and write texts in the learned language. Another skill is the speaking of the language, which also includes the knowledge of how to pronounce different words within a specific sentence. From a psychological standpoint there exist two important factors that influence the learning process. These are called motivation and attitude, which are explained in the first section. And the second and third section teaches about the importance of repetition and context for language learning.

2.1.1. Motivation and Attitude

A key aspect of learning a language is the motivation and attitude for learning the language. Without the initial motivation, a person will never start learning a new language. Furthermore, motivation is the driving force to keep the learning process running even if it gets tedious in the later stages. [OV11] Gardner defines the term motivation within the context of second language learning with the following words: "Motivation refers to the extent to which the individual works or strives to learn the language because of a desire to do so and the satisfaction experienced in this activity" (Gardner [Gar85] cited in [OV11]). This definition differentiates between intrinsic and extrinsic motivation. First is the motivation to learn the language because of the benefits having the skill provides, which is the extrinsic motivation [Hen14]. The second type of motivation is the enjoyment during the learning process [Hen14]. This is called intrinsic motivation.

Another crucial factor for language learning is having a positive attitude towards learning the language in question. The term attitude is defined by Smith as "a relatively enduring organization of beliefs around an object or a situation, predisposing one to respond in some preferential manner" (Smith [Smi71] cited in [OV11]). To differentiate attitude from motivation one can think of attitude as a set of beliefs and motivation is the reasoning behind one's actions. [OV11]

2.1.2. Repetition

When learning a certain grammar, word, or expression of a language one first must comprehend it. After that, the repetition of the learned is crucial to maintain the knowledge and reduce the time needed for recalling the knowledge. One of the reasons repetition is important for the learning process is because a frequently occurring and thus more familiar word can be expected to have a stronger memory representation in the brain than less frequently occurring words. [Kim17]

2.1.3. Context

In the previous section 2.1.2 the importance of repetition was explained, but learners still need to study how to use the vocabulary in specific sentences.
For learning vocabulary, a common and popular method is using flashcard applications. Users try to memorize the words by frequently repeating them. To improve the understanding of the word context-based learning can be used. Yamaguchi describes a context-based learning mobile application, which should ease the process of vocabulary acquisition. By performing experiments where the context-based application was compared to a non-context-based flashcard application, it was found out that context has a significant advantage. [Yam+20]

2.2. The Japanese Language

In this chapter, some basic knowledge about the Japanese language is provided. Comparing Japanese to English neither the writing system nor the grammar has many similarities. For Japanese beginners, the learning of the Japanese writing system can be an issue, because it is complex. Three different kinds of characters are used, hiragana, katakana, and kanji. While hiragana and katakana can be compared to the Roman alphabet, because each letter represents a sound, Japanese kanji do not represent a specific sound and therefore can not. Each kanji represents at least one meaning and has at least one readings. In a single Japanese sentence, all three different types of characters may be used. [BOS11, p.24]

2.2.1. Hiragana

There exist 46 basic hiragana (平仮名) characters and their appearance is roundish. In Japanese sentences, hiragana characters can have different purposes. The most basic case is that there are Japanese words that are usually written in hiragana. These kinds of words have to originate from Japan. Furthermore, each Japanese word has a reading, which can be written in hiragana, even if it is normally written with kanji. Therefore when writing a Japanese text, each kanji can be replaced with its hiragana reading. There also exists the option to write the reading in hiragana above the kanji. These are called "furigana" (振り仮名) [AAPb], and they help people that do not know how to read the kanji below the furigana. There also exist Japanese words that are a combination of kanji and hiragana. When a Japanese word starts with a kanji and ends with a hiragana it is called "okurigana" (送り仮名) [AAPc]. hiragana also has different use cases in Japanese grammar. For example, each verb in Japanese ends with a hiragana, and conjugating it changes the hiragana ending, while the kanji stays the same. [BOS11; Con13]

n	w-	r-	y-	m-	h-	n-	t-	s-	k-		
ん	わ	ら	や	ま	は	な	た	さ	か	あ	**a**
		り		み	ひ	に	ち	し	き	い	**i**
		る	ゆ	む	ふ	ぬ	つ	す	く	う	**u**
		れ		め	へ	ね	て	せ	け	え	**e**
	を	ろ	よ	も	ほ	の	と	そ	こ	お	**o**

Table 2.1.: The basic hiragana characters [BOS11, pp.24f].

Table 2.1 shows the basic hiragana characters, but there are 23 additional sounds and therefore 23 additional hiragana in Japanese. These are obtained by adding diacritic marks (Dakuten or Handakuten) and then the sound of the hiragana slightly changes. For example, applying the Dakuten to は(ha) it becomes ば(ba) and applying the Handakuten it becomes ぱ(pa). By combining two hiragana more different sounds can be represented. When these combinations are written down the first hiragana is always written with normal size and the second one is always smaller. For example, りょ stands for "ryo" while りよ for "riyo". Furthermore, there is the small っ that can be put before any other hiragana. The pronunciation of the following hiragana is then more emphasized. [BOS11, p.24-25]

2.2.2. Katakana

Katakana characters look more angular than the hiragana characters. They look similar to certain kanji characters because they are based on some of these kanji. They are mostly used for loanwords or foreign names. A typical loanword from the German language is "アルバイト"(arubaito) which is based on "Arbeit". But the meaning is slightly different, it means "part-time job". Table 2.2 shows the basic katakana characters. For every hiragana character, there is a katakana counterpart. The Dakuten and the handakuten, which are introduced in the hiragana section above, can also be applied to katakana characters, as well as the combination of two katakana and the small ッ. Additionally only applicable for words written in katakana ー is used to stretch the pronunciation time of the katakana that is written before. For example, コーヒー is pronounced "koohii" instead of kohi and translates to "coffee". Because words that are written in katakana often represent a foreign loanword, the range of sounds that can be written in katakana is more than in hiragana. For example, there is no "fa" hiragana but it can be written in katakana with "ファ". [BOS11, p.24, 28–30]

n	w-	r-	y-	m-	h-	n-	t-	s-	k-		
ン	ワ	ラ	ヤ	マ	ハ	ナ	タ	サ	カ	ア	a
		リ		ミ	ヒ	ニ	チ	シ	キ	イ	i
		ル	ユ	ム	フ	ヌ	ツ	ス	ク	ウ	u
		レ		メ	ヘ	ネ	テ	セ	ケ	エ	e
	ヲ	ロ	ヨ	モ	ホ	ノ	ト	ソ	コ	オ	o

Table 2.2.: The basic katakana characters [BOS11, p.28].

2.2.3. Kanji

Over 1500 years ago the Chinese characters (漢字) were introduced to Japan. Until then Japanese didn't have a writing system. The introduction of hiragana and katakana into the writing system was done later. The Ministry of Education from Japan has selected a total of 2136 commonly used kanji that Japanese students learn until they graduate from high school. Although most of these kanjis are already taught until the end of middle school and therefore Japanese people learn most kanji over a period of nine years. ([BOS11, p.30]) Every kanji has a meaning and at least one possible reading. Readings for kanjis are categorized into Onyomi(音読み) and Kunyomi(訓読み). The Onyomi reading is derived from the Chinese and the Kunyomi reading is the native

Japanese reading [.] Genki1 Figure 2.1 displays the kanji for fire with all it's components (1-4). The writing of the kanji with the correct stroke order (1), the meaning of the kanji (2), the Kunyomi-reading/s (3), and the Onyomi-reading/s (4). For each stroke, there is also a defined start- and endpoint. [Con13, p.11]

Figure 2.1.: Kanji with the meaning "fire" [AAPa].

2.2.4. Grammar

It would be too much to explain the Japanese grammar in detail, but some useful basic facts will be provided in this section. The knowledge about Japanese grammar was acquired by the book "A Guide to Japanese Grammar" by Tea Kim [Kim15] and the integrated coursebook "Genki" part 1 [BOS11].
In contrast to English, a Japanese sentence only needs a verb to be complete. So the directly translated sentence "Go." is grammatically correct in Japanese. To understand the sentence's meaning, context is needed. For example, if the topic of the conversation is currently about the field trip that will be performed next week, then the sentence "Go." would be correctly translated into "I will go to the field trip". Furthermore, there exists no verb "to be" in Japanese. So a sentence like "I am Peter" is not written with a verb. So besides all the verbs there exists the declaration of things, which is done with "だ"(da). To extend the statement from before, a Japanese sentence only needs a verb or a declaration to be complete. So the sentence "Peter da" is correct and could mean according to context "I am Peter". But it's generally a declaration of a person named "Peter", which could also mean "Peter is the one you're looking for". "da" declares a statement with the highest certainty possible, and because politeness is an essential part of the Japanese language, "da" is often omitted in spoken Japanese. To make a declarative sentence more polite "です"(desu) is used at the end. So a common way to introduce yourself in Japanese in a polite way is "Name + です." [Kim15; BOS11]
To make some more complex sentences, like including a subject, an object, a time, a place, and so on the Japanese grammar uses the so-called "Particles". The ordering of the parts of a sentence is not important, except the verb or declaration must be placed at the end of the sentence or sub-clause. For declaring different parts of a

sentence the particles are used. A particle always comes after the sub-clause it is used on. So declaring a location in a sentence is first writing the location followed by the location-particle. An example sentence is "I will go to Tokyo", which translates to "東京に行く"(toukyouu ni iku). "東京"(toukyouu) is the word for Tokyo and に(ni) is the particle that declares a location. At the end of the sentence is "行く"(iku) which is the verb for "to go". [Kim15; BOS11]

2.3. Serious Games

In this section, the term "Serious Game" is introduced. First, a definition is provided, followed by an explanation of why a "gamified" application is not a Serious Game. Then the concept of "Flow" is explained, which is a term that originated from psychology.

2.3.1. Definition

A serious game is a game that not only serves the purpose of entertainment but also benefits the player's health or education (Dörner [Dör+16] cited in [APK20]). The main purpose is still to entertain, while the "serious" part of the game is an additional goal. Therefore a serious game should still conform with the definition of a game. Chandler defines a game as "A play activity defined by interactive challenges, discernible rules, and attainable goals" [Hea11].

In the context of serious games, one often encounters the term "Gamification". Gamification is defined as the adoption of playful or game-like elements into a non-playful situation [Bib]. In contrast to a serious game, the application does not fall into the definition of a game. Within the area of language learning, many platforms use the concept of Gamification. One of these platforms is "Duolingo". In the chapter 3 more information on these platforms is provided.

2.3.2. Flow

Flow is a psychology term that refers to a state of mind with the following characteristics. High focus on the activity one is doing at the moment, action and awareness are merged, the loss of the reflective self-consciousness, the feeling of being in full control and that one can manage every obstacle in his way, a feeling that time passes faster than usual, and the activity itself is rewarding enough. To enter a state of Flow the activity one is doing has to comply with certain conditions. One condition is that the challenges one has to perform should neither be too hard nor too easy. [CN02, p.240]

The level of difficulty should be appropriate to one's capabilities. Figure 2.2 visualizes this. The right amount of difficulty is achieved by balancing the amount of challenge to the usable skills. Another condition is that the goals of the activity should be clear and the feedback about the progress must be immediate [CN02, p.240]. Utilizing the state of flow can ease the learning of a language like Japanese. The battle system that will be developed for this bachelor thesis tries to accomplish this.

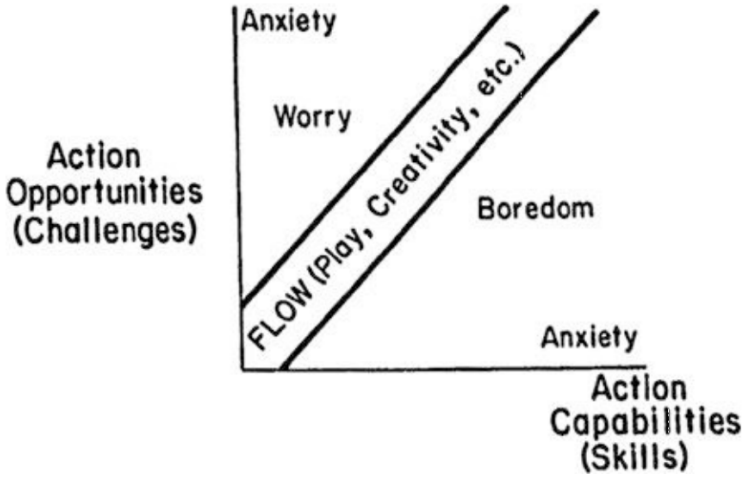


Figure 2.2.: Visualization of the Flow Channel [CN02, p.212].

3. Related Work

In this chapter already existing language learning applications are analyzed. Starting with gamified language teaching platforms like Duolingo. Followed by Serious Games that try to accomplish the same goal, helping the player to learn Japanese. In the last section, an application named "Wanikani" is introduced that does not fall in the category of gamified applications nor Serious Games.

3.1. Gamified Language Teaching Platforms

The following applications are using the concept of Gamification to make the learning process easier by motivating the user with the adoption of game-like elements. These applications can not be considered games, because there is no playful situation.

3.1.1. Duolingo

Duolingo is a platform that teaches various languages in a gamified way free of charge. It splits the language into chapters, which is displayed in figure 3.1. Each chapter covers a certain group of vocabulary that are related to each other. For example, the chapter "Home 1" in the Japanese course teaches vocabulary that occurs at home like "table". Furthermore, each chapter also teaches a set of expressions or grammar associated with the current topic. For example, the chapter "greetings" in the Japanese course teaches various expressions like "Hello", "Nice to meet you" and so on. Figure 3.1 displays a screenshot that was taken of the chapter overview from the Duolingo website. [Duob]

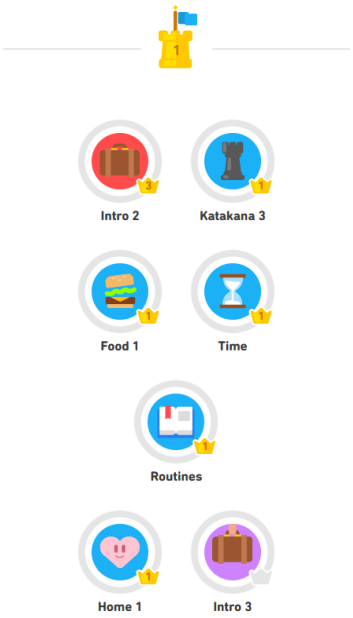


Figure 3.1.: Screenshot of Duolingo's chapter overview [Duob].

After starting a chapter the user is confronted with different types of tasks. These vary in their difficulty and therefore the easier tasks will be performed first. The following list gives examples of frequently used tasks. [Duob]

- Match the pairs: The user is given multiple-choice options in both languages. The user then has to decide which of these cards in each language represent the same meaning. After matching every pair correctly the task is finished. Figure 3.2 shows the task. For the Japanese version of this task, a kanji is matched with its reading in kana, and reading in kana is matched with its romaji writing.
- Select the correct answer: The user is given a set of answers and has to decide which one is correct.
- Sentence Puzzle: The user has to form a grammatically correct sentence with a set of given puzzle pieces, where each represents a part of the sentence. These pieces have to be selected in the correct order to finish the task.
- Writing with a keyboard: The player has to translate the given sentence by typing the answer into a text field.

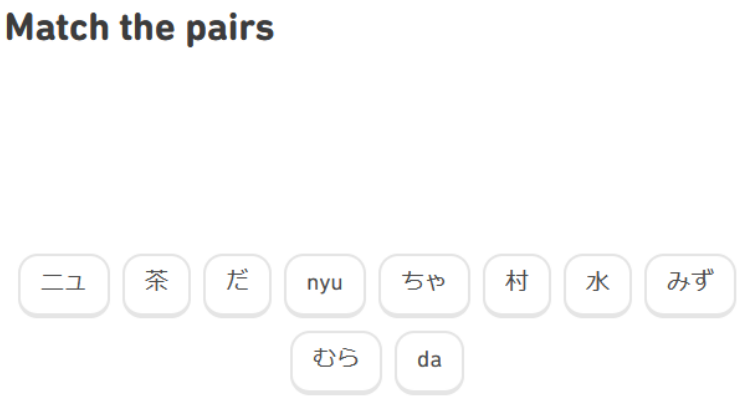


Figure 3.2.: Screenshot of Duolingo's "Match the pairs" task [Duob].

After the user finished a task a notification about the correctness of his answer appears. Duolingo also has a voice-over for each sentence or vocabulary, which helps the user to get familiar with the spoken language from the beginning. [Duob]

3.1.2. Lingodeer

Multiple platforms offer a similar service to Duolingo. One of them is Lingodeer. In this platform, the lectures are also divided into topics like it is done in Duolingo. Lingodeer also offers a diversity of different tasks for the user. Multiple tasks that are found in Duolingo, are also used in Lingodeer, but there are differences. Also, the overall design is different. For example, the "Match the Pairs" equivalent in Lingodeer separates the two matching parts into two groups (left and right). Figure 3.3 is a screenshot showing the difference to figure 3.2. [Zhu]

Figure 3.3.: Screenshot of Lingodeer's "Match the pairs" task [Zhu].

The name of another similar platform is Babbel, which will not be further analyzed in this bachelor thesis. [Bab]

3.2. Serious Games

The following list of applications includes serious games for learning Japanese. Especially the use of a battle system within the game is often used in Japanese learning games.

3.2.1. Slime Forest Adventure

There is a free demo version and also a non-free full version of Slime Forest Adventure. The difference is that the free version only offers the learning of kana characters, while the full version also offers the learning of Japanese kanji. The following information was gathered by play-testing the free version of the game. Information on the game and the download links are available on the website. [Joh]

"Slime Forest Adventure" is an RPG that also features a battle system. The player needs to save "the princess" which leads him into a dungeon. The dungeon is full of slime enemies attack randomly while walking within the dungeon. When the player is in a battle he can defeat a slime by typing in the reading of a given kanji or kana in the form of romaji. While he is thinking the slime enemies attack him, which reduces his health. So the faster the player can perform the Japanese task the better he performs in

the fight. After the player has defeated all the enemies, he will gain experience points and gold. Gold is the currency within the game, which can be used to buy items, that are useful for fighting the slimes. There are different types of slime enemies. Each type represents a group of kanji or kana. While fighting a certain slime type one kanji or kana of his group is then used as the Japanese task. With this technique, the player's overall learning level is decided by the number of types of enemies he has already faced.

Figure 3.4.: The battle system of "Slime Forest Adventure" [Joh].

3.2.2. Koe

The game "Koe" is a game developed by Strawberry Games. It is an RPG that offers the learning of the Japanese language. It combines the RPG elements including a turn-based battle system with communicative language learning techniques. Within the battle system, each vocabulary represents a different attack move with its distinct animation. The animation helps to memorize the vocabulary. For example, after selecting the attack which is represented by the word for sakura, a storm of flying sakura blossoms is used as the animation for this attack. The player does not have to perform a Japanese task, but knowing the Japanese words for the attack helps the memorization process for each attack the player can perform. Furthermore, after selecting a word, its reading is displayed with kana characters. [Vala]

Figure 3.5.: The battle system of "Koe" [Vala].

3.2.3. Learn Japanese To Survive! Kanji Combat

Learn Japanese To Survive! Kanji Combat is a game of a series that teaches the player the Japanese language with RPG-like elements including a turn-based battle system. Each enemy represents a kanji. The appearance of the enemy is either the symbolic representation of the kanjis writing or its meaning. The player has to select the correct meaning or reading to deal increased damage to the enemy. [Valb]

Figure 3.6.: The battle system of "Learn Japanese To Survive! Kanji Combat" [Valb].

3.3. Wanikani

Wanikani is a platform mainly for learning Japanese kanji and vocabulary. Their content is grouped by a level which is called the "Wanikani-Level". The higher the user's Wanikani-Level is the more Japanese kanji and vocabulary the user has already learned. Each level consists of three phases. First, learn a certain amount of kanji radicals. Second, use the learned radicals to study kanji which include one of the previously learned radicals. The last step is using the already known kanji to learn Japanese words that can be written with these kanji. The progress of the user is monitored and used for providing suitable review lessons. Figure 3.7 shows the Dashboard of a user currently having Wanikani-Level two. When clicking on the top-right corner located button "Reviews", the review session is initiated. New lessons can be started by clicking on the top-left located button "Lessons". [Tof]

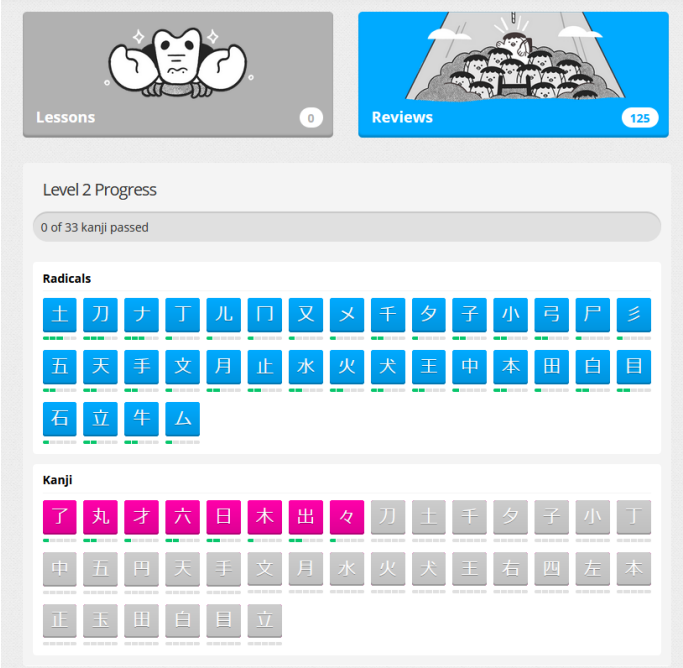


Figure 3.7.: Screenshot of Wanikani's dashboard [Tof].

Wanikani uses a text field as input, where the user has to type in the correct answer. The review lessons switch between each category(Radical, Kanji, Vocabulary) randomly. Different background colors indicate the category of the current question. There exist radicals, kanji, and words with equal symbolic representation. The distinct background color helps in avoiding confusion. [Tof]

4. Dragon Tale: The Base Game

The battle system developed for this bachelor thesis will be an extension of the game "Dragon Tale". The following chapters give an introduction to the game and explain the reason why a turn-based battle system is suitable.

4.1. Game Concept

Dragon Tale is a Serious Game for learning Japanese. It fits into the adventure game genre and also uses RPG elements. No specific target audience is determined for Dragon Tale because learning a new language does not depend on characteristics like gender and age [Özt20, p.19]. To reach a higher number of people that can play the game, it was developed for Android devices. It is playable on a smartphone or a tablet. The language of the game is English, which results that the potential target audience is narrowed down to English-speaking people that are also interested in learning Japanese.

The player can freely move around in a 3D world where he experiences a story that includes Japanese mythology. A quest system leads the player through the story, where the player learns different Japanese kanji in situations that fit the kanji's meaning. For example, there is a quest requiring the player to look into a kettle containing cooked meat. After the player performed the task, the game introduces the kanji representing the meaning "meat". The player increases his knowledge of Japanese kanji as the story progresses, which he later has to apply for solving puzzles. There are different types of puzzles within the game. A frequently occurring one is where the player has to select the correct kanji in a given situation. For example, the player gets the task of letting a plant grow. The game gives him a set of possible kanji, and he needs to select the correct one suitable for this situation. To accomplish this, he is required to choose the kanji that represents the meaning "up". For some puzzles, the player has to use AR-based game mechanics. In section 4.3 these kinds of puzzles are presented.

4.2. Story

The protagonist of the story is a girl named "Yuni". The story begins in a small village, where Yuni finds a mysterious egg at the beach. Later the dragon "Ryu" hatches from this egg. Yuni then embarks on a journey to find the dragon's parents. She interacts with the dragon by using the "ancient language". The dragon has various spells at his disposal, which can be used to accomplish different tasks. Before Yuni leaves the village a fire breaks out. She overcomes the crisis by using one of the dragon's spells to extinguish the fire. Eventually, Yuni embarks on her journey to find the parents of the dragon. On her adventure, she has to face various difficulties. But she overcomes those with the help of the dragon and increasingly learns more about ancient language along the way. The so-called "ancient language" is the Japanese language. The game currently introduces approximately 30 different kanji to the player.

4.3. AR Puzzles

Currently, Dragon Tale features two different puzzles that involve the use of AR. Both require the user to work with AR-markers. The AR application can detect these markers and then lays a 3D model above them. For example, if the application detects the marker for "fire", it places a 3D model of a campfire over it. One of the puzzles uses the concept of the kanji compounds to build a Japanese word. In the Japanese language, words often consist of two or more kanji. The compound meaning of these often gives a hint on what the word means. The puzzle uses this concept to let the player find the correct compound to form the word in question by placing two markers in the appropriate order besides each other. Figure 4.1 displays an example of a correctly solved compound puzzle. The puzzle requires the player to build the word "fireworks", which is a compound of first the kanji for "flower" and then for "fire". The player has placed the correct markers in their proper order. Then the AR application places a flower and campfire above the corresponding kanji. The green line signalizes that the puzzle was solved correctly.

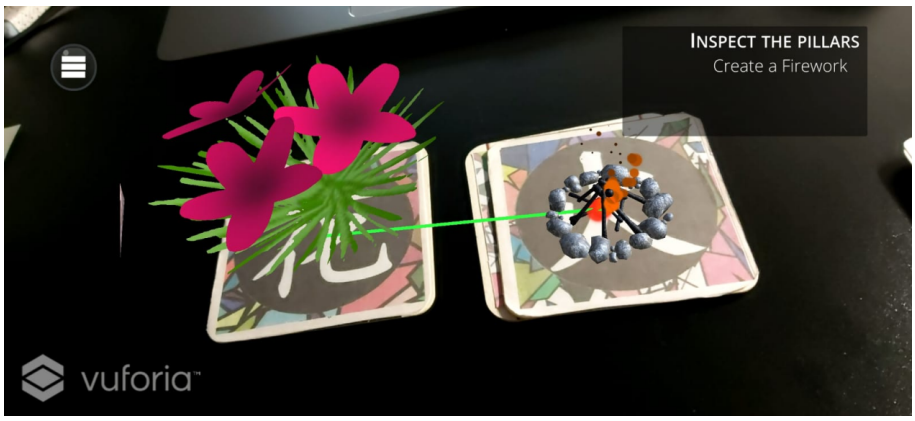


Figure 4.1.: AR Puzzle in Dragon Tale [Özt20].

The second puzzle involving AR is a counting puzzle. The player has to bribe the antagonist's paid soldiers to win a fight. Each soldier wants a certain amount of money, and the player has to place markers for kanji representing a certain number next to each other. When the player places the correct amount of money with a set of kanji markers, then the puzzle is solved.

4.4. Suitability of a Turn-Based Battle System

In "Dragon Tale" the player gets introduced to various kanji as the story progresses. The player has to memorize the meaning of these kanji to use them later in puzzles. In the current version of "Dragon Tale" the number of occasions where the player must review the kanji is limited. Increasing the number of puzzles is challenging because a puzzle must be unique. Furthermore, solely knowledge about the meaning of the learned kanji is needed to solve these puzzles. But for mastering Japanese kanji it is also required to know how to read them. A kanji-reading is problematic to include in contextual puzzles, therefore another game mechanic is required.

A turn-based battle system can be considered as an additional serious game mechanic. As explained in section 1.1, it is assumed to fit into the RPG genre and is frequently used in related works, which also offer the learning of the Japanese language. It can be used to increase the frequency of reviewing and can additionally feature the learning of kanji-readings. Therefore a turn-based battle system has the potential to be a useful addition to the game "Dragon Tale". But the specific design of the battle system still has to be figured out. For finding a suitable design, all the related work introduced in chapter 3 will be analyzed.

To summarize, the objective is to design a battle system suitable for the game "Dragon Tale" with the help of design guidelines gather from related works. Playing the game could substitute the tedious reviewing of Japanese kanji and vocabulary.

5. The Battle System

For this bachelor thesis, a turn-based battle system for learning Japanese kanji will be developed. This chapter covers this battle system with all its components by providing information on the design process and also explaining how these were implemented.

There are three fixed requirements for the battle system. First, it should provide turn-based combat. Second, the player should be able to learn Japanese with it. Third, the battle system must fit into the game "Dragon Tale" (A introduction to this game gives chapter 4). With these requirements as the basis, the development process was started by analyzing related works (See chapter 3). Using the gathered information, the approximate design for this battle system was documented. Then a coarse class diagram according to this design document was created. Then the implementation phase was started. The game engine Unity3D was used and the programming code was written in the programming language C#.

Not only the different aspects of the battle system have to be designed, but also the software itself. Desirable characteristics for the software are modularity, extensibility, and maintainability. The reason for this is that after the part of the battle system for this bachelor thesis will be completed, it will be integrated into the "Dragon Tale". Different developers are working on that game and a modular software design will reduce the amount of work needed for the integration. Furthermore, with a higher level of extensibility, the battle system will be able to be extended by further features with fewer difficulties. One way to accomplish these characteristics is two create a program code with low coupling and high cohesion. Having low coupling means that different modules of the software should not rely on various other parts of the software that do not belong to the module [SK12]. High Cohesion means that the different parts of a module must have a close relation to each other, which means that a module should not include functions that barely relate to the purpose of the module [SK12].

Furthermore, two architectural patterns were used for programming the software. First, the overall structure of the software is based on the MVC (Model View Control) pattern. It is an architectural pattern that divides the program code into three parts: model, view, and control ([MVC] provides more information on the MVC pattern). For

programming, the view part of the software, a layered architecture was used, and the Facade design pattern was applied ([Hau] provides more information on the Facade pattern). A rough class diagram of the whole application can be found in the appendix. More detailed class diagrams will be shown in the corresponding section of this chapter. All the class diagrams have been created with the tool "Apollon" [APO].

5.1. Initiating a Battle

In Dragon Tale the player can move around freely in a three-dimensional world. The player has to move to specific locations to fulfill quests. With a battle system included, a proper way of starting a fight has to be designed and implemented. One way could be to define areas within the game world where a battle is randomly initiated while walking in the area. With that approach, the player does not have the option to avoid fights while walking within these areas. To give the player more control over when to start a battle and when to not, a different approach is needed.

One way to accomplish this design goal is by not defining areas with a certain probability to get into a battle but instead define a location where a battle awaits. If the player walks within a specified range to these locations, a battle will begin. This method also has disadvantages. Because the battle locations have a static position, the player can avoid them by finding a way around. These battle locations define where enemies are located, and therefore this approach harms the game's immersion. But an immersive world with a rich story is part of the traits that "Dragon Tale" tries to hold. Therefore, the following three design extensions will be used. First, an animated 3D model of the enemy represents the battle location. These enemies can walk around freely in an individually determined area. With the enemy's location changing, the battle location changes accordingly. The last design extension is letting the enemies follow the player after the player was detected. This increases the level of immersion and the difficulty of avoiding fights. The implemented result is visualized in figure 5.1.

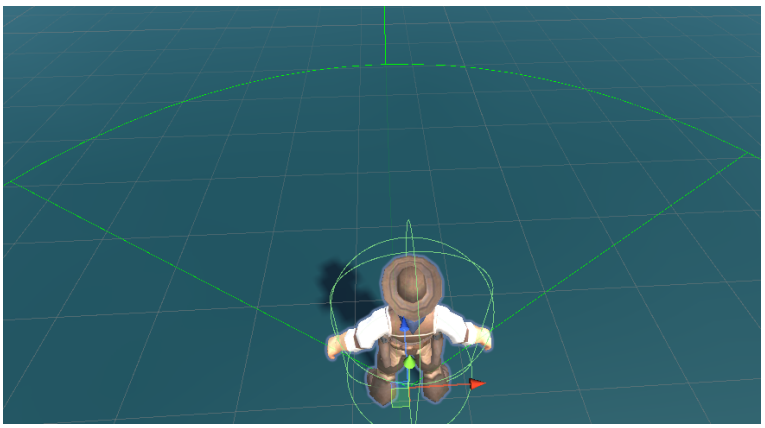


Figure 5.1.: Visualization of the collider and the view angle of an enemy (green lines).

In the figure, an enemy is shown, who was placed in a 3D world. The green lines are gizmo visualizations, which are not visible in the actual game. The outer circle around the enemy displays the maximum distance where the enemy can spot the player. The two lines that start from the enemy's center and continue until the circle is intersected, visualize the maximum view angle of the enemy. If the player is within range and maximum view angle of the enemy, he will be spotted and followed by the enemy. The following speed of the enemy is higher than the walking speed. This gives the player a feeling of being chased by the enemy. In the figure, there is also a visualization of a capsule collider around the enemy. If the player comes into contact with this collider the battle will be initiate.

5.2. The Battle Scene

The battle scene is where the turn-based combat takes place. After the player comes into contact with an enemy, the load screen is shown. As soon as the battle scene is loaded, the load screen will disappear and the completely initialized battle scene is displayed. Figure 5.2 shows the battle scene directly after the loading is completed.

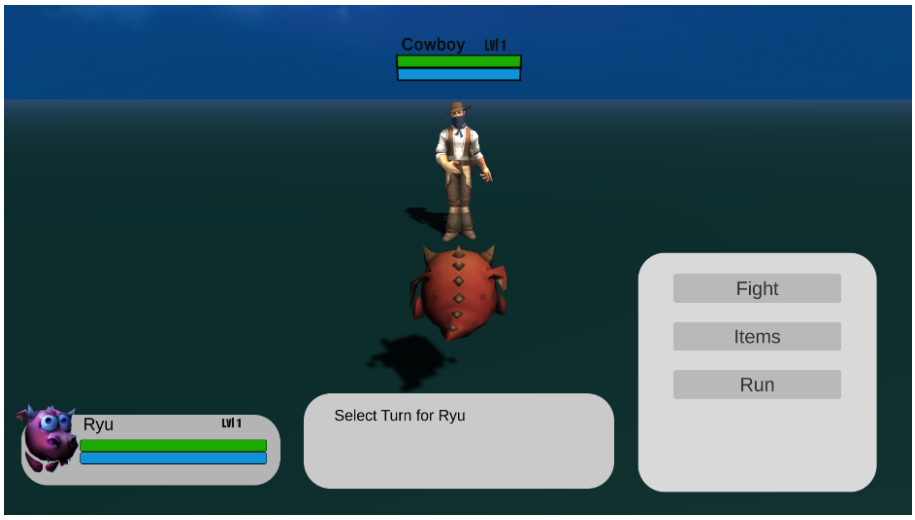


Figure 5.2.: The Battle Scene

In the 3D scene, all the battle participants are instantiated as animated 3D models. In the figure, there is one ally "Ryu" and one enemy "Cowboy". Most UI input elements of the battle system are implemented as HUDs (Head-Up-Display). Each battle participant has a status-HUD. It displays the current health, current SP (Skillpoints), and currently active buffs. In section 5.4 these three terms in respect of the battle system are explained. For enemies, the status box is placed directly above the 3D model. For the allies, a HUD at the bottom-left of the screen canvas is used. A dialogue box is located at the bottom center of the UI canvas. It provides textual feedback on what is currently happening within the battle. It also gives the player information on what to do next like "Select Turn for Ryu", which is displayed in figure 5.2. For issuing commands like selecting a certain skill a battle menu with multiple buttons and menu states is used. Within the other sections of this chapter, a more detailed explanation is given to these scene elements. There are also parts of the UI that are not visible in figure 5.2. These will be explained in later sections of this chapter as well.

5.3. Battle Procedure

After the player walks within a certain range to a "battle location", a battle will be initiated. Each battle consist of three phases. The Initialization phase, the turn-based combat phase, and the battle result phase. The component "BattleManager" handles all three phases.

5.3.1. Initialization Phase

This phase serves two main purposes. The first is the setting up of the battle scene. This includes the instantiation of all the battle participants at the correct locations. For this purpose, the location where the battle takes place has to be defined. The place could be located on a different scene or within the currently loaded scene at a "save" location. In this context, a "save" location means that there is no possibility that other game objects can interfere with the battle. Such a location can be included in every level of Dragon Tale. Because of that a scene switch is not needed and therefore not used in the current implementation of the battle system.
The second purpose is to set up all the needed references. Each battle participant has a component of the type "BattleBehaviour" and a component of the type "BattleData". In sections 5.4 and 5.6 these two components are explained. After obtaining the references to the battle data of all participants, the UI canvas is initialized with the values contained from this data.

5.3.2. Turn-Based Combat Phase

In this phase, the actual gameplay takes place. In turn-based combat, each battle participant first selects a turn, and then the turns are executed in a certain order afterward. There are two parties in a battle, the allies, and the enemies. A battle ends after one party defeats the other or if a battle participant successfully escapes from the fight. Figure 5.3 displays the routine for handling each turn-based combat round.

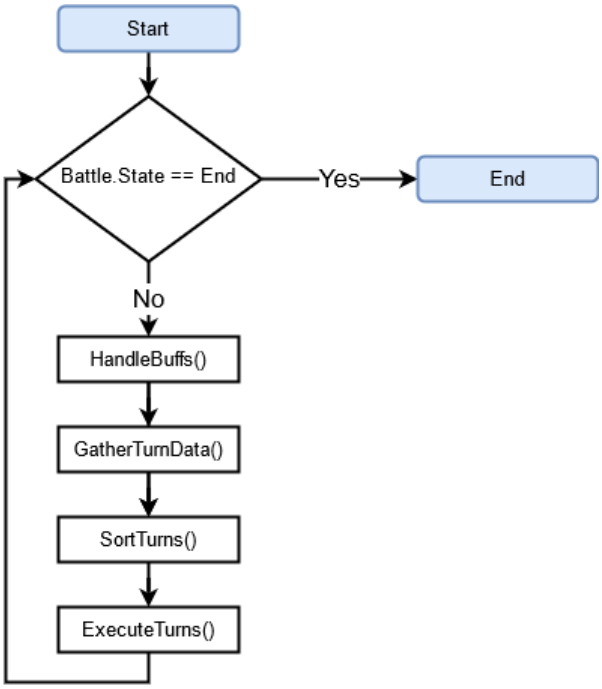


Figure 5.3.: A flow chart of the battle routine.

Before the beginning of each round, it is checked if the battle state was set to "End". If this is the case, the turn-based combat phase will end. If this is not true the following four steps will be executed in the given order. At the beginning of each turn, all the currently active buffs will be handled. A buff is a status effect that has a certain number of rounds of which it is active. So "HandleBuffs()" decreases the lifetime of all buffs and removes a buff from the battle participant if its lifetime is zero. The next step is calling the Coroutine "GatherTurnData" which gathers all the turn data from the participants. In Section 5.6 a more detailed explanation will be given for this topic. After the turn data is collected it has to be sorted, because the order of execution is not the same as the order of gathering the turns. In section 5.4 will be explained how the turn execution order is determined. The last step is the execution of each turn in the previously set order. Each turn has exactly one performer but can have multiple receivers. How a performer performs and a receiver receives the turn is handled in their "BattleBehaviour" component. How it is handled is explained in section 5.6.

5.3.3. Battle Result Phase

A battle can end whenever a turn was executed. Currently, there are two possible ways to end a battle. The first one is that one party was defeated after the turn, which means that all battle participants belonging to this party have no health left. The other possible way is if one battle participant successfully escapes from the battle. From the perspective of the player, there are three possible outcomes. The player wins the battle, the player loses the battle, and the undetermined victor outcome. The battle result phase deals with each of these three scenarios. More information on how the battle result is managed gives section 5.8.

5.4. Battle Data

Each battle participant needs a reference to a "BattleData" object. It holds all necessary information for the battle-turn selection and battle-turn execution. Furthermore, this data is the basis for various functions executed in the "BattleBehaviour" component, which is explained in section 5.6. In figure 5.4 a class diagram is shown of the battle data implementation.

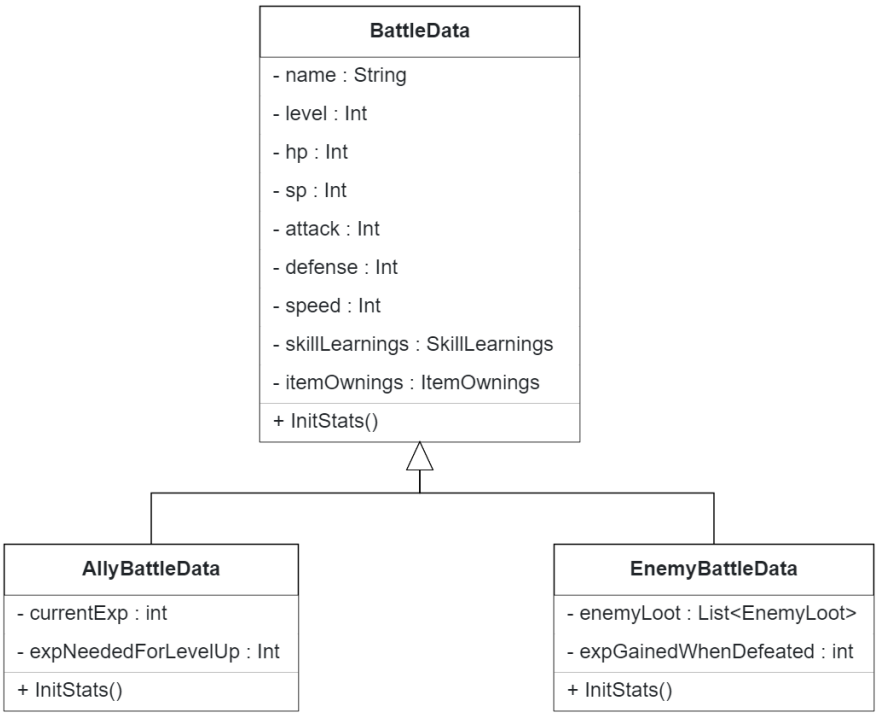


Figure 5.4.: A simplified class diagram of the "BattleData".

The base class "BattleData" includes all the information needed for turn-based combat. Each battle participant needs at least this data for the turn execution. Two classes specialize in the implementation of "BattleData" by inheriting from that class. "AllyBattleData" additionally includes data needed for handling the allies of the player and "EnemyBattleData" includes data needed for handling the enemies that the player fights against.

5.4.1. Level

The level of a battle participant is a measure for determining its strength in a battle. A level is an absolute value, which only increases after a certain amount of Exp (Experience Points) are gathered by winning battles. Allies can increase their level but enemies have a fixed level. Within the "AllyBattleData" there are two attributes that handle this. The currently gained experience points (currentExp) and the experience points allies needs for increasing their current level (expNeededForLevelUp). After the current points are equal to the needed points, the level will be increased by one and the current points will be reset to zero. For determining how many experiences

points the ally will receive after defeating an enemy, each enemy has an attribute "ExpGainedWhenDefeated", which is stored in his "EnemyBattleData" object.

5.4.2. Stats

The term "stats" is a common word used in the context of battle systems. It refers to parameters that are used for different calculations within the combat. For the battle system developed in the scope of this work, there are five different parameters. The value of a stat increases with the level of the battle participant. For the allies of the player, the current value of a stat needs to be saved in a permanent save file. This can be accomplished in different ways. A possible solution would be to save the value for each stat in two different attributes. The initial value defines the base value for this stat and the current value. For each battle participant, a base value has to be defined, and with this approach, a change of the base value will not change the current value of the stat, because calculations were based on an older version of the base value. To avoid this problem the actual implementation only defines the base value. When the application is started or the level of the ally changes, the method "InitStats()" will be called. This method uses the base values of all stats and the current level to calculate the current value of all stats. For the calculation, an exponential increase in value is applied. The battle system offers some parameters for the developers to adjust the exponential curve.

Hp and Sp

HP (Health Points) represent the health of a battle participant. For the skill execution, an extra amount of points is needed. These points are called SP (Skill Points). Each skill has a defined number of SP needed for performing the skill. A battle participant can not perform a skill if the number of his SP is insufficient for the skill execution. Compared to other stats, Hp and Sp need an extra value, because there is a difference between the full and the current value. During the turn-based combat, the battle participant can increase or decrease the current value, which is not the case for the other stats like "attack". For increasing the other stats during the combat, buffs are used.

Attack, Defense and Speed

These three values are needed for the turn execution. In each round of a turn-based battle, there exists a certain order in which the different turns are executed. The speed value is the factor that directly influences this order. The higher the speed value of the battle participant, the higher the chance of performing one's turn before others.

The attack and defense value are factors needed to calculate the received damage. When a battle participant performs a move that causes the receivers to take damage, this damage is increased by the performer's attack value. When a receiver of the turn receives damage, the actual damage is reduced depending on the defense stat of the receiver.

5.4.3. Skills and Items

Within a battle, each battle participant can perform different moves. There are three moves that each participant can perform without a limit. A standard attack move which deals moderate damage to a single target. A standard defense buff, which increases the performer's defense for one round. And the option of trying to escape from the battle. To make the gameplay more versatile skills and items are introduced. The current version of the battle system features different parameter options for the creation of skills and items.

Skills

Each battle participant can have a certain number of skills. A skill can be seen as a special attack move a battle participant can perform. With skills included in the battle system, the developer using the battle system has the opportunity to make the gameplay more diverse by defining a variety of different skills for allies and enemies. Each battle participant has a reference to an object of the class "SkillLearnings", which handles their skills.

When an enemy or an ally is created the developer has to determine their skills. Each skill has a minimum level required for using the skill. The combination of skill and level is called a "Skill Learning". Each battle participant has a reference to an object of the type "SkillLearnings", that handles the participant's skills. With a minimum level requirement, the player's allies have the number of usable skills increased with the player's progression through the game. Another advantage is that enemies can also have different skills depending on their level. And this leads to more versatile gameplay because enemies seem to adapt while the player is progressing through the game. When determining the skill for a "Skill Learning" a string key has to be defined. This key is used as a reference to a database entry where detailed information about the skill is stored. This kind of implementation has the advantage of not having to define skills multiple times for different battle participants. A skill within the database has the following attributes.

- name: The name of the skill, which is also shown in the battle menu for selecting the skill.
- cost: The amount of SP needed to perform the skill.
- speed: The speed of the skill. Multiplying the speed of the performer of the skill and the speed of the skill itself is used as the turn's execution speed. The turn with the highest speed will be executed first.
- battleEffect: A Reference to an object of the type "BattleEffect". Defines what impact the skill has within the turn-based battle.

Items

Each battle participant possesses a set of items, that are usable within battles. An item is a consumable which is useful within a battle. For example, an item named "Health Potion" can be included in the game, which restores the health of the user like the name suggests. An object of the class "ItemOwnings" manages all the currently owned items for a battle participant. Like skills, items also have a database for storing detailed information about them. When an item owning is defined a string key is used as the database entry reference. The second attribute that has to be defined is an integer that represents the number of times the item is owned. An item within the database has the following attributes.

- name: The name of the item, which is also shown in the battle menu for selecting the item.
- battleEffect: A Reference to an object of the type "BattleEffect". Defines what impact the item has within the turn-based battle.

When creating an enemy, the developer can define their so-called "loot". "Loot" is a term that describes what kind of items the players can receive after defeating an enemy. Each enemy has an individual list that consists of objects from the class "EnemyLoot". This class has two attributes. First is "Rate", which defines the probability that the player will receive the item after the enemy is defeated. And second, there is a string attribute for the item key, which can be used to access the database.

5.5. Turn Execution

For each round in a turn-based battle, each participant selects a move and a target. In the current version there exist two different ways of selecting a turn. Either the player

decides on a turn or an AI. Both turn selection types have the same result, which is an object of the class "TurnData". It holds all the information needed for the turn execution. The following list describes all attributes of this data structure.

- performer: A reference to the battle behavior script of the performer of the turn.
- receivers: A list of references to battle behavior scripts that belong to all the receivers of the turn.
- name: The displayed name of the turn, when it is executed.
- turnType: A variable of the enum type "TurnType", which includes values for every action a battle participant can perform.
- battleEffect: A reference to an object of the class "BattleEffect". This class defines what kind of effect the turn has on its receivers.
- speed: A integer value that defines the speed of the turn. After the turn gathering phase and before the turn execution phase the turns are sorted according to their speed. The higher the speed value the higher is the possibility of the turn being executed before other turns.
- cost: A integer value that represents the amount of SP needed for execution.

An object of the type "BattleEffect" describes what kind of effect a turn has on the receivers. As described in section 5.4.3, an object of "BattleEffect" is also used to define the effect of a skill or item. After selecting a skill or an item their battle effect object reference is used as a basis to create the actual battle effect object used in the turn data. The reason for this is because the turn's effect differs depending on the performer. The implementation of the battle effect is described by the class diagram in figure 5.5.

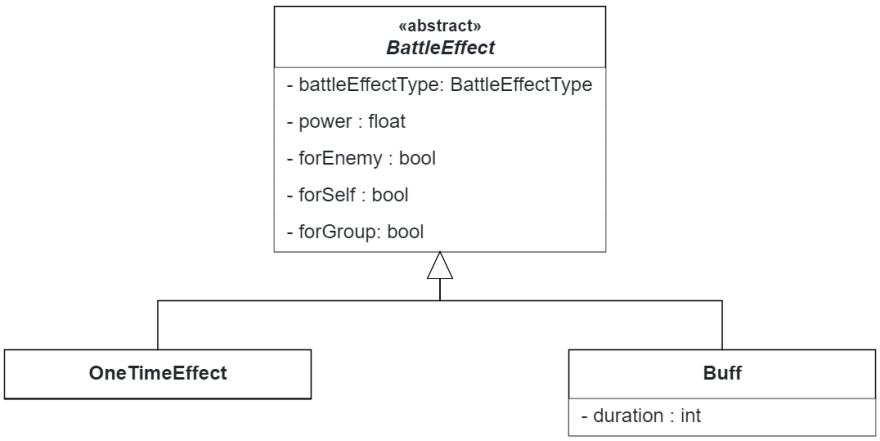


Figure 5.5.: A simplified class diagram of the "BattleEffect".

There exist different effect types, like "Damage", "Healing" and "Attack Up". This is stored in the attribute "battleEffectType" of the battle effect object. There is also an attribute "power", which is a float value that is a measure of how much influence the battle effect has on its receivers. For example, a battle effect that deals damage reduces the HP of the receiver more if the power attribute is higher. Furthermore, there are three boolean values defined for a battle effect. "ForEnemy" defines if an enemy or ally is the target of the move. Some moves require the receiver to be the same as the performer. This is accomplished by setting the boolean attribute "ForSelf" to true. At last, there is the "ForGroup" attribute, which defines if a whole group is the target of the attack. A group is either all the enemies or all the allies.

The class "BattleEffect" is abstract. An object of one of the child classes has to be instantiated. The current version of the battle system features two types of battle effects. The "OneTimeEffect" is as the name implies an effect that only takes place one time directly after the execution. And a "Buff" is an effect that has a certain duration. The duration is the number of turns the buff will be active after execution. For example, a battle effect of the sub-type buff for increasing the attack stat of a battle participant for three rounds can be created by the developer's using the battle system.

5.6. Battle Behaviour

Each battle participant needs a component of the type "BattleBehaviour". As the name suggests, it controls the battle participant within a turn-based battle. Figure 5.6 shows a class diagram displaying the implementation.

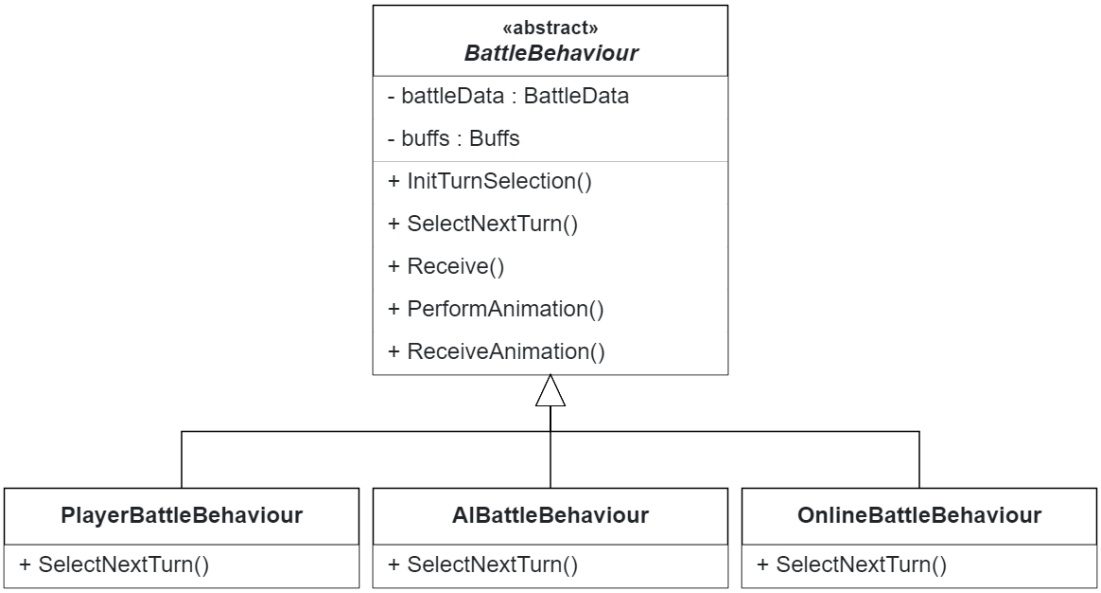


Figure 5.6.: A simplified class diagram of the "BattleBehaviour".

The class "BattleBehaviour" is abstract, which means that no object can be instantiated directly from this base class. An object of the child class has to be used as a component for each battle participant. In the base class, all the functionality for the behavior within turn-based battles is defined, except the selection of a turn. When a class inherits from "BattleBehavior", it must define the function "SelectNextTurn()", which determines how the turn selection is handled. In the current version, two different ways of selection exist. Either the player controls the battle participant or an AI does. In this section, first, the base functionality of the parent class is introduced. Then in section 5.6.2 the sub-type for handling a battle participant by player input is explained. The other sub-type is the handling of an AI-controlled participant, which is covered in section 5.6.3. The sub-type "OnlineBattleBehviour" has not been implemented in the current version of the battle system. But it is a placeholder for future works on the battle system, where the turn selection via a connection to a server could be implemented.

5.6.1. Base Functionality

As explained in section 5.3 turns for each participant are first collected, then sorted, and then executed. How the turn is selected is not part of the base functionality, because it is defined in the child classes. The main task defined in the base class is, handling the reception of a turn, animations, and the buffs of the individual participant.

Receive

As explained in section 5.5 a turn can have multiple receivers. The reception of these moves is handled in the function "Receive()". Each battle effect type has to be handled differently. There is also a difference between handling "OneTimeEffects" and "Buffs".

Animations

All animations associated with turn-based combat are handled within the battle behavior object. The performing of a turn triggers different animations depending on the turn type. The receiving of a turn triggers different animations depending on the battle effect type. When a battle participant is defeated an animation is triggered to communicate this defeat to the player.

Handling Buffs

As explained in section 5.3 the first step in each battle-round is to handle all the currently active buffs. These are stored for each battle participant individually in their BattleBehaviour component. For handling buffs first, all the current lifetimes of the buffs are reduced by one. If a buff afterward reaches a lifetime of zero then the buff will be removed. The second task is to handle buffs related to the HP or SP. Developers using this battle system can create skills or items with a battle effect, that results in healing the battle participant with a certain amount of HP at the beginning of each round.

5.6.2. Player Turn Selection

The player turn selection is implemented in the class "PlayerBattleBehaviour" and defines how the player selects the turn. It is done via the battle menu. For every battle participant that is controlled by the player, the battle menu becomes visible for the turn selection. When the player controls two battle participants, for example, the battle menu selection turn selection will be performed two times.

Battle Menu Selection

The first step in the turn selection is the selection of the move the player wants to perform. Each battle participant has three fixed options, which are explained in 5.4.3. These three moves do not require SP for performing. "Attack" deals moderate damage to a single target. "Defend" provides the performer with a boost in defense for one round. Then there is the option to "run", which, gives the player the chance to escape from the battle by a certain probability. The probability depends on the level difference between allies and enemies. Figure 5.7 shows all the options that can be selected within the battle menu.

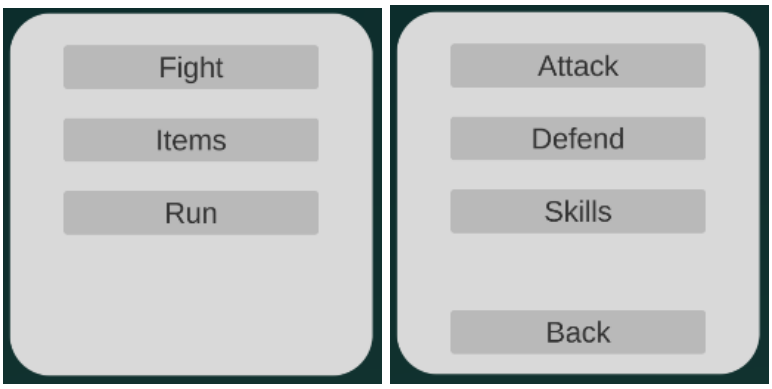


Figure 5.7.: The battle menu for selecting the player's next move.

The left image shows the initial state of the menu. When clicking on the "Fight" button, the menu's state switches to what is displayed in the right image. When clicking on "Items" or "Skills" the menu displays all possible skills and items, respectively. Because the amount of items or skills a player can have is not limited, these menus have been implemented via Unity's "ScrollRect" component. It lets the user scroll down if the buttons do not fit into the menu box all at once.

Target Selection

After the move is selected the next step is the selection of the target. This step will be skipped if there is only one possible target or if all allies or enemies are the targets. The selection works via ray-casting. A ray is shot via touch or mouse input. If a game object is hit by the ray, selection conditions are verified first. It must have a "BattleBehaviour" component equipped and the target group must comply with the "ForEnemy" attribute of the battle effect. For example, when the player selects a turn, which effect is for enemies, only game objects of the enemies can be selected as targets.

Japanese Task Execution

After the target is determined, the last step of the turn selection will be executed. For that, a random Japanese task is initialized and the player has to solve it. After the player finished the task, an evaluation of the task will be displayed. If the player answered correctly the current the selected turn will gain a power increase. More details about this phase of the player turn selection are given in section 5.7.

5.6.3. AI Turn Selection

In contrast to the player turn selection, the AI turn selection does not need interaction with the player for the turn selection. This is because an algorithm determines the next move for the battle participant with an "AIBattleBehaviour" component attached to it. The algorithm can be configured by the developer for each enemy individually, by determining the probability of selecting a certain turn type. In a regular situation within the battle, the AI algorithm always makes a random move to a random target using the configured probabilities. But there exist special situations based on the current status of the battle participant. In the current version of the battle system, two special cases have been included in the algorithm. One of the situations is that the HP of the battle participant is low. In that case, the algorithm prefers a skill that restores their own HP, provided that such a skill is available. The probability of selecting a heal-move in situations like this can also be configured by the developer. The algorithm handles low SP analogous to the low HP case, which is the second special case.

5.7. Japanese Tasks

The turn-based battle system should also feature the learning of the Japanese language. This was accomplished by including Japanese tasks. These tasks are part of the player turn selection and are handled in three steps: task determination, task execution, and task result. Before the implementation is explained, an explanation for the design is given in section 5.7.1.

5.7.1. Design

At the beginning of this bachelor thesis, there were no fixed requirements determined for the Japanese tasks. First, a proper way of placing the tasks within the turn-based battle system has to be determined. After that, the concrete Japanese tasks have to be designed. For finding a way to include the Japanese in the battle system, the related

works were first analyzed to get design guidelines. The related works are covered in chapter 3.

Analysis of other Serious Games for Learning Japanese

The game "Koe" includes Japanese knowledge in the naming of their attack moves. For each Japanese kanji or vocabulary, a separate attack move exists. The player has to select the kanji that is the most proficient against a certain enemy. This has the advantage that the player is not directly confronted with a Japanese task, which increases the feeling of playing a game. Each attack move has its animation that helps to memorize the kanji represented by the attack move. But this has the disadvantage that every time a new kanji is introduced a new animation has to be created. There also exists a disadvantage with the kanji selection approach of Koe. Some kanji may be more proficient against certain enemies, but because there exist no concrete Japanese tasks, there exists no correct solution. Therefore the feedback the player gets after selecting the kanji is limited by the animation.

The game "Learn Japanese To Survive! Kanji Combat" has a concrete Japanese task included in the battle system. An enemy visually represents the concrete kanji and the player selects the explicit correct answer. The proficiency of the attack against the enemy is feedback on the correctness of his selection. High proficiency means the correct answer was selected. The approach used here avoids the feedback disadvantages of "Koe", but there still exist other disadvantages. The first is that in this every enemy is a kanji and this means the introduction of a new kanji leads to the creation of a whole new enemy. Because it is time-consuming to design enemies properly, it would lead to various enemies that behave the same and only have a different visual representation. This results in the player fighting against the "same" enemy over and over again, which could cause the player to get bored. And a bored player is believed to unlikely enter a state of flow, which is the objective of a serious game. Another disadvantage of this battle system is that the feedback the player gets is either "correct" or "incorrect". To help the player to improve himself after failing in a Japanese task, the correct solution should be explained to him afterward.

The game "Slime Forest Adventure" features a battle system, but it is not turn-based. The Japanese task is performed by typing in the correct solution with the keyboard. The game focuses on the reading of kanji or kana characters. To defeat an enemy the correct reading of the character has to be typed in by the player. While he is solving the task, the enemies attack him, which reduces his health. This pressures the player to act quickly. This kind of challenge is not present in the other related works, but for entering

a state of flow, the player has to be challenged in the right amount, as explained in section 2.3.2. Using the principle of time pressure is a design aspect worth considering to include in the battle system developed in the scope of this work. Another advantage is that the player's Japanese skill is tightly coupled with how well the player performs against the enemies in a fight.

The following list is the lesson learned after the analysis of the related serious games.

- The player should get proper feedback. Not only "correct" and "incorrect" but also an explanation.
- Using skills as a Japanese selection task should be used as a secondary task.
- Using time pressure is a way to increase the challenge, which can be used to get the player into a state of flow.
- The player's level of Japanese should influence the battle result

Analysis of other Applications

"Duolingo", "Lingodeer", and "Babbel" are all platforms that help the user to learn a language by using gamification. A variety of different tasks is offered to decrease the chance that the player gets bored. The gamified Japanese tasks can be used in the battle system.
Furthermore, "Wanikani" offers design guidelines. First is the separation of a Japanese task into radical, kanji, and vocabulary. Each of these three types has a different color used in the background, which helps to identify the type of the task. The second design guideline is the usage of a certain color for each task type.

5.7.2. Implementation

The implementation of the Japanese tasks should conform with the design guidelines stated in section 5.7.1. A way to do that is by letting the player perform a certain task each time he selects a turn. The task should not be coupled to the skill that is used or the enemies the player is facing. Furthermore, a variety of Japanese tasks should be provided. This decreases the chance of the player getting bored. To support various Japanese tasks, the software implementation should provide a modular architecture that supports extensibility. In figure 5.8 shows a simplified version of the implementation with a class diagram.

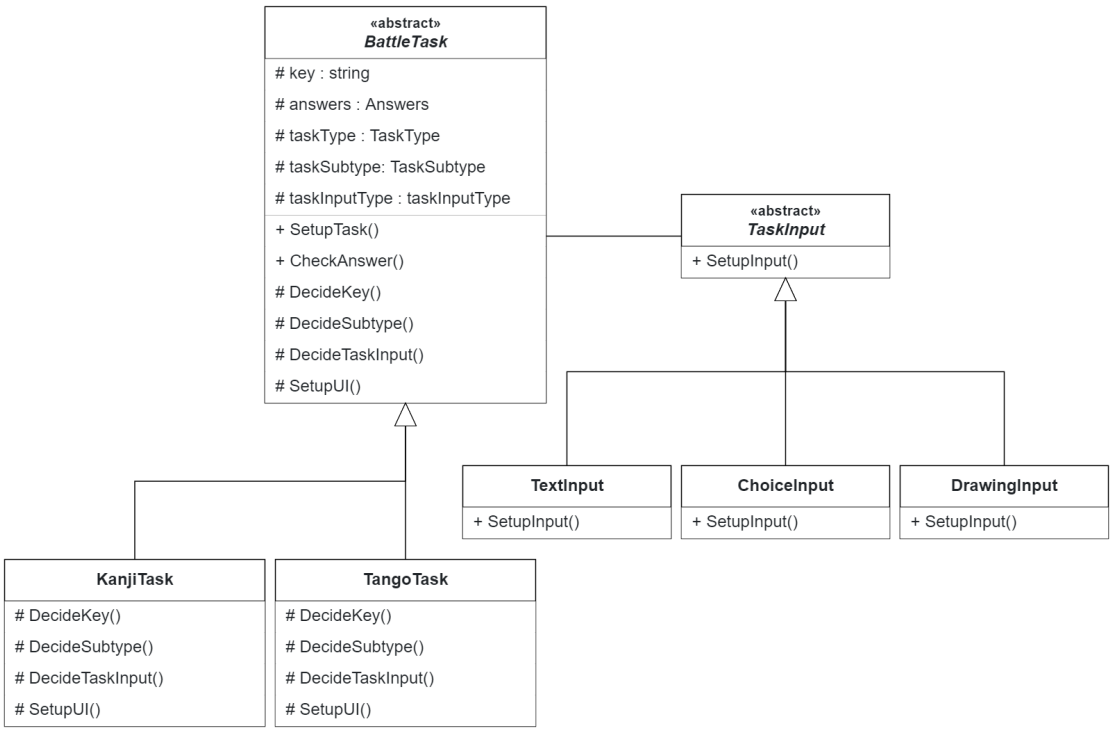


Figure 5.8.: A simplified class diagram of the "BattleTask" and "TaskInput".

Task Determination

The task determination includes four decisions. First, decide on the task type. Currently, there are two task types, "KanjiTask" and "TangoTask". When the task type is decided the object of the type "BattleTask" is instantiated. This class offers the "SetupTask()" method, which decides the other three steps. Because these decisions are based on the first decision, the methods are defined in the corresponding child class. The second decision is the decision of the key. This key represents access to the database where all data of the kanji and vocabularies is stored. Therefore by deciding on a key, a kanji or vocabulary is selected for the current task. The third step is to decide on the task sub-type. For example, there exist the sub-types "Meaning/Translation" and "Reading". An explanation of the sub-types can be found in the appendix.

The last step is to decide on the input method. Currently, there are three different input methods implemented.

- Text Input: Using a text field to type in the answer.

- Choice Input: The player is given four choices but only 1 is correct. He has to select the correct one, to solve the task.
- Drawing Input: Using a drawing panel. A kanji watermark is given, and the player has to draw over it. Drawing is graded by accuracy and the correct number of strokes.

After these four decisions are made, the gathered data is sufficient to initialize the UI. The Setup differs according to the task type and the input method. Each task type child class defines its part in the overwritten "SetupUI" method. Within this method, the "SetupInput()" method of the corresponding "TaskInput" sub-class is called, which defines the input type-specific parts of the setup.

Developers using the battle system can configure the Japanese tasks for each level of "DragonTale". They can define a list of kanji and vocabulary that are used in the level's pool of possible Japanese tasks. Moreover, the probabilities of each task type and sub-type are configurable.

Task Execution

After the task determination step, the UI is ready for the task execution, and the player can perform the Japanese task. In the current version of the battle system, three different UIs exist, and each of these covers one of the three input methods mentioned in section 5.7.2. Figure 5.9 shows the UI setup for the choice input. The battle scene which is displayed in figure 5.2 is blurred out in the background.

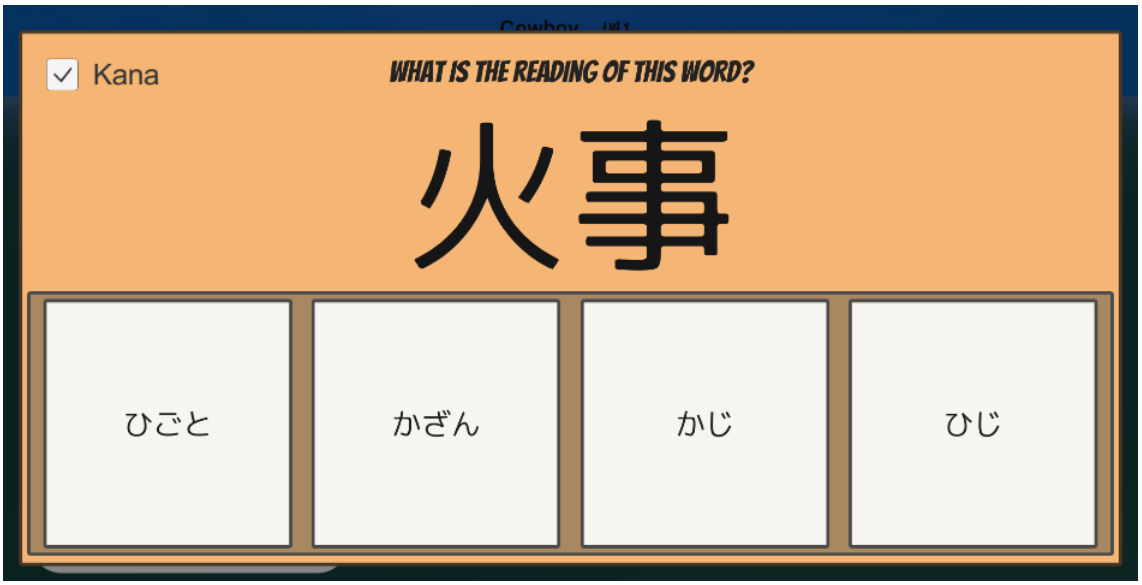


Figure 5.9.: Japanese Task: Player has to select the correct answer.

In the center, the Japanese expression for the task is displayed. If the task type is "KanjiTask", then there is always one kanji displayed. For a "TangoTask" at least one kanji is displayed because a word can exist of one or more kanji. For the task displayed in figure 5.9, there exist two sub-types. One that asks the player for the meaning and one for the reading of the Japanese word. In the figure, the player is asked for the latter. Below the task description, the input UI is displayed. The task of the figure has the input type "ChoiceInput" and when that is the case, four buttons representing the four answer possibilities are displayed. The player has to click the button with the answer he thinks is correct. One issue that has to be addressed for this input method is the selection of incorrect answers. The developers can give the algorithm via a database a pool of wrong answers. These answers and some random answers from the whole database are used as a selection pool for the algorithm. The selection of the wrong answers is then performed by randomly choosing answers from the pool. Furthermore, the algorithm tries to at least choose one of by the developer provided wrong answers. The developer also has the opportunity to provide the algorithm with a pool of correct answers, instead of determining one fixed correct answer.

In the top left, there is a checkbox with the label "Kana", which controls the display mode of Japanese readings. A reading can be displayed in romaji or kana and the checkbox switches between these two modes. The reason this is an option is to give players who haven't mastered the kana writing system yet the opportunity to play the game.

There are Japanese tasks, that do not have the option because the answers are in English. An example is a task, where the player has to select the correct meaning of a kanji.

There exist two other input UI's. One of them is similar to the choice input. The input type "TextInput" has all the UI elements the "ChoiceInput" has, except that the answer buttons are switched out with one text field. In this text field, the user has to type in the correct answer by himself with a keyboard (virtual or physical). This kind of input method has the issue that some unintended typos can cause the player to get the answer wrong, even if the answer he tried to type would have been correct. To tackle this problem, each Japanese task can have multiple correct answers within the database. There the developers can include correct answers with typos.

The third input method is handled differently. This is because the solution of the tasks is not a text, but a drawing of a kanji. This input type only exists for the task type "KanjiTask" and the player has to draw directly over a watermark, that displays the kanji in question. Furthermore, the user has to draw the kanji with the right amount of strokes. Each kanji has a fixed amount of strokes with fixed order and each stroke has a fixed start and endpoint. Japanese learners need to learn these regulations for each kanji because it provides a uniform and structured way of writing Japanese. Figure 5.10 shows the UI of the drawing input.

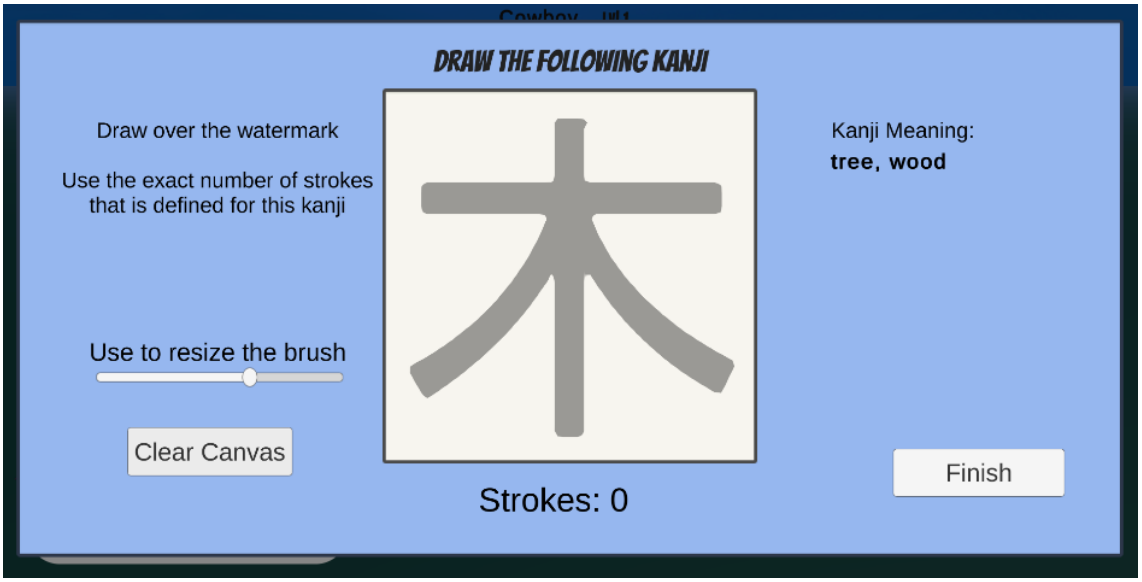


Figure 5.10.: Japanese Task: Player has to draw the kanji for "tree".

The kanji in the middle of the canvas is the watermark. The brush of the user is easily distinguishable from the watermark. After the player finished his drawing he can continue to the task result by clicking "Finish". For the evaluation, the watermark is compared to the player's drawing. Because the watermarks of different kanjis can have differences in the thickness of their lines, no universal brush size can be used. Therefore the slider on the left part of the UI resizes the brush. Furthermore, when the player makes a mistake or wants to test different brush sizes before drawing, he can clear the Canvas with the "Clear Canvas" button. Because a kanji has to be drawn with the exact stroke count defined for it, a counter for them is displayed below the drawing canvas.

There is a disadvantage of using these kinds of Japanese tasks for selecting a turn. One of the goals of "Dragon Tale" is to immerse the player in a fantasy world with a "rich story". The protagonist of the story communicates with the dragon by the "ancient language", which is Japanese. Within the battle system, the naming of skills with Japanese kanji and vocabulary suits this part of the story, but doing a Japanese task after the turn selection does not. This harms the player's immersion within the story world.

Task Result

After the player finishes a task a result window appears and presents the evaluation of the answer. If the player passed the task, the window will be displayed in green colors and otherwise in red colors. This way the failing and passing have a strong visual representation. Additionally, for people with red-green deficiency, a second visual representation is used. At top of the window, either a check (passing) or across (failing) is displayed in black. For every task, two textual feedback are given. First, the correct answer is displayed, which gives the user a way of learning when he answers wrong. The second text is an explanation of why the player passed or failed, respectively. For tasks that have a textual answer the explanation of the evaluation is done by comparing the player's answer to the pool of possible correct answers.
For the drawing input, this is not the case. The player submits a drawing and then it will be compared to the watermark on a pixel basis. If the drawing matches the watermark to a certain percentage (currently 50%), the first part of the evaluation will get a passing grade. The second part is verifying if the correct stroke count was used in the drawing by the player. This kind of evaluation provides feedback on how precise the player can redraw a given kanji. But there still exist issues. First, the stroke order of the kanji does not matter for the evaluation, which is a factor for drawing uniformly and structured. The non-existent evaluation of the stroke's start and end-point is also an issue analog to the stroke order issue.

The second issue with this kind of evaluation is the basic concept of using a watermark. The player does not have to know how the kanji is written from scratch. A task without a watermark is something desirable for the battle system, but the amount of work it would need for implementation was too much for the scope of this work. Other things have been prioritized because noways Japanese can be written on a keyboard. When using a keyboard the reading of the kanji is typed in and then the software gives you possible kanji to select from. So knowing the reading and recognizing the kanji is sufficient in most modern writing applications of Japanese.

5.8. Ending a Battle

After the turn-based combat phase has ended the ending of the battle has to be handled. Three different cases have to be considered: Player wins, Player loses, and the undecided winner case. The battle result is communicated via the dialogue box that can be seen in 5.2. Furthermore, the ending of the battle is the only occasion where the player data gets updated, which is the save-data for the battle system. After a battle ended the battle scene will be left and the player can continue exploring the 3D world. HP and SP will not be restored after a battle, but buffs and debuffs will be deleted.

Player Team Wins

The only case where the player gets rewarded is when the player wins the battle. Each ally receives the same amount of Exp (Experience Points), which is calculated by summing up all the "ExpGainedWhenDefeated" attributes of all the enemies that have been defeated (See section 5.4). If the receiving of the Exp causes a "level up", then this will also be shown as a text within the dialogue box. A "level up" could also lead to the ally learning a new skill, which then will also be displayed in the dialogue. Moreover, each enemy can have loot specified that the player can gain, like explained in section 5.4. So after determining the actual items, the player will receive, this will also be displayed in the dialogue box.

Player Team does not Win

Neither Exp nor items can be gained by the player if the battle is not won. This is true for two cases: the player loses the battle, or neither the player nor the enemy wins the battle. In both cases, after the battle ends, the enemy will not disappear but will stand idle for a certain amount of time. After this time passed, the enemy continues walking around and searching for the player. The idle state is necessary for the case when neither player nor enemy wins because both are intersecting each other in the 3D

world. This is because the battle was initiated by the intersection. With the idle state of the enemy, the player has time to escape by moving away from the enemy. This is less of an issue if the player loses the battle because then the player respawns at the respawn position, which can be arbitrarily defined by the developers using the battle system.
A further issue exists when the player loses the battle. As a result of losing the battle, every ally has no health left and therefore their HPs have to be restored. Fully restoring the HP would harm how a loss is perceived by the player. In a situation where the respawn position is near the location where the player was defeated, it could be a reward instead of a penalty. But the player should also not get stuck in the game because winning is impossible with too little HP. Restoring the health of the allies to only half of the full health is a compromise. This amount could have been reduced if the battle system in its current state had offered a game mechanic for restoring the health of allies without having to fight enemies. In the current version, there are two possible ways to restore HP. Either by using a skill or an item that restores HP.

6. User Study

To evaluate the developed battle system a formal user study has to be conducted. Due to the COVID-19 pandemic, this was not possible. But in section 6.3 a design of a suitable formal user study is proposed. Nevertheless, feedback is still needed and therefore an informal user study was performed, which is covered in 6.2.

6.1. Expected Findings

Because the battle system is a game mechanic that serves the purpose of entertainment a high Game Experience score is desirable. Furthermore, the UI should be usable with a low learning curve. The best scenario would be that the users are intuitively able to use the application. For parts of the battle system that they don't know about, it should be clear to them after they tested the feature. And because this serious game is designed to teach the user Japanese, a for the time the appropriate amount of Japanese knowledge as the learning outcome is also an expected finding of the user study.

6.2. Informal Study

The focus of the informal study is to get some general feedback, gather suggestions for future work, and evaluate the UI's usability.

6.2.1. Setup

Because the user study should only provide feedback on the battle system itself a separate level for this purpose was built. The scene only consists of a floor to walk on and walls to prevent the player from leaving. Furthermore different enemies have been placed in this restricted area. To provide a more diversified experience three different enemy types were created. Multiple instances of these enemy types were placed within the scene. After a user study participant starts the application, he can directly start walking towards an enemy and start a turn-based battle. After the participant finished one fight he can walk towards the next enemy that he is chosen to fight. To provide an accurate experience of the battle system the difficulty must be appropriate. The further

the player moves from the starting location the more difficult the enemies get. This has the advantage of letting participants choose a difficulty suitable for them.

After the participant played between 15-30 minutes, he will fill out the provided questionnaire. There are three different sections within the questionnaire. First, some information about the participant is asked. This includes some general things like age and gender and also should rate the participant's level of expertise with similar applications. The second part is the evaluation of the usability of the UI. For that purpose, the SUS questionnaire (System Usability Scale) was used, which consists of ten questions [US]. The last section consists of questions that should provide general feedback and suggestions. The for the user study provided handout can be found in the appendix.

6.2.2. Results

The informal user study was conducted with a group of eight people, with their ages ranging from 20 to 51 and an average of 31 years. Six participants were 25 years old or younger, and two were 48 years old or older. Seven of the participants identify themself as male and one as female. Most of the participants have experience with mobile applications and video games in general. As for experience with turn-based combat used in other video games, 50% stated that they have little or no experience and the other half stated that they have much experience. The last question asks about the prior Japanese knowledge of the participant. One person stated he had a knowledge level within the range of JLPT N3 - JLPT N5 and seven other participants stated they have little to no prior knowledge.

For the evaluation of the usability of the UI, the SUS score was calculated with the filled-out SUS questionnaires from the participants. The rounded average SUS score is 86 with a standard deviation of 5.8%. The following figure shows how the score is to be interpreted.

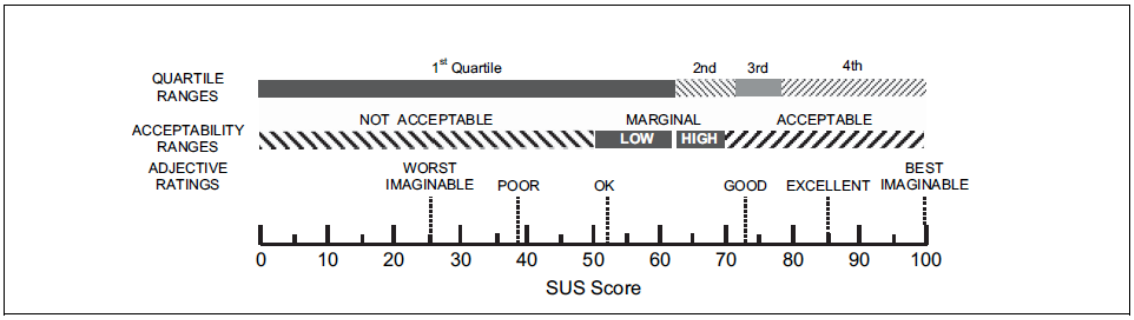


Figure 6.1.: SUS Evaluation [Rau].

The overall score is ladled as "excellent" with a high level of acceptability. This could also be confirmed while observing the participants. Almost no explanations were needed, they all started playing and could intuitively handle the UI. Furthermore, in the "General Feedback and Suggestions" section of the questionnaire, the most-liked point of the battle system is its intuitiveness. Five of the eight participants mentioned this in the question section "What did you like about the battle system?". The following list includes all points the participants liked about the battle system.

- Learning effect. While playing the game the speed of answering a Japanese task and the rate of correctness increase.
- The battle system "feels complete", which means the participant that wrote this, does not miss any essential functions he desires for the battle system.
- For "harder" enemies strategic thinking is necessary.
- The way Japanese knowledge affects the outcome of a battle.
- The combination of learning and playing in general
- The integration of the Japanese tasks is "seamless"

In the questionnaire, there was a question that would ask the participants about what they did not like and how they would improve it. Three of the participants found the lack of audio within the battle system something to be improved. Two of these three want to have audio in general and one participant also suggested that the readings should be voiced and played within the Japanese task. Also suggested by three participants is that they would raise the difficulty. They found the test application not difficult enough and therefore said the motivation for correctly answering a Japanese task was too low. Other participants also stated that the penalty for answering a

Japanese task incorrectly should be higher. This shows the importance of configuring the battle system correctly. With the current version of the battle system, the developers can increase the difficulty by tweaking some parameters. The problem is the level of skill the player has to be taken into account for the difficult configuration. A possible solution is to give the player a chance to decide on the difficulty level. Furthermore, something similar was mentioned by another participant, who wanted to have the option to decide on how difficult a Japanese task can be. In chapter 7 more possible improvements are explained.

6.3. Description of a Formal User Study Setup

Only an informal user study with 8 people was conducted. The results show that the turn-based battle system was well received by the participants. Even so, a reliable statement about the usefulness of the battle system can not be made with this user study. Only acquaintances and family have partaken in it and therefore a chance that the results have been influenced by that is not negligible low. The formal user study should be conducted in public with unrelated people. But because of the current circumstances with the COVID-19 pandemic, this was not possible. Even conducting the user study with people using video chat is problematic. The participant needs to have a suitable device at hand and also must be able to install and start the application.

To make a statement about the usefulness of the battle system, participants have to be divided into two groups. Because the battle system is a part of the game "Dragon Tale", its usefulness should be measured by comparing the results of a group that played the game with the battle system and a group that played it without the battle system. Furthermore, it would be beneficial if each participant has a similar level of Japanese knowledge and the kanji and words learned within the user study should be according to that level of knowledge. To find out the learning outcome, the participants need to fill out a questionnaire with the Japanese knowledge questions before and after playing the game. These two questionnaires should differ in some questions to avoid the issue that participant focus on questions they heard before while playing the game. The difference in knowledge can be used to find out the learning outcome. Then the last step is comparing the average learning outcome between the two participant groups. Should the average of the group that played the game with the battle system be considerably higher, then the battle system could be declared useful.

To gain some additional insights about the user experience or rather the game experience the PET (Player Experience Inventory) can be used. It rates how the player

experiences a game and a score can be calculated for each filled-out questionnaire. Comparing the score to other games, via a benchmark data set, can help to rank the game [Cam].

7. Future Work

The developed battle system already provides all essential features for performing turn-based combat. Developers of "Dragon Tale" can use the battle system in its current state and would be able to offer various battles within the game. But there is still room for improvements and additional features, which will be discussed in this chapter. The different suggestions are also influenced by the conducted informal user study.

7.1. Improvement of Existing Features

The first improvement is the utilization of the skill names. Each skill should have a Japanese name and the effect playing while performing the move should help the player to memorize the kanji or word. Then for the Japanese task, the time needed for completing is not relevant in the current version. In chapter 5 section 5.7 there was mentioned that giving the player limited time can motivate the player by raising the difficulty of the challenge. So the first step would be to define a time limit for each task. Furthermore, extra time can be defined, which classifies the task's completion as fast. A "fast" task completion could be rewarded with an extra amount of power increase of the selected turn.

Taking the UI into consideration, there is also still room for improvements. First of all the assets used in the current version should be replaced by ones more befitting the Dragon Tale style. Then there is no information on the skills, items, and different actions the player can take within a turn-based battle. A possible solution would be by implementing some kind of tool-tip, which also works with touch screens. For example, after pressing the button for two seconds the tool-tip will show as a description of the skill that was selected. The same can be done for the buffs. Each buff has an image that represents it in the status UI. For recalling what the buff does in detail a tool-tip can be useful. Furthermore, an essential part of the UI is missing, the audio feedback. Each effect should have a different sound effect and in the background, motivational music can be played. Moreover, each button click should have either auditory or haptic feedback. Haptic feedback can also be used for the attack effects. For example when the player gets hit by the enemy the player's device vibrates, which signalizes the loss of Hp.

7.2. Possible Additional Features

In the current version, the player only gets confronted with Japanese kanji or words. When displaying a reading the player can choose between romaji and kana. But learning the kana characters has not yet been a part of the battle system. With the current software architecture, it is easy to include more different types of Japanese tasks. Therefore when the need for learning the Japanese kana characters arises it could easily be included in the battle system. A similar addition is the inclusion of a different Japanese kanji drawing task. Currently, the player only needs to redraw the given kanji but does not need to remember the exact writing. By implementing an algorithm that detects handwritten kanji another drawing task can be included, where the player has to write a kanji without the help of a watermark. A good starting point would be to read Kamate's article "Creating a Japanese Handwriting Recognizer" [Kam]. He describes a way to implement a handwriting recognizer using different libraries in the programming language Python.

Each player has a different level of Japanese. This has not been taking into account yet. Each level has its defined set of Japanese tasks. A better solution is by keeping a record of the player's Japanese language progress. For example, with such a profile a list of kanji and words that the player has not been reviewed for an extended time can be found out. A review list tailored for the player can be included in the battle system task determination phase. In addition after starting a new game, the player can be given the option to select a difficulty. There could be a difficulty setting for the Japanese language difficulty and the turn-based combat difficulty.

There are also possible additional features, which increase the complexity of the turn-based combat mechanic. In the current version, it is possible to create a variety of skills and items for enemies and allies. Both use a reference to a "BattleEffect" object to define their effect. The class could also offer an attribute "type" and each battle participant also gets a type. Then a battle effect with a certain type could be more or less effective against a target of a certain type. This kind of game mechanic is frequently used in other battle systems. Well-known examples are games from the "Pokemon" franchise.

There are possible additional features that relate to the battle system but are not directly involved in the turn-based combat. The player can gather items, by defeating enemies. In the current version, these items can only be viewed and used within a battle. A useful addition would be to include a menu outside of a battle where the player can view and use items. Furthermore, the player should be able to create items outside of a battle. This prevents the player from getting stuck in a level because he is not able

to defeat the enemies with the current health of his allies. A possible implementation that also includes the use of Japanese knowledge is brewing potions by applying Japanese grammar. A puzzle-like game for the brewing process is an option that can be considered in future work on Dragon Tale. An example puzzle could make use of the "particles" in Japanese grammar. The player needs to select the correct particles for each part of a Japanese sentence.

The last additional feature introduced in this chapter is the implementation of social gaming elements. For example by including leader boards, which gives the players extra motivation to improve their Japanese. Furthermore, an Online-Multiplayer could be a useful addition. By letting players compete against each other or compete together in difficult Multiplayer-Events is also an extra motivation for learning Japanese. The groundwork for the Online-Multiplayer is already set with the structure of the "BattleBehaviour" classes. As described in chapter 5 section 5.6 the class OnlineBattleBehaviour needs to be implemented. Moreover, to make it work, the turn data for each battle participant regardless of what kind of BattleBehviour component is has attached, needs to send its turn's information to a server for the data exchange.

8. Conclusion

In this thesis, a turn-based battle system for a Serious Game for learning Japanese was developed. In the beginning, a design had to be specified for the battle system. By analyzing related work, design guidelines were gathered, which helped in the design process of the battle system. Afterward, the implementation was started by first sketching a rough class diagram of the whole application. Then the actual implementation of the battle system with the game engine Unity3D began. Various features had to be designed and implemented, from the battle initiation to the handling of the battle result. During this phase, multiple meetings with the development team of "Dragon Tale" were held, which concluded in feedback on the current state of the battle system. After the development was finished, an informal user study showed that the battle system was well received by the participants. In this bachelor thesis also a statement about the effectiveness of a turn-based battle system for learning Japanese was tried to be made. Due to current circumstances, it was not possible to conduct a formal user study, which would be necessary for making such a statement. But the positive feedback gathered from the informal user study gives reason to believe that it is effective. In future work on the battle system, the formal user study that was proposed in this thesis should be conducted to ascertain this statement.

At the end of the bachelor thesis, the battle system was integrated into the base game. The integration could be done without issues. Furthermore, the development team of "Dragon Tale" was informed on how to use the battle system. While the battle system in its current state is already usable and gives the developer numerous possibilities to create content, there is still room for improvement. This thesis presented various improvements to the current features and additional features. For future work on the battle system, these are a good starting point.

A. Overview Class Diagram

The following class diagram gives a simplified overview of the whole application. In chapter 5 parts of this class diagram are explained in more detail.

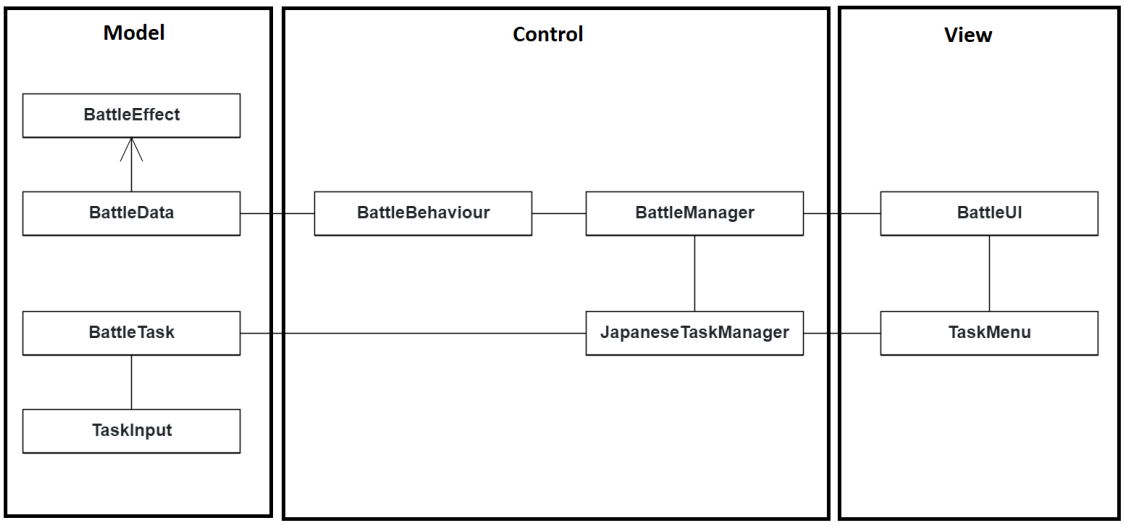


Figure A.1.: Overview of the whole application.

B. Enums

The following table represents the enum "BattleEffectType", which defines the types of effects a turn in the battle system can have.

damage	Damages the target. Reduces the target's HP.
healing	Heals the target. Recovers the target's HP
recoverSp	Recover's the target's SP.
eraseSp	Erases a part of the target's SP.
attackBuff	Attack Buff: Increases the target's Attack stat for a certain number of rounds.
attackDebuff	Attack Debuff: Decreases the target's Attack stat for a certain number of rounds.
defenseBuff	Defense Buff: Increases the target's Defense stat for a certain number of rounds.
defenseDebuff	Defense Debuff: Decreases the target's Defense stat for a certain number of rounds.
speedBuff	Speed Buff: Increases the target's Speed stat for a certain number of rounds.
speedDebuff	Speed Debuff: Decreases the target's Speed stat for a certain number of rounds.
escape	The battle participant tries to escape. Succeeds with a certain probability.
nothing	Nothing will happen.

Table B.1.: Enum "BattleEffectType"

The next table shows the possible task sub-types for a Japanese task, which are all defined in the enum "TaskSubtype".

meaning	Player has to give the meaning of the kanji or the translation of the vocabulary.
reading	Player has to give a reading of the kanji/vocabulary.
readingOn	Player has to give an onyomi-reading of the kanji.
readingKun	Player has to give a kunyomi-reading of the kanji.
writing	Player has to give a writing of the kanji (Using drawing input).

Table B.2.: Enum "TaskSubtype"

C. User Study Handout

The following pages represent the handout that each user study participant received. But one part of the questionnaire that was in the original document was removed for the appendix version. The evaluation of the usability of the battle system. Because these ten questions are exactly the ones from the SUS questionnaire, there is no need for including them in the appendix.

Dear participant,

Thank you for participating in the evaluation of the turn-based battle system for learning Japanese. You do not need any prior knowledge about Japanese. You'll be provided with all the information about the Japanese language you need for the test. The objective of the test is to evaluate the user interface of the battle system and gather feedback and suggestions.

The test includes the following steps:

1. Read the info sheet given to you (starting Page 2), which includes all the Japanese language knowledge you'll need. You can use this information while you play the game.
2. Install the game on a suitable Android device.
3. Play the game. A walkthrough can be found on page 4.
4. Fill out the questionnaire. (Starting on page 5)
5. Send back the filled-out questionnaire or this whole document.

Infosheet

The Japanese tasks you need to perform are random and are divided into two categories. Japanese Kanji and Japanese Words. Japanese Kanji are a part of the Japanese writing system and are used for writing Japanese words. For each of the categories there exist different sub-types.

For Japanese Kanji

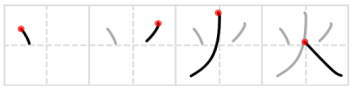
- Meaning: You need to give the meaning of the kanji.
- Reading: A kanji can have multiple different readings. There exist two different types of readings. Kunyomi (Japanese origin), Onyomi (Chinese origin). You need to give either a kunyomi-reading, an onyomi-reading, or one of both.
- Drawing: You need to redraw a given kanji with the right amount of strokes. Each kanji has a fixed stroke count, with each stroke is defined by a start and endpoint.

For Japanese words

- Translation: You need to give a suitable English translation for the word
- Reading: You need to provide one way to read/pronounce the word

For the test the following kanji and words are used. (The information was copied from https://jisho.org)

Kanji: Fire

火 Stroke order 

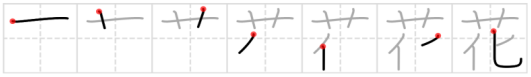
Meaning: Fire

Kunyomi-Reading: hi, bi, ho

Onyomi-Reading: ka

2

Kanji: Flower

花 Stroke order 

Meaning: Flower

Kunyomi-Reading: hana

Onyomi-Reading: ka, ke

Kanji: Mountain

山 Stroke order 

Meaning: mountain

Kunyomi-Reading: yama

Onyomi-Reading: san, sen

Word: Firework

はな び
花火

Translation: fireworks

Reading: hanabi

Word: Volcano

か ざん
火山

Translation: volcano

Reading: kazan

3

Walkthrough

1. Start the game. You'll find yourself in a world with enemies. The dragon Ryu that follows you is not an enemy, he will fight for you.
2. Walk towards an enemy with the joystick on the bottom-left screen. Fights are initiated by coming into contact with an enemy. An enemy will also chase you when he has spotted you.
3. After you made contact with an enemy, the fight will begin. You'll be given instructions and information by the dialogue box in the bottom-middle of the screen.
4. You'll be given different options to choose from in the menu in the bottom-right. Select what you want to do. Afterward select the target, which can be an ally (Ryu) or an enemy. That depends on the move you selected. The last step is the Japanese task. The information given for each task should be sufficient for you to deal with the task. Use the kana checkbox on the top left, if you are not able to read hiragana characters. If you do not know the answer you can use the info sheet. Answering correctly will at least double the move's power.
5. Continue with the battle, till either Ryu (you) or the enemy is defeated.
6. Fight other enemies. Play the game for 15-30 minutes. Try different things out.
7. Close the game and continue with the questionnaire on the next page.

Questionnaire

Please fill out the following questions after playing the game.

Personal Data

1. Age: ____
2. Gender: ____
3. Experience with mobile applications?
   - ☐ No experience
   - ☐ A little experience
   - ☐ Decent experience
   - ☐ Much experience
4. Experience with video games?
   - ☐ No experience
   - ☐ A little experience
   - ☐ Decent experience
   - ☐ Much experience
5. Experience with turn-based combat in video games?
   - ☐ No experience
   - ☐ A little experience
   - ☐ Decent experience
   - ☐ Much experience
6. Prior Japanese knowledge?
   - ☐ No prior knowledge
   - ☐ I know a few things
   - ☐ JLPT N5 - JLPT N4
   - ☐ JLPT N1 - JLPT N3

5

General Feedback and Suggestions

1. What did you like about the battle system?

2. What did you not like about the battle system? How would you improve it?

3. Do you have suggestions for additional features for the battle system?

8

List of Figures

List of Tables

Bibliography

[AAPa] K. Ahlström, M. Ahlström, and A. Plummer. *Kanji: Fire*. URL: https://jisho.org/search/%E7%81%AB%20%23kanji (visited on 09/02/2021).

[AAPb] K. Ahlström, M. Ahlström, and A. Plummer. *Word: Furigana*. URL: https://jisho.org/word/%E6%8C%AF%E3%82%8A%E4%BB%AE%E5%90%8D (visited on 09/02/2021).

[AAPc] K. Ahlström, M. Ahlström, and A. Plummer. *Word: Okurigana*. URL: https://jisho.org/word/%E9%80%81%E3%82%8A%E4%BB%AE%E5%90%8D (visited on 09/02/2021).

[APK20] D. A.Plecher, A. Pongratz, and G. Klinker. "Effects of Augmented Reality in a serious Gmes for Learning Japanese Kanji." In: (2020). DOI: doi:10.1145/1122445.1122456.

[APO] *Apollon*. URL: https://apollon.ase.in.tum.de/ (visited on 09/10/2021).

[Art] ArtMap Inc. *Top 10 Most-Sold Consoles of All Time Ranked*. URL: https://hackernoon.com/top-10-most-sold-consoles-of-all-time-ranked-e22t34xs (visited on 08/16/2021).

[Bab] Babbel GmbH. *Babbel website*. URL: https://www.babbel.com (visited on 07/27/2021).

[Bib] Bibliographisches Institut GmbH. *Defintion of Gamification*. URL: https://www.duden.de/rechtschreibung/Gamification (visited on 05/27/2021).

[BOS11] E. Banno, Y. Ohno, and Y. Sakane. *Genki: An Integrated Course in Elementary Japanese 1*. 2nd. The Japan Times, 2011. ISBN: 9784789014403.

[Cam] Campus Group T. *Player Experience Inventory: User Guide*. URL: https://playerexperienceinventory.org/docs (visited on 08/24/2021).

[CN02] M. Csikszentmihalyi and J. Nakamura. "The Concept of Flow." In: Jan. 2002. ISBN: 9780195135336. DOI: 10.1007/978-94-017-9088-8_16.

[Con13] A. S. Conning. *The Kodansha Kanji Learner's course*. 2nd. Kodansha USA, 2013. ISBN: 978-1-56836-526-8.

[Dör+16] R. Dörner, S. Göbel, W. Effelsberg, and J. Wiemeyer. *Serious Games: Foundations, Concepts and Practice*. Jan. 2016. ISBN: 978-3-319-40611-4. DOI: 10.1007/978-3-319-40612-1.

[Duoa] Duolingo Inc. *2020 Duolingo Language Report: Global Overview*. URL: https://blog.duolingo.com/global-language-report-2020/ (visited on 08/16/2021).

[Duob] Duolingo Inc. *Duolingo website*. URL: https://www.duolingo.com (visited on 07/27/2021).

[Gar85] R. Gardner. "Social psychology and language learning: The role of attitudes and motivation." In: (1985). DOI: https://doi.org/10.1017/S0272263100007634.

[Hau] P. Hauer. *Das Facade Design Pattern*. URL: https://www.philipphauer.de/study/se/design-pattern/facade.php (visited on 09/03/2021).

[Hea11] R. C. Heather Maxwell Chandler. *Fundamentals of Game Development*. Jones and Bartlett Learing, 2011. ISBN: 9784789014403.

[Hen14] B. Hennessey. *extrinsic and intrinsic motivation*. 2014. URL: https://www.academia.edu/36033031/Hennessey?bulkDownload=thisPaper-topRelated-sameAuthor-citingThis-citedByThis-secondOrderCitations&from=cover_page (visited on 08/31/2021).

[Joh] D. Johnson. *Slime Forest Adventure website*. URL: https://lrnj.com (visited on 07/27/2021).

[Kam] A. Kamate. *Creating a Japanese Handwriting Recognizer*. URL: https://towardsdatascience.com/creating-a-japanese-handwriting-recognizer-70be12732889 (visited on 08/26/2021).

[Kim15] T. Kim. *A Guide to Japanese Grammar*. 2nd. 2015. ISBN: 9781495238963.

[Kim17] L. Kimppa. "Rapid formation and activation of lexical memory traces in human neocortex." In: (2017).

[Mam] M. Mamerow. *Gaming Industry vs. Other Entertainment Industries (2021)*. URL: https://raiseyourskillz.com/gaming-industry-vs-other-entertainment-industries-2021/ (visited on 08/16/2021).

[MVC] *Model View Controller*. URL: http://wiki.c2.com/?ModelViewController (visited on 09/03/2021).

[Nin] Nintendo Co., Ltd. *Nintendo History*. URL: https://www.nintendo.co.uk/Corporate/Nintendo-History/Nintendo-History-625945.html (visited on 08/16/2021).

[OV11] N. Oroujilou and M. Vahedi. "Motivation, attitude, and language learning." In: (2011). ISSN: 1877-0428.

[Özt20] I. Öztürk. "Impact of Augmented Reality in a Serious Game to Learn Japanese Kanji." Technische Universität München, 2020.

[PIL] L. Pillar. *Top 10 Richest Video Game Companies*. URL: https://airentertainment.biz/top-10-richest-video-game-companies/ (visited on 08/16/2021).

[Rau] M. Rauer. *Quantitative Usablility-Analysen mit der System Usability Scale (SUS)*. URL: https://blog.seibert-media.net/blog/2011/04/11/usablility-analysen-system-usability-scale-sus/ (visited on 08/24/2021).

[SK12] V. Saxena and S. Kumar. "Impact of Coupling and Cohesion in Object-Oriented Technology." In: (2012).

[Smi71] A. Smith. "The importance of attitude in foreign language learning." In: (1971). DOI: https://doi.org/10.1111/j.1540-4781.1971.tb00916.x.

[Son] Sony Group Corporation. *Sony History*. URL: https://store.steampowered.com/app/672430/Koe/ (visited on 08/16/2021).

[The] The Japan Foundation. *Change in number of examinees (1984-2018)*. URL: https://www.jlpt.jp/e/statistics/index.html (visited on 08/16/2021).

[Tof] Tofugu LLC. *Wanikani website*. URL: https://www.wanikani.com/ (visited on 07/27/2021).

[Toy] Toyota Motor Corporation. *Toyota Overwiew*. URL: https://www.sony.com/en/SonyInfo/CorporateInfo/History/ (visited on 08/16/2021).

[US] U.S. General Services Administration (GSA) Technology Transformation Service. *System Usability Scale (SUS)*. URL: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html (visited on 08/21/2021).

[Vala] Valve Corporation. *Steam website - KOE*. URL: https://store.steampowered.com/app/672430/Koe/ (visited on 07/27/2021).

[Valb] Valve Corporation. *Steam website - Learn Japanese To Survive! Kanji Combat*. URL: https://store.steampowered.com/app/759440/Learn_Japanese_To_Survive_Kanji_Combat/ (visited on 07/27/2021).

[Yam+20] K. Yamaguchi, M. Iwata, A. Vargo, and K. Kise. "Mobile Vocabulometer: A Context-Based Learning Mobile Application to Enhance English Vocabulary Acquisition." In: New York, NY, USA: Association for Computing Machinery, 2020, pp. 156–159. ISBN: 9781450380768. DOI: 10.1145/3410530.3414406.

[Zhu] W. Zhulong. *Lingodeer website*. URL: `https://www.lingodeer.com` (visited on 07/27/2021).