Feed It
Game Design Document

Lilly A. Helmersen	helmerse@kth.se
David Magnusson	dmagnuss@kth.se
Hugo Bergqvist	hugobe@kth.se
Björn Winge	bjorna3@kth.se
Kristoffer Almroth	kristoffer.almroth@gmail.com

Computer Game Design

Stockholm, December 18, 2020
KTH Royal Institute of Technology

Contents

Design History

In the current stage we have only achieved one major release. As of now we have created a first draft demo of the first zone to show as a proof of concept and implementation of concept art (see figure 1, 2).

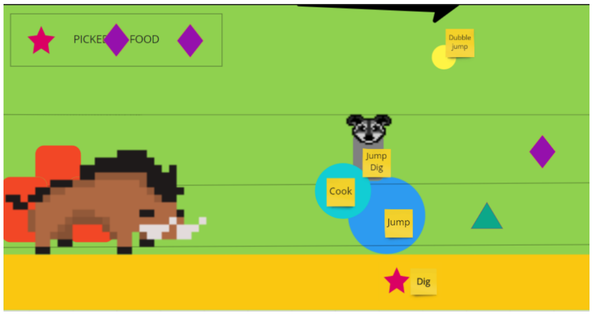


Figure 1: First draft concept art

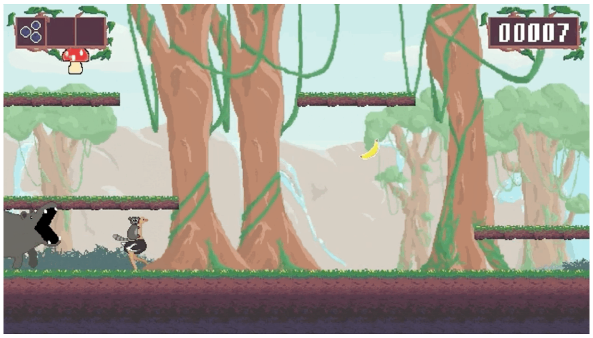


Figure 2: In game screenshot

The version we have now is a first demo which is planned to go through a user testing to let people play the game. This way we will get first hand feedback in order to balance the game, adjust existing features and adding new ones before the release to Steam. Conducting user tests will tell us if the foundation is solid and if there is anything crucial that needs to be changed, added or removed.

1. Section I - Game Overview

1.1 Game Concept

Feed It is an endless runner. The scene consists of two main characters, a raccoon and an ostrich, cooperatively collecting ingredients as they appear in front of them on different levels in order to cook dishes to feed an angry beast chasing them. Depending of what ingredients you collect, different dishes gets cooked. The different dishes affect the beast in different ways. Some make the beast faster, thus reducing the distance between it and the players, and dishes that contain mushrooms tires the beast which increases the distance between them. There is no guide to which combination of ingredients cooks which dish. Therefore the players have to work together in order to conclude what ingredients they want to collect and which they want to avoid in order to cook a certain dish. This results in a "learn by doing" aspect to the game and potential for unexpected events to occur.

The ingredients and platforms that appear in front of the players to interact with are procedurally generated from prefabs and chunks(see section 2.1), resulting in unexpected encounters and situations so the player cannot learn a pattern and are faced with different situations every time. As the game layout is ever changing it increases the aspect of "learning by doing" when cooking which motivates the player to keep playing and testing different approaches and learning new recipes. This in turn prevents the game becoming plain and predictable.

The game consists of different zones with different locations combined with a specific beast. Each zone has a specific start difficulty which increases for every zone. The longer you stay in one zone the more difficult it gets. The concrete strategy for the game is to stay in one zone as long as you can manage and when it gets too difficult you try to feed the beast enough for it to exit the screen before it reach you. Entering a new zone *resets* the difficulty to a set level and the objective is to manage as high of a difficulty as possible in one zone before you reset and enter the next zone. This way you will maximise the amount of score gathered in each zone and reach as high of a score as possible (see Figure 2.2. The figure will be more thoroughly explained in 2.3).

1.1.1 Motivation

The motivation and our vision with this game is to bring back couch gaming and by the help of retro features create a nostalgic atmosphere. A lot of the games that are created today require unique consoles or devices. Modern games rely on internet connection and digital tools for communication and aim towards the online multiplayer platform. For Feed It we want to bring back physical multiplayer and cooperation, therefore our game is played on one shared keyboard, having both players physically present. When developing the game we decided to go with pixel art since it could also be an art style associated with retro gaming or how it used to look *back in the days.* Combining how the game is played with this specific art style made it possible for us to create a truly retro game and bring back the good old days!

1.2 Secondary game features

1.2.1 Cookbook

The gameplay and mechanics are quite simple as described under **??**. This gives us the opportunity to research and implement other features and aspects of the game. One aspect that is planned for the finished game is to have a cookbook in the menu for the dishes you have made. The cookbook will by default have all the dishes available but *blacked-out*, making them mysteriously unidentifiable. As you cook dishes this book will update and the dishes you have explored will be visible. This could create incentive for the player to chase other game objectives than just high scores, adding secondary aspects which can increase the lifespan of the game.

1.2.2 Easter eggs and Achievements

We are also planning to incorporate several easter eggs and achievements for the player. The way we have made the game gives us the potential for several easter eggs. We see a huge potential for sound easter eggs and are planning to implement several. Some of them are ”favourite dishes” for different character which triggers a voice message and random conversations between the raccoon and ostrich (perhaps related to trending topics). Visual easter eggs could be to add seasonal and trending dishes as well as incorporating objects in the parallax backdrop.

We are also looking into and planning to add several Steam achievements. For example when you have cooked rare och legendary dishes or when you have hit certain number of cooked dishes, 100/1000/10000 etc.

1.3 Feature Set

- Beast chasing players.
- Couch gaming experience. Two people playing on a shared keyboard.
- Procedurally generated platforms.
- Gather ingredients to cook and feed the beast in order to get away.
- Automatic pick up and cooking of ingredients
- Unspecified cooking recipes.
- Two playable characters.

1.4 Genre

The genre of the game is ”high pace indie co-op endless runner”.

1.5 Target Audience

The target audience is friends that enjoy playing collaborative games of an indie fashion, appreciate endless runners and grinding for exploring the entire game and improving ones skills to master the mechanics.

Even if the game is playable as a single player, the experience and development of the game has been considered for two players sharing a computer keyboard together. The expected time for a single play session is set between 30 minutes to an hour. A big motivation for this game is to bring back couch gaming and the collaborative and shareable experience of using the same device, which requires the players to be in the same room to play together.

The age restriction for this game has been set to *pegi*[1] 7.

1.6 Game Flow Summary

The players automatically runs in the game leaving the them with the option to jump up and down from platforms collecting ingredients as they appear at different levels. The players have to make sure they reach the level on which the wanted ingredient spawns to be able to collect it. In order to cook the dish, the players have to make sure that they are joint and attached to each other. The players collide with the ingredients they want to collect and use in their dish while avoiding the rest. However, the players have to consider if they want to collect the ingredients on the ground level, because if they are left they are in reach of the hungry beast who, by eating the ingredients, gets a speed boost.

1.7 Look and Feel

The game is in 2D with pixel art. It is a side-scrolling game where the characters runs parallel to the camera and the player can see a backdrop of the environment. The general feel and mood in the game is high pace. The surroundings consists of an environment combined with a beast. This combination repeats itself consistently over each zone, resulting in different visuals but with a consistent feel throughout the entire game.

1.8 Project Scope

For the scope of this project we have decided to only provide and create a demo of the game. The demo consists of one zone where the beast is a hippo and you are being chased in a jungle. However, the full game would include 8 zones with 8 different beast and environment combinations.

1.8.1 Number of NPC's

Each zone has one NPC, namely the beast chasing the player. The NPC gets replaced as the player completes each zone. Therefore, there only exists one NPC at a time but the complete number of NPCs is 8.

[1] https://pegi.info/

2. Section II - Gameplay and Mechanics

2.1 Gameplay

2.1.1 Game Progression

Our ambition for the first release of the full game is for it to consist of 8 different zones in the following order:

Jungle	Cave	Dessert	City	Sky	Water	Ice Cave	Volcano

Since the game in an endless runner, completing the 8th and final zone will return you to the first zone and then repeat itself. The difficulty still progressively increases even when the zones repeats itself. This repetition of zones when completing the final one correlates with our vision of having the characters running around the world and eventually you revisit the first location where the first beast has gotten hungry again. Our ambition is to continue to create more zones in the future making it harder and less repeating.

There are three main goals in the game: complete all zones, get as much score as possible and unlock all recipes in the cookbook. When first playing the game the goal will be to defeat each beast to survive for as long as possible and get through all zones. As stated in 1.1 Game Concept, the difficulty will increase in the later zones, making it more challenging to survive and outrun the beast. The terrain generation and the ingredients will change for each area. Since the game is randomly generated using predefined chunks of 50 tiles that are spawned in a random order, the player will after a while start learning the level generation of each zone and know what types of generations are more likely to include mushrooms. That knowledge will persist between runs, making it easier to reach further on each attempt.

After the player has successfully finished the last zone and beaten the beast there, the completionist's objective will be to fill the cookbook with recipes from all zones. The demo does not include a cookbook but it would be one of the first things for us to add in the future. Testing out recipes will add a new challenge to the game since the player need to cook dishes that will decrease the distance between the beast and the players, adding more challenge to the game. Once all recipes are filled in and the cookbook is completed, the main objective left is to gain as high score as possible. With the knowledge of all recipes and their corresponding score value, the game will become a challenging balance between risk and reward for feeding the beast with valuable dishes compared to feeding it with mushrooms to slow it down.

2.1.2 Puzzle Structure

As mentioned above, when we have an established foundation of recipes we will implement a cookbook. This main puzzle mechanic is to figure out all different recipes in all eight zones. There are some logic to the recipes, for example collecting three of the same ingredient usually makes something special. In all cases the recipes are logical in the sense that the final dish is something you can recognise as a real food item that contains those ingredients. Part of the charm here is that you do not know before you have tested something, and the levels are randomised so you will get different possibilities each run. The player needs to figure out the best recipe to cook based on the ingredients currently available.

2.1.3 Objectives

The objectives of the game differs between what type of player is playing it. For a more casual indie game player the goal will be to beat all zones. For the completionist the goal will be to fill in the cookbook with all recipes from all zones. Lastly, for the more competitive players the goal will be to get as high score as possible and compete with their friends and the rest of the world.

2.1.4 Play Flow

The game flow can be described as a cycle that repeats for each new zone the player encounters. At first, everything is new, new ingredients, unknown level generation and a new beast that hunts the player. How do you survive in this zone? One of the first things the player needs to realise is how you get the tranquilliser to make the beast sleepy. In some zones you find mushrooms in the upper part of the level, in others the raccoon needs to dig down in the ground to find the tranquilliser. Once the player has figured this step out, the goal becomes to learn the level generation well enough to be able to keep the beast at bay, and start to figure out the unique recipes for this zone.

After the player has beaten a zone a couple of times, the last step of the cycle is to optimise the zone for the specific play style of the player. If the goal is to beat all zones, then it will be optimised for completion as fast and risk free as possible. If the goal is to fill all recipes then the zone will be filled with logical reasoning and trial and error. For the third play style of high score, the optimisation will include balancing the risk reward as high as possible.

These steps will repeat for each new zone a player encounters until all zones are in the last step of the cycle, in which the player has achieved their goal of the game.

2.2 Mechanics

In this section we will explore the main mechanics of the game.

2.2.1 Physics

The physics in the game works similar to real life physics with some heavy modification to improve the feel of the game. The ostrich, raccoon and the dishes all have a weight and are affected by gravity. The gravity constant used in the game is $g = -9.81$. This constant is later amplified by the objects using it, where the ostrich and the raccoon scales their gravity by a factor of 2 and the dishes scales it by a factor of 3. When it comes to jumping the game deviates from physics which will be discussed in the next section.

2.2.2 Movement

The game is an endless runner where the terrain moves in relation to the player with the speed of 10 tiles per second. Ingredients and dishes move with the terrain. The ostrich and the raccoon interact with the world using two primary methods; they can jump up in the air and they can jump down through semi solid platforms. Their x-position in relation to the camera is always static. To make the jump feel good it is created by an initial upward force, then the gravity is increased by a factor of 2.5 if the jump button is no longer being held. The gravity also increases with the same magnitude after the peak of the jump. Low gravity at the start of the jump and high gravity at the end of the jump makes it easier for the player to perform a precise jump while also keeping the jump from feeling floaty.

In addition to playing around with the gravity, there is also a jump buffer of a tenth of a second and Coyote time of the same size. This gives the player a small window where they can input a jump just before landing on the ground or input a jump just after leaving the ground. Without these buffers the game feels more unresponsive and laggy.

Another mechanic for jumping occurs when the raccoon is sitting on the back of the ostrich, then you are able to perform a double jump by first jumping with the ostrich then jumping with the raccoon. This mechanic is needed to reach ingredients high up in the sky and to reach certain higher platforms. This mechanic will be more explored in section 2.2.4.1 Mounted.

The beast can only move in the opposite axis of the ostrich and the raccoon, which means horizontally. When it eats ingredients that it can reach on the ground it gains energy and moves towards the right, reducing the distance to the players. If it is fed a dish that does not include a tranquilliser it also moves to the right, with a higher velocity than a simple ingredient. To get it to move to the left it needs to be fed a dish including a tranquilliser. If the beast goes outside of the screen it falls asleep and stops hunting the ostrich and the raccoon.

2.2.3 Objects

The beast and the two playable characters, the ostrich and the raccoon has already been described in the previous sections. This section will focus on ingredients and dishes.

2.2.3.1 Ingredients

Ingredients are part of the game world and moves with the same speed as the terrain. They can either be floating above the ground or higher up in the air, and in some cases be below the ground. In general they are spawned randomly, where one position in a certain chunk can be a strawberry, an egg or nothing for example. Their spawn chance it dependent on the position. In general, the most common ingredients are spawned close to the ground level, at a height where the beast can eat them. Examples of common ingredients are a strawberry and an egg. More rare ingredients are spawned in places where they are harder to reach, such as along platforms high up on the screen, or below the ground. Tranquillisers will mostly spawn in these locations, as well as some ingredients that are part of recipes that are very valuable.

2.2.3.2 Dishes

The dishes are created when the raccoon cooks three ingredients. When the cooking is done, the dish is thrown forward in an arc and when it lands on the ground friction will stop it so it moves with the terrain. Different dishes are created depending on the recipe used. If only common ingredients were used then the outcome will be ”bowl of goodies”, which will only give 30 points towards the score counter. Successfully following a recipe will create a unique dish, that will reward the player with a much higher amount of points. All dishes cooked using a mushroom will create a type of ”bowl of tranquilliser” which also gives 30 points. See 2.2.2 movement for the effect of the dishes on the beast. Since the dishes follows regular physics it is possible to throw one on a platform too high up for the beast to reach, that way the player can avoid the effect of the dish on the beast.

2.2.4 Actions

We have already talked about one action, jumping. in this section cooking, digging and eating will be described in more detail.

2.2.4.1 Mounted

The raccoon starts the game by being mounted to the ostrich. By either pressing the jump or the down key, the raccoon can dismount from the ostrich and run around on its own. To mount again the raccoon either has to jump down onto the back of the ostrich or the ostrich has to jump up and pick up the raccoon. While mounted the raccoon follows the ostrich in its movement, making it possible to do a double jump if the raccoon jumps off the ostrich's back while it is in the air. This is used to reach higher ledges and collect ingredients high up in the sky. If played using two players, then coordination will be an added challenge when trying to double jump of start cooking as fast as possible.

2.2.4.2 Cooking

Cooking is an integral part of the game. It is the main way you interact with the beast. The player needs to collect ingredients both for following their intended recipe but also to hinder the beast from eating the ingredient and get closer. You can hold a total of three ingredients at a time, and the only way to get rid of a collected ingredient is to cook it into a dish. Running around with three ingredients is mainly a waste of time. To be able to cook the raccoon needs to sit on the back of the ostrich, which starts a process that takes three seconds. During that time the raccoon cannot move away from the back of the ostrich. After the cooking is done a finished dish will will be thrown forward and you will be awarded with a score bonus. See figure 2.1 for the recipes seen in the demo.

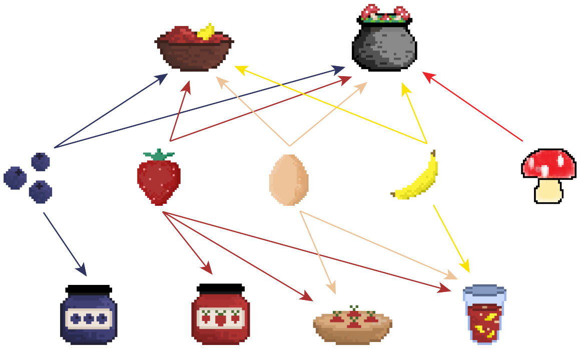


Figure 2.1: The recipes available in the demo.

2.2.4.3 Digging

In the first zone, jungle, the main objective for the raccoon is to traverse the upper part of the level to collect more rare ingredients and tranquillisers. To create a more dynamic movement and more objectives for the raccoon some of the zones will have ingredients, mainly tranquillisers and other more rare ones, under ground that the raccoon needs to dig down to get to. The raccoon will have to balance between sitting on the ostrich, searching for ingredients on high platforms and searching for ingredients under ground. Digging is not implemented in the current state of the game but it is a prioritised feature to add.

2.2.4.4 Ostrich Eating

The ostrich will spend most of its time on the ground and on the lower platforms, where the beast can reach the ingredients that are left behind. Since you can only hold three ingredients and some that spawn may be unwanted to collect, it is common to skip them and let the beast eat them. This becomes a problem when you have three ingredients and are either cooking or moving the raccoon closer to the ostrich to begin cooking. Ostrich eating is an action that gives the ostrich an option to just straight up eat one ingredient. That way you can eat low floating ingredients that you either do not want to collect or cannot collect, preventing the beast from closing in on you. This mechanic is very powerful and needs to be balanced well. The initial cool down of this action is 15 seconds, which will be iterated on using user testing. Currently, the ostrich can only eat ingredients and not dishes.

2.2.5 Intrigue

The main conflict of the game is between the beast and the two playable characters. The ostrich and the raccoon want to run away and the beast wants to eat them. If the beast is getting close to the ostrich and the raccoon then it will get more angry, signalling that the player need to do something to push the beast back. If the best is getting close to going outside of the screen then it will get sleepy, signalling that the player is getting close to run away and can take more risks. As mentioned in previous sections, the distance between the beast and the ostrich and the raccoon is changed by the beast either ingredients and dishes. If the beast gets fully outside of the screen then the game will transition to the next zone and a new beast will appear.

2.2.6 Score

For those that wants to master the game and those that just likes to collect points there is a score system. The score system is dependent on how far you have ran and what dishes has been cooked. One point is awarded for each ten tiles or one second of play time. This will be the least effective way to gain score. A more effective way is to memorise which combination of ingredients cooks into the most valuable dishes and then focus on them. The score system will create a risk/reward balance, where you are only getting a small amount of points for actions that will increase the distance between you and the beast, and get a larger amount of points if cooking special dishes that makes the beast close in on you. The score does not scale during a zone, but it does scale between zones. Cooking a rare dish in a later zone would yield more points than cooking a rare dish in one of the earlier zones.

2.3 Difficulty

The difficulty of the game is dependent on several factors. The main difficulty comes from the beast closing in on the player. More ingredients close to the ground will make the beast faster, as well as ingredients that gives more of a boost to the beast. Later zones will have ingredients that both gives a greater boost and are more common to be spawned close to the ground, depending on the type of zone. The availability of tranquillisers is also important, because without them you can not increase the distance between yourself and the beast. Both the number of tranquillisers and their placement is important. Earlier zones will have a lot of them in easy to reach areas, and later zones will make them more sparse and harder to reach.

Another factor to consider is the quota between the normal ingredients and the tranquillisers. If you are trying to increase the distance between yourself and the best, then you want to avoid normal ingredients until you have at least one tranquilliser, else you risk cooking something that will get you closer to your death. Avoiding certain ingredients will become harder the more non-tranquilliser ingredients there are on the screen at a given time.

2.3.1 Difficulty level

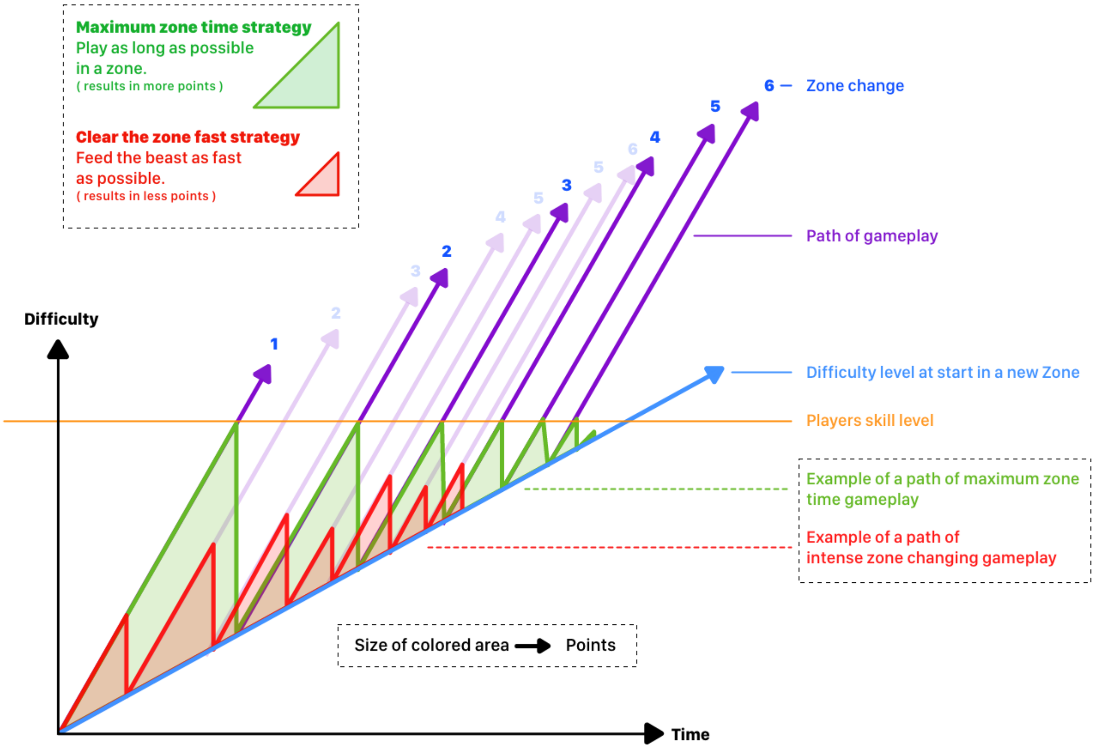

Maximum zone time strategy
Play as long as possible
in a zone.
(results in more points)
Clear the zone fast strategy
Feed the beast as fast
as possible.
(results in less points)
6 — Zone change
1
2
3
4
5
Path of gameplay
Difficulty
Difficulty level at start in a new Zone
Players skill level
Example of a path of maximum zone time gameplay
Example of a path of intense zone changing gameplay
Size of colored area → Points
Time

Figure 2.2: A chart describing the balance of difficulty levels in the game.

Above is a graph describing gameplay in relation to difficulty. The x-axis is time spent in a game run. The y-axis is the increasing difficulty level. The light blue line describes the minimum level of difficulty at the start of a zone depending on time. The level of difficulty can never go below this line (y is always equal to or greater than the light blue line). The purple lines represent the different zones. A game sessions progression of difficulty must always follow the purple line corresponding to the current level (y is always equal to or greater than the purple line). So when a player manages to change zone, the difficulty moves straight down the y-axis until it reaches the light blue line, and the next level's purple line begins.

Two types of gameplay are illustrated in the graph. The green path represents a strategy where the player stays as long as possible in a zone. The yellow line in this case represents the player's skill level. This is the difficulty level on the y-axis where the player is no longer capable of feeding the beast frequently enough to push him out of the screen and thereby change the zone and temporarily decrease the difficulty to a manageable level.

The area underneath the path could be seen as an amount of point due to the relationship between time and points.

The red path is an example of gameplay where the player feeds the beast as fast as possible and thereby changing zones with a higher frequency than the green gameplay. Even though the skill level will result in similar time spent in a run, the amount of points will be higher for the green path.

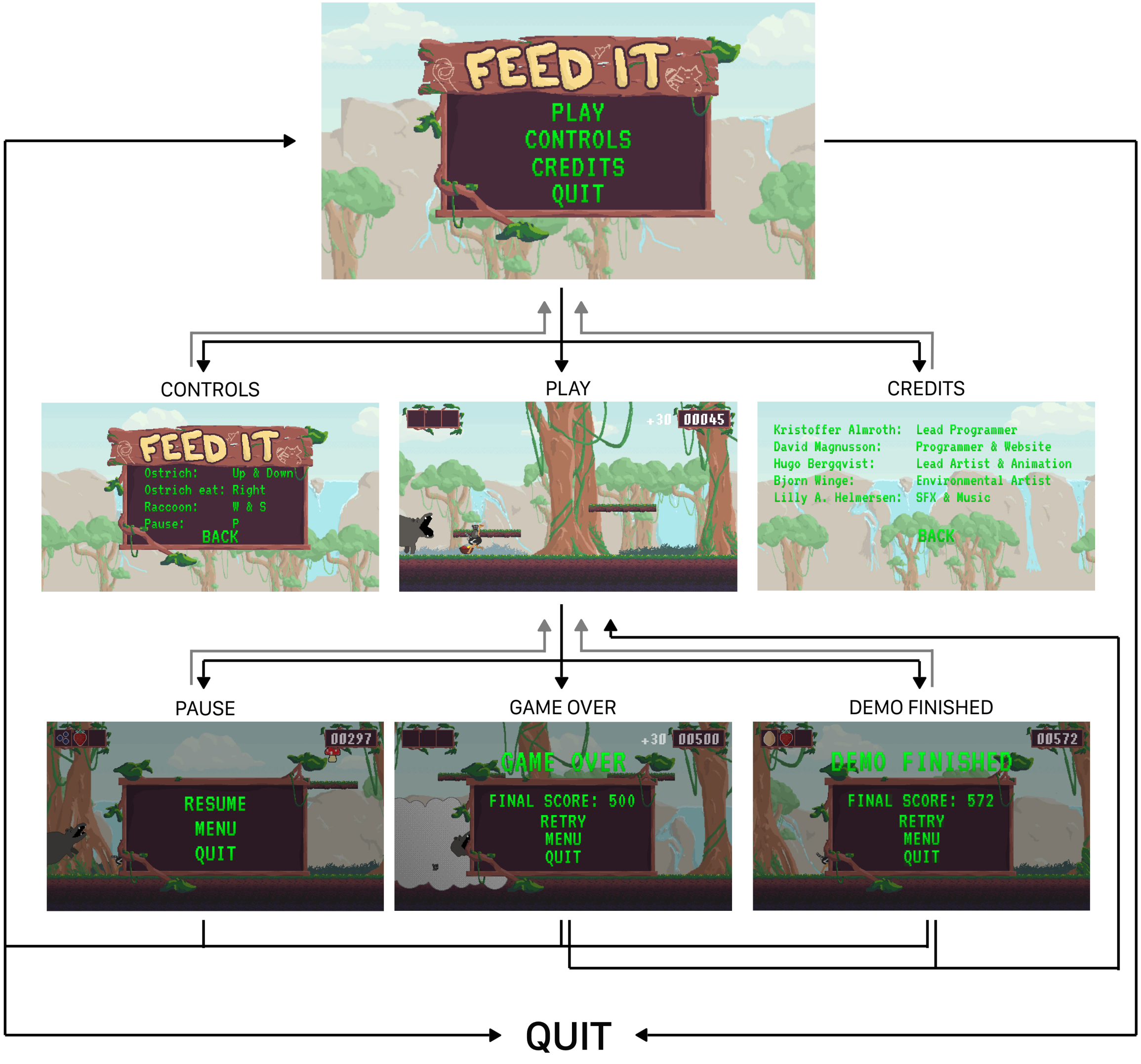
This graph could also be used in the decision of setting the difficulty of the game. The gradient of the light blue and the purple lines is based on the previous mentioned factors resulting in difficulty in the game. So by tweaking the gradient of the different lines, e.g. reduce or increase the food necessary to feed the beast off the screen, or the time it takes for the food to appear, it is possible to increase or decrease the gradient an approximate time of gameplay based on the players skill level.

The increase in difficulty is now also linear and the gradient of the two different paths are constant. A possible solution for further game play development could be to experiment with increasing gradients and exponential curves.

2.4 Screen Flow

In this section we will explore the screens in the game.

2.4.1 Screen Flow Chart



Ostrich:	Up & Down
Ostrich eat:	Right
Raccoon:	W & S
Pause:	P

Kristoffer Almroth:	Lead Programmer
David Magnusson:	Programmer & Website
Hugo Bergqvist:	Lead Artist & Animation
Bjorn Winge:	Environmental Artist
Lilly A. Helmersen:	SFX & Music

Figure 2.3: Flow chart describing the path between all screens in the demo.

2.4.2 Screen Descriptions

In the following section we will go through all screens in the game.

2.4.2.1 Main Menu Screen

The first screen you see when playing the game is the main menu. There you have the options to start the game, go to options menu, see the credits and quit the game. See figure 2.4.

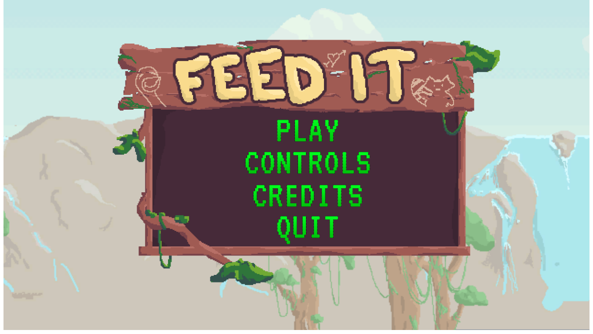
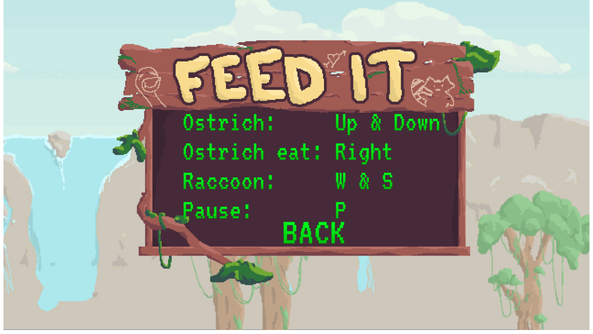


Figure 2.4: Main menu form the demo.

2.4.2.2 Options Screen

Options menu for all needed options. This will likely include rebinding of controls as well as settings for volume and graphics. See figure 2.5 for the controls screen in the demo.

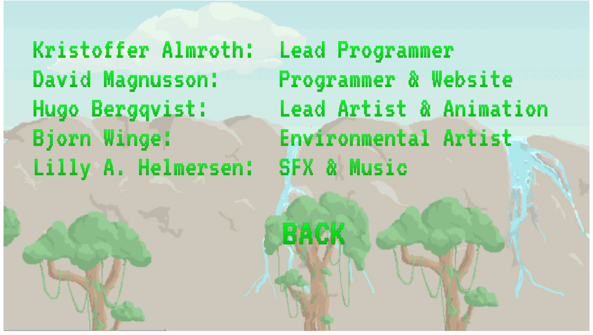


```
Ostrich:     Up & Down
Ostrich eat: Right
Raccoon:     W & S
Pause:       P
```

Figure 2.5: Controls menu from the demo.

2.4.2.3 Credits Screen

Credit screen displaying everyone who worked on the game and their role. These roles may change in the future, which should be reflected here. See figure 2.6 for the credit screen in the demo.



Kristoffer Almroth: Lead Programmer
David Magnusson: Programmer & Website
Hugo Bergqvist: Lead Artist & Animation
Bjorn Winge: Environmental Artist
Lilly A. Helmersen: SFX & Music

Figure 2.6: Credits screen from the demo.

2.4.2.4 Game Screen

The game screen, where the player is expected to spend most of their time. We have the ingredient collection HUD in the upper left of the screen and the score counter in the upper right. See figure 2.7 for the game screen in the demo.

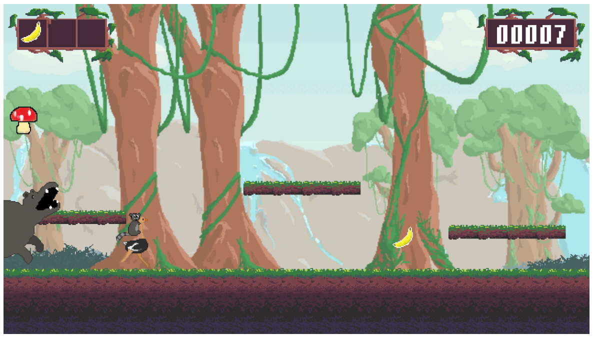


Figure 2.7: Game screen from the demo.

2.4.2.5 Pause Screen

Pausing the game is important. It lets you take a quick break if you need to do something in the middle of a run, and gives you the option to go back to main menu to restart the game or quit the game entirely. See figure 2.8 for the pause menu in the demo.

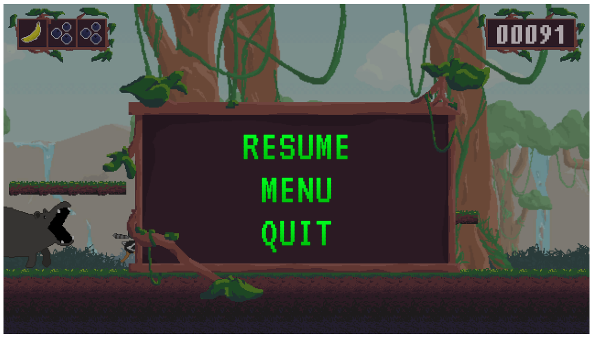


Figure 2.8: Pause menu form the demo.

2.4.2.6 Game Over Screen

After dying the player expects to be able to see how well they performed in the form of their score, as well as being able to restart the game. There is also an option to go back to main menu and to quit the game. See figure 2.9 for the game over screen in the demo.

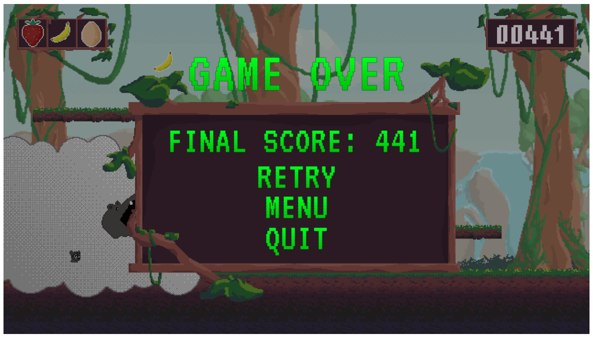


Figure 2.9: Game over screen from the demo.

2.4.2.7 Demo Over Screen

Demo over screen is unique to the demo and will not be in the full release of the game. It marks the point of the game where the demo ends and the user needs to buy the full game to go further. It works similar to the game over screen. See figure 2.10 for the demo screen.

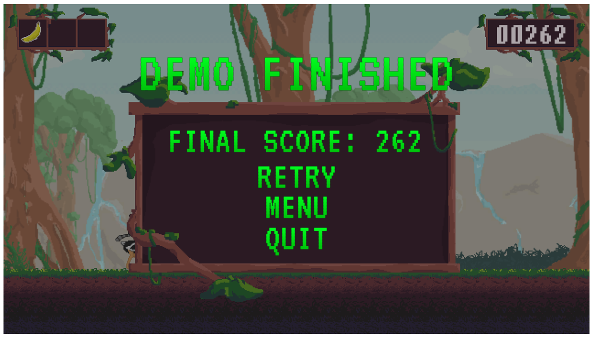


Figure 2.10: Demo over screen from the demo.

2.5 Game Options

Graphics settings will include the following:

- Resolutions
  - 640x360
  - 1280x720
  - 1920x1080
  - 2560x1440
- Window
  - Full screen

    - Borderless
    - Border
- VSync
    - On
    - Off

For volume settings there will be the following audio sliders, where each goes from 0 (no audio) to 10 (max audio).

- Master Volume
- Music Volume
- Sound Volume
- Ambience Volume

In the section for controls the user will be able to rebind all inputs in any way they desire.

2.6 Replaying and Saving

One of the core parts of the game is replay-ability. The user is not meant to beat the game on the first try or even the tenth try. In section 2.1.4 Play Flow we discuss the different steps the player goes through each time they enter a new zone. That cycle will repeat until the player manages to complete the game for the first time, by which the player instead will focus on the other objectives of the game that can be seen in section 2.1.3 Objectives.

Since the player starts over from the start each time they play the game, there is not much that needs to be saved. In total there are three aspects of the game that needs to be saved down.

- Settings
- Cook Book Progression
- High Score

3. Section III - Story, Setting and Character

3.1 Story and Narrative

The story begins in the jungle in an unspecified time era. The players are instantly thrown into the game as the two main characters, the raccoon and the ostrich, where they have to run and cook to save their lives! They are being chased by an angry beast who wants to eat them, and he gets angrier and faster by the minute. Not only do they have to keep running, they also have to try to collect and feed the beast mushrooms with a drowsing effect to make him slow down enough to outrun him. But as we all know, the jungle is full of delicious and nutritious fruits and berries, so the beast gets an energy boost along the way by eating everything it can reach, except the dangerous mushrooms of course! Two heads are better than one, so the raccoon and the ostrich have found out that if they cook the fruits and berries together with the mushrooms, they are able to trick the beast into eating the cooked dish. But will they be able to feed the beast fast enough before he catches up with them?

3.1.1 Plot Elements

The plot element of the story is that collaboration is necessary for survival. By helping each other to collect and cook the items necessary for slowing down the beast, the players will be able to outrun the beast.

3.1.2 Game Progression

The game progresses through different zones. Once the player outruns a beast, a seamless transition occurs into a new environment. A new environment also introduces a new threat, a new angry beast to outrun. There are 8 different zones in the game, all representing different parts of the games world. So passing through all of the zones could be seen as running a lap around the world.

3.2 Game World

3.2.1 General look and feel of world

The game world has a cartoonish feel to it. A lot of rich colours and somewhat over exaggerated characteristics of trees and waterfalls etc.

3.3 Characters

3.3.1 Character #1 - The Raccoon

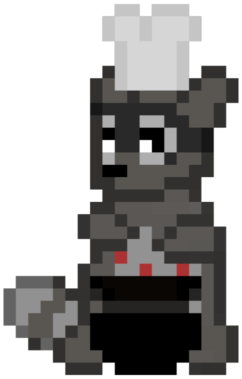


Figure 3.1: The Raccoon.

3.3.1.1 Back story

As a child the raccoon lived happily with his mother and father in a big hollow jungle tree. The raccoon's parents, just as their parents, and their parents' parents' neighbours, ran a restaurant at the bottom floor of a tree serving only striped food. One day when his parents were out stealing supplies for the next day's cooking, he heard a roar so loud that it shook all the bananas out of their big jungle tree. He never saw them again after that. To survive, he kept the family business going, and is today considered the fourth best cook in the whole ten tree radius from his restaurant.

3.3.1.2 Personality

The raccoon is a survivalist. He is clever, an average cook, and not afraid to go after the things he needs. He likes to add cool comments to his actions.

3.3.1.3 Physical characteristics

The raccoon is small and cute with big eyes. He has a high pitch voice.

3.3.1.4 Animations

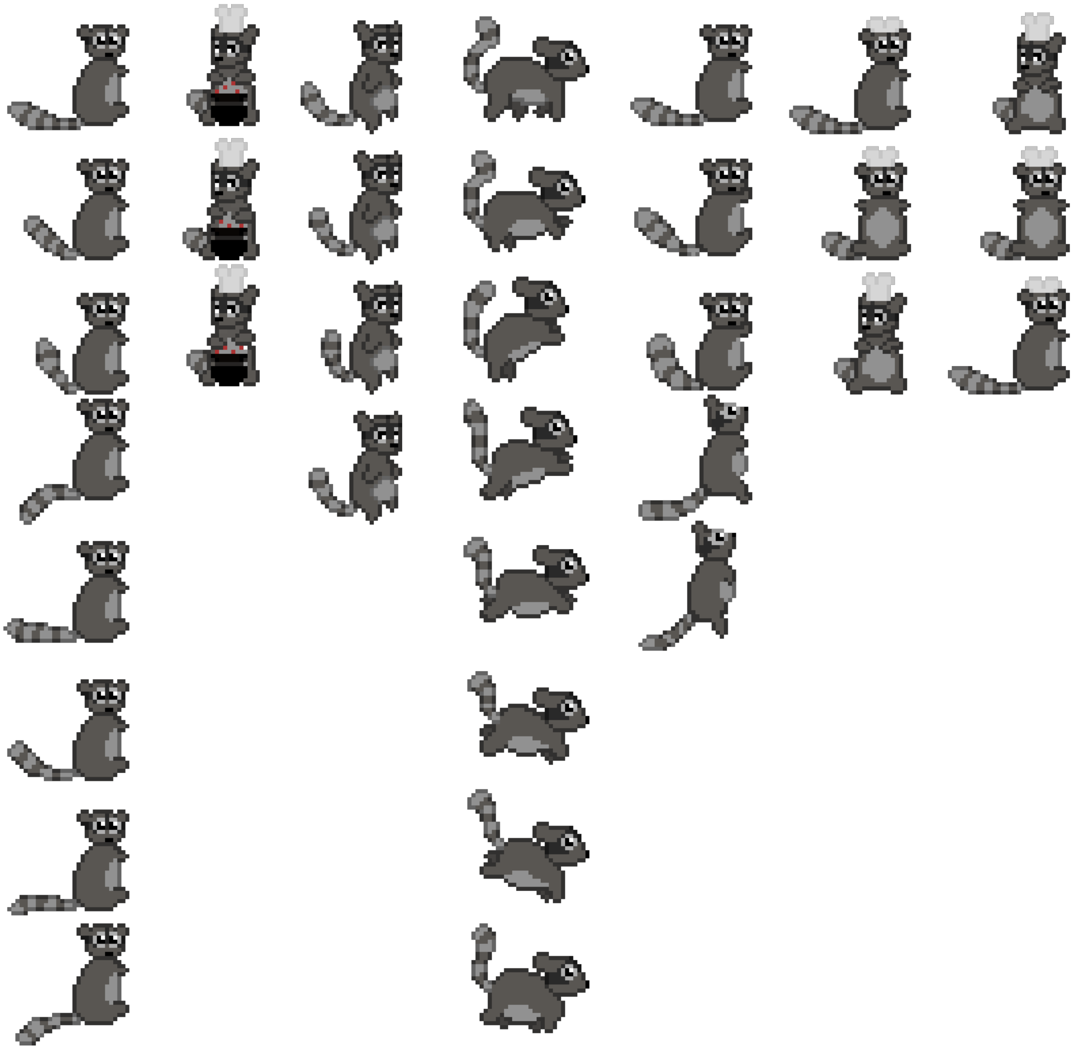


Figure 3.2: Raccoon animations

3.3.1.5 Special Abilities

- The ability to cook meals
- Reach places an ostrich can not

3.3.1.6 Relevance to game story

He is one of the two main characters. Without him, you would not be able to cook meals containing mushrooms which is necessary to beat the beast. The raccoon is also needed to reach the mushrooms on the higher platforms.

3.3.1.7 Relationship to other characters

Best friend with the ostrich.

3.3.2 Character #2 - The Ostrich

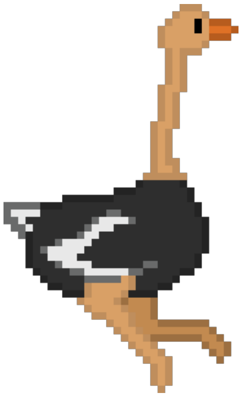


Figure 3.3: The Ostrich

3.3.2.1 Back story

The ostrich is not the wisest bird in the flock, but he always has a positive attitude. He can not talk, but instead communicates with his eyes. But due to the size of his eyes being around 2 pixels each, and is lacking a blinking animation, he can't really communicate at all. When he was young he lost his family while chasing his own shadow into the jungle. One day while standing still doing nothing, as he usually did, he felt something on his back. He turned his head and saw a raccoon jump off his back into a tall tree. The raccoon noticed the ostrich head move and apologised for jumping on him and explained that he thought he was a small three. They both laugh, one out load and one on the inside, and have been best friends ever since.

3.3.2.2 Personality

The ostrich is a so called follower. He trusts the raccoon, and everyone else, with his life.

3.3.2.3 Look

3.3.2.4 Physical characteristics

The ostrich is tall and big enough for the raccoon to fit on his back.

3.3.2.5 Animations

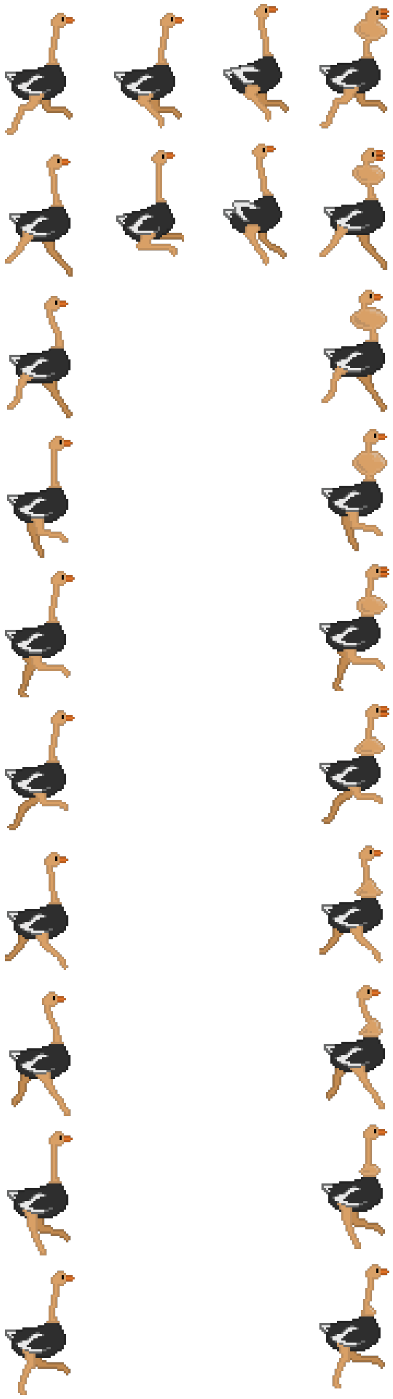


Figure 3.4: Ostrich animations

3.3.2.6 Special Abilities

- Able to run without getting tired
- Can boost the raccoon to the highest platforms
- Able to eat food so the beast does not get it

3.3.2.7 Relevance to the game story

He is one of the two main characters. The raccoon needs him to always keep running to avoid getting eaten by the beasts. The ostrich also boosts the raccoon while jumping to reach the mushrooms placed at the hard to reach places.

3.3.2.8 Relationship to other characters

Best friend with the raccoon.

3.3.3 Character #3 - The Hippo

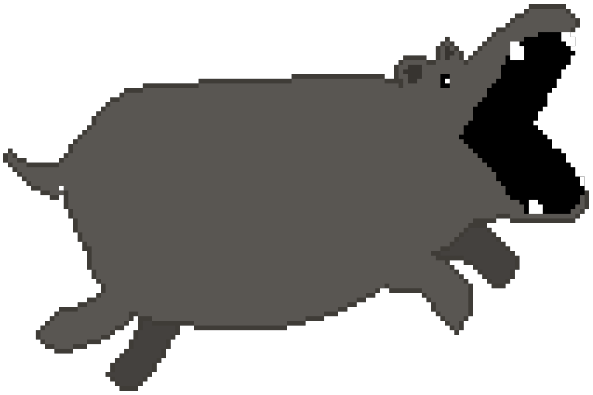


Figure 3.5: The Hippo

3.3.3.1 Back story

The hippo is taking a nap in a soft bush full of berries. He suddenly wakes up because someone is pulling his ear.

3.3.3.2 Personality

Very angry, and very hungry.

3.3.3.3 Look

3.3.3.4 Physical characteristics

The hippo is big and has a huge gape.

3.3.3.5 Animations

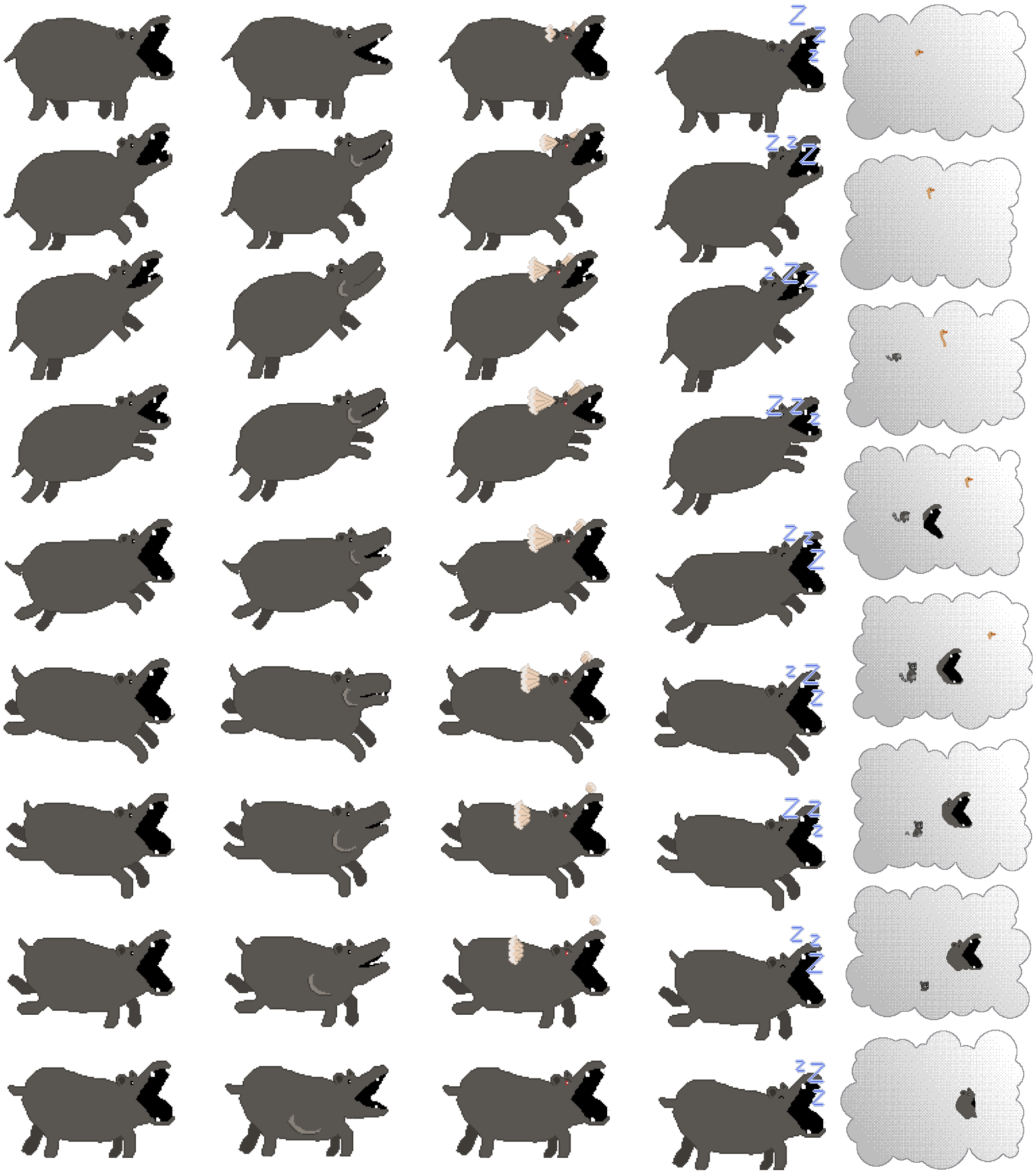


Figure 3.6: Hippo Animations

3.3.3.6 Special abilities

- Runs faster when ingesting fruits and berries

3.3.3.7 Relevance to the game story

The hippo is the first beast to chase the raccoon and the ostrich.

3.3.3.8 Relationship to other characters

The hippo wants to eat the raccoon and the ostrich.

4. Section IV – Zones

4.1 Zone overview

4.1.1 Recurring pattern

The zones in the game represent the whole game world. The last zone transitions back into the first zone, but with the increasing difficulty as in previous level transitions. So the difficulty level at the beginning of the second cycle of the zones is harder than difficulty at the end of the first cycle, and so on.

4.1.2 Zones overview

The 8 zones are called:

The Jungle	The Cave	The Dessert	The City	The Sky	The Water	The Ice Cave	The Volcano

4.1.3 Physical Description

All zones consist of a ground plane, and platforms in the air at different heights. Food appear on the different levels of the scene.

4.1.4 Critical Path

The player needs to collect mushrooms in order to cook dishes that slows down the beast. The mushrooms are never located at the ground level.

4.1.5 Objectives for all zones

Collect as much points as possible by surviving and keeping the beast at a distance. To keep your distance to the beast you need to feed it cooked dishes containing mushrooms. To end the zone the player needs to feed the beast enough to push it out of the screen.

4.2 Zone #1 - The Jungle

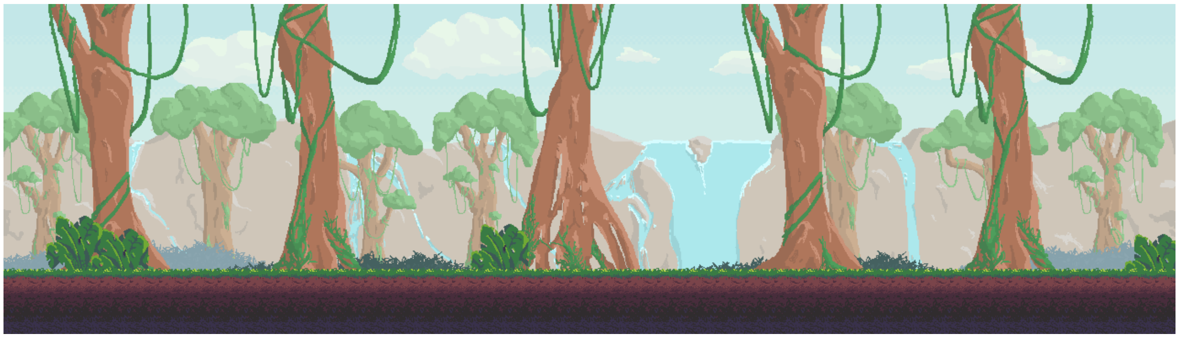


Figure 4.1: The Jungle zone environment.

4.2.1 Synopsis

The raccoon and the ostrich are out on a stroll minding their own friendly businesses collecting food when they suddenly picks the beasts ear instead of a berry. Out from the bush jumps a hungry beast! Here begins the story old as time, the law of the jungle, cook or be eaten, run!

4.2.1.1 General Description

In the back of the environment you are able to see tall rocky walls with waterfalls. There are tall trees with vines hanging down, and the ground consists of dirt and mud covered with green vegetation.

4.2.1.2 Physical Characteristics

All physical characteristics are over exaggerated for a cartoonish feel. It does not necessarily need to follow the logic and physical laws of the real world.

4.2.1.3 Zones that use area

Zones 1+n*8, n=0 (n: current complete cycles of all zones)

4.2.1.4 Connections to other areas

...Zone 8 - Zone 1 - Zone 2...

4.2.2 Cut scene

4.2.2.1 Introduction scene

The raccoon and the ostrich are walking in the jungle picking berries and other sorts of fruits from the bushes and trees. The raccoon does not notice a weird looking berry in a bush and tries to pick it, but it is stuck. He tries even harder and starts to pull with both of his hands. The “berry” rises with the raccoon hanging on to it, and it turns out to be a hippos ear. The Hippo roars in anger and begins to chase the raccoon and the ostrich.

4.2.2.1.1 Actors The main characters: The raccoon & the ostrich
Secondary character: The first beast, the hippo

4.2.3 Closing Material

4.2.3.1 Cut scene - Celebration scene

When finishing a zone the beast disappears out of the screen. The raccoon and the ostrich celebrates with a little dance. The environment seamlessly transitions into the next zone. A roar from the new zones beast can be heard and the beast jumps in from the left side of the screen, and the chase begins.

4.2.3.1.1 Actors The main characters: The raccoon & the ostrich

4.3 Zone #2 - The Cave

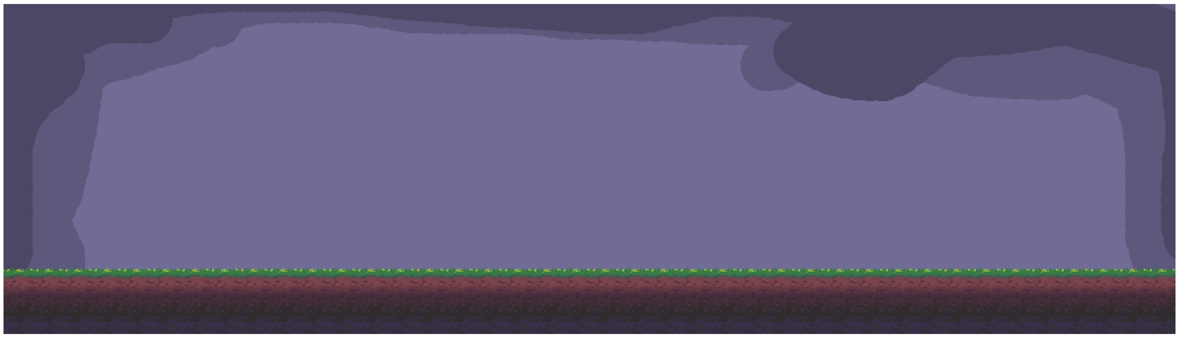


Figure 4.2: Early concept sketch of The Cave zone environment.

4.3.1 General Description

When the raccoon and ostrich finish the jungle zone they run into a cave through a waterfall similar to the once in the background of the jungle zone.

4.3.2 Physical Characteristics

This cave is darker in its light- and colour scheme. It is possible to see scary details in the background such as red eyes and fangs. Water is also a present element around the environment.

4.3.3 Zones that use area

Zones 2+n*8, n=0 (n: current complete cycles of all zones)

4.3.4 Connections to other areas

...zone 1 - zone 2 - zone 3...

5. Section V - Interface

5.1 Visual System

For the entirety of the visual system we have tried to keep the theme integrated. This means that our different User Interface (UI) elements are designed in pixel art as well, using most of the same colour palette as we have used in the rest of the game.

5.1.1 HUD - What controls

We have used two Heads-Up Displays (HUDS) in the game, placed at the top of the screen in each corner. The left HUD has three slots which are used to display which ingredients the players are currently carrying. The right HUD is used to display the amount of points the players have managed to get. We used similar sizes and art styles for both HUDS to keep a sense of symmetry in the scene and to make the displays distinct but not obtrusive.

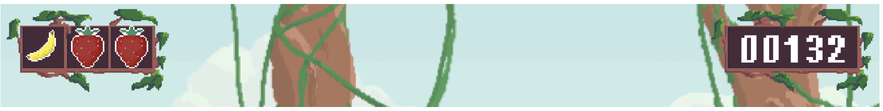


Figure 5.1: HUDs

5.1.2 Menus

We have three different menus in the game:

- Main Menu (See 2.4)
- Pause Menu (See 2.8)
- Game Over Menu (See 2.9)

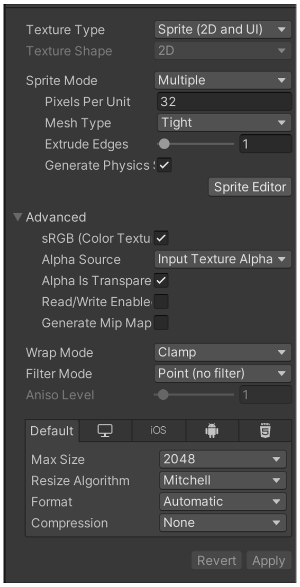
All menus are kept simple in their layout with a few, easy read, options which the players can choose from. They are also made in the same manner as the HUDs, using pixel art and the same colour palette as the rest of the game. Our choice of font and using capital letters is motivated by both increased readability and our ambition to give the game a retro feeling. The main menu sticks out from the rest as it is decorated with the logo of the game on a wooden board, all of which is carefully designed to portray a certain feeling. If you look closely on the wooden board (See figure 5.2) you can see carvings of both the raccoon and the ostrich. This was added to give the characters some personality and is inspired by the signs often seen at summer camps for children. Spending additional time on designing this feature for the main menu was a conscious decision, as it is the first impression the player will have of the game.

Figure 5.2: Main Menu Decorations

5.1.3 Rendering System

All graphical assets are rendered using Unity's built in *Sprite Renderer*[8]. This is the default tool when creating a 2D game in Unity and it served our purpose well. When importing our custom made 2D sprites into Unity we first had to adjust the import settings so that all assets follow the same rules. This is important both for the game to function properly and to keep the aesthetics of the game uniform. The settings we made for all assets when we imported them can be seen in figure 5.3 and were as follows:

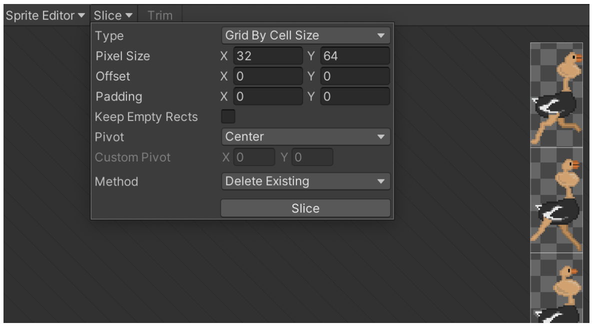
- Sprite Mode: If it was an animation we used Multiple, otherwise Single
- Pixels Per Unit: 32
- Filter Mode: Point (no filter)
- Compression: None



Texture Type	Sprite (2D and UI)
Texture Shape	2D
Sprite Mode	Multiple
Pixels Per Unit	32
Mesh Type	Tight
Extrude Edges	1
Generate Physics S	✓
Advanced	
sRGB (Color Textu	✓
Alpha Source	Input Texture Alpha
Alpha Is Transpare	✓
Read/Write Enable	☐
Generate Mip Map	☐
Wrap Mode	Clamp
Filter Mode	Point (no filter)
Aniso Level	1
Max Size	2048
Resize Algorithm	Mitchell
Format	Automatic
Compression	None

Figure 5.3: Sprite Import Settings

To import animations into the game we used sprite sheets containing all frames of the animation in a column. In order for Unity to understand where each frame of the animation starts and ends we had to slice the sprite sheet using the build in sprite editor. This tool let us specify how wide and how tall each frame should be and then apply the slice which can be seen in figure 5.4. In this example, each frame our animation of the ostrich eating is 32 pixels wide and 64 pixels tall.

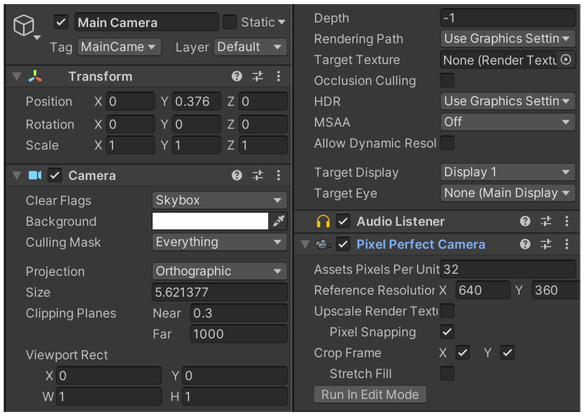


Type: Grid By Cell Size
Pixel Size: X 32 Y 64
Offset: X 0 Y 0
Padding: X 0 Y 0
Keep Empty Rects: □
Pivot: Center
Custom Pivot: X 0 Y 0
Method: Delete Existing

Figure 5.4: Sprite Slicing

5.1.4 Camera

The camera used in this game is Unity's standard camera using orthographic projection with the addition of a *"Pixel Perfect Camera"*[7]-component. With just a few settings provided, such as pixels per unit and reference resolution, this component makes sure the game looks equally crisp at different scales. This solution suited us perfectly, the only time we ran into problems was if the camera position was moved as all objects placed in the game depends on it being static. Our exact settings for the camera used in the game can be seen below in figure 5.5.


☑ Main Camera □ Static
Tag: MainCame Layer: Default

Position: X 0 Y 0.376 Z 0
Rotation: X 0 Y 0 Z 0
Scale: X 1 Y 1 Z 1

Clear Flags: Skybox
Background:
Culling Mask: Everything
Projection: Orthographic
Size: 5.621377
Clipping Planes: Near 0.3 Far 1000
Viewport Rect: X 0 Y 0 W 1 H 1
Depth: -1
Rendering Path: Use Graphics Settin
Target Texture: None (Render Textu
Occlusion Culling: □
HDR: Use Graphics Settin
MSAA: Off
Allow Dynamic Resol: □
Target Display: Display 1
Target Eye: None (Main Display

Assets Pixels Per Unit: 32
Reference Resolutior: X 640 Y 360
Upscale Render Textı: □
Pixel Snapping: ☑
Crop Frame: X ☑ Y ☑
Stretch Fill: □

Figure 5.5: Camera Settings

5.2 Control System

We discussed a way for two players to cooperate and play together and quickly settled for using the keyboard as control input for both. Using the same keyboard to play together we feel was more common in the past and we liked the idea of revisiting it, further enhancing the retro feel of the game. The Raccoon is controlled by pressing "W" to jump upwards and "S" to jump down a level. The Ostrich has the same actions but is controlled by the "up"-arrow to jump upwards and the "down"-arrow to jump down a level. Apart from jumping, the ostrich can also eat ingredients, this is triggered by the player holding down the "right"-arrow and running into an ingredient. These specific controls were chosen for two reasons.

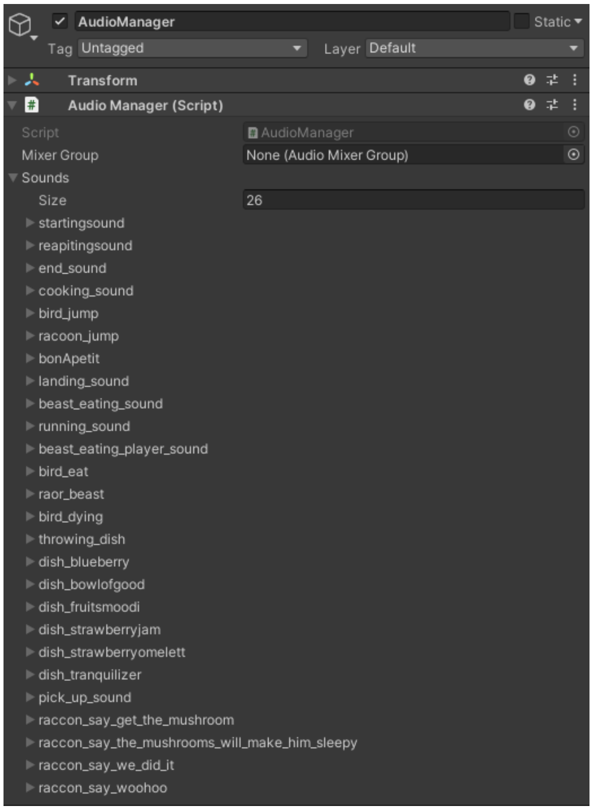
1. Both "W" and "S" as well as the arrow keys are common for these types of games and will feel natural and familiar to the players. This enables new players to quickly focus on the game itself rather than learning the controls.

2. Both the players controls are situated at each side of the keyboard which makes it easier for them to play together.

5.3 Audio

The audio in the game was either collected from various online sources or made by us. The sound effects and background music were sourced from *OpenGameArt* ,*Unity Asset Store*, *freesound.org* and *zapsplat.com* while the raccoon's announcement was made by us. To edit the sounds we used Audacity. When we used online resources, we made sure that all the audio was free to use and edit. In addition to this we also search for audio with a specific team in mind. This was to ensure that the sounds in the game were cohesive and presented a united front. We wanted the game to have a retro arcade like sound in addition to being comical.

The audio implemented using the AudioManager object (5.6) that contained a list of all the sounds. The Audio Manager script attached to the AudioManager object could be called to play the desired sound when an event was triggered in the code, like the characters jumping.



☑ AudioManager ☐ Static
Tag Untagged Layer Default

Transform

Audio Manager (Script)

Script AudioManager
Mixer Group None (Audio Mixer Group)
Sounds
Size 26

startingsound
reapitingsound
end_sound
cooking_sound
bird_jump
racoon_jump
bonApetit
landing_sound
beast_eating_sound
running_sound
beast_eating_player_sound
bird_eat
raor_beast
bird_dying
throwing_dish
dish_blueberry
dish_bowlofgood
dish_fruitsmoodi
dish_strawberryjam
dish_strawberryomelett
dish_tranquilizer
pick_up_sound
raccon_say_get_the_mushroom
raccon_say_the_mushrooms_will_make_him_sleepy
raccon_say_we_did_it
raccon_say_woohoo

Figure 5.6: Audio Manger Object

5.3.1 Music

The only music we have in the game is the background music that stay the same throughout the whole demo. For this we used parts of an existing piece and edited it, so it became repeatable. When expanding the game we will have different tracks as background music for each zone the characters visits.

5.3.2 Sound Effects

In the game we have four main categories of sound effects.

1. Action: sounds that corresponds to the players interaction
   - The characters sound when starting a jump
   - The ostrich eating sound
2. Individual character: sounds that indicates what the character are doing,
   - Raccoon cooking [3] [2] [4]
   - The character hitting the ground after jumping
   - The running sound when characters are in contact with the ground
   - The raccoon scream when the player looses
3. Event: sounds that signifies that something has happened
   - A character picking up an ingredient
   - The beast eating something
   - The beast eating the player at the end of the game
   - The raccoon throwing the dish when finished [1]
4. Announcement: sounds giving the impression that the characters are saying something
   - The raccoon saying what dish he made
   - The raccoon hinting to gather mushrooms
   - The raccoon celebrating

The announcement were made by recording our voices using a regular computer headset and then modifying the voice in Audacity. For the majority of the announcements we only changed the pitch without altering the tempo. For the other few we increased the tempo a little as well to either amplify the cartoonish feeling of the game or to create a sense of emergency. An example of where we also increased the tempo are the messages which we triggered when the hippo was close enough to the players to enter the ”angry”-state. In this scenario we played either ”The mushrooms will make him sleepy” or ”Get the mushroom” in high pitch and high tempo to give the effect that the raccoon is hinting the players to hurry up and gather more mushrooms.

5.4 Help System

For the game we wanted to facilitate player exploration, because of this we do not provide much help to the players. The two forms we have are:

1. A menu that explains the controls of the character accessible from the main menu.2.4
2. The raccoon telling the player to collect mushrooms if the beast gets angry. 5.3.2

6. Section VI – Technical

6.1 Target Hardware

The game is targeting the PC market. Since Feed It is a couch co-op game, you either need to have two players on the same keyboard or binding one or both players actions to controllers. This makes the game unable to be played on smart phones and tablets in its current form. The main target platform is PC, where the game with its tiny size of less than 10 MB and minimum resolution of 640x360 pixels could run on any modern or old computer that can run Windows 10 or Mac OS. The game could also be ported to consoles without requiring extra optimisations, but that is not something that is planned. It may be considered depending on how successful the Steam launch is.

6.2 Development hardware and software

To target the PC and Mac platforms the game will be developed on machines running those operating systems. The hardware required for developing the game is similar to the one required for running it, which is any computer capable of running Windows 10 or Mac OS. The Unity engine is the most performance heavy part of the development, and it will run on any computer with a processor of 1.1GHz or more and 2GB RAM or more.

The software used for development is Unity as the game engine and GitHub for version control. To coordinate the team we used Trello for task management and Miro for brain storming and concept art. For the graphics Paint3D is used for place holders, Aseprite and Adobe Illustrator are used to create sprites and Aseprite alone is used for animations (See section 7.3). Music and sound effects are created using Audacity which is documented in section 5.3 audio.

6.3 Development procedures and standards

All sprites and sprite sheets needs the following settings to display clean and sharp pixel art (also mentioned in 5.1.3):

- Pixels Per Unit: 32
- Filter Mode: Point (no filter)
- Compression: none

All game objects should be prefabs that are then dragged into the scene or instantiated using scripts. That way we can easily change properties or sprites for the game objects without needing to change it for each single copy. When creating a script on a game object, all parameters that you would want to iterate on and all constants should be in the form of private serialised fields. This will create properties that are visible in the editor and make development more efficient.

Code should be reused using inheritance in all cases where it would make sense. For example the two playable characters, the raccoon and the ostrich both derive from a general player class that handles collision, jumping and calculates if grounded. The subclass should only contain functions and data that only one of the objects uses.

All zones are randomly generated using chunks. These chunks should be 50 tiles long and 11.25 tiles high. Out of these 11.25 tiles height, 2.25 tiles should be the ground. That leaves 7 tiles for the playable area of the demo and the other zones that do not use the digging mechanic of the raccoon. For the above ground area, all platforms should be semi solid platforms, which are platforms that only have an active collision on the top and the playable characters can jump through if they press down. The platforms should in general be of two different layers, one near the ground for the ostrich and one high up that only the raccoon can reach by jumping off the back of the ostrich.

Ingredients should be placed using ingredient spawners, which sets the spawn chance of all ingredients. For the first zone, the most common ingredient should the strawberry, followed by the egg and the banana. They should commonly appear near the ground where the ostrich and the beast can reach them. Blueberries should be more rare, and be more common to spawn near the high platforms. Mushrooms should exclusively only appear higher up on the screen, with the prime goal of the raccoon to collect them using teamwork with the ostrich. In certain cases the level generation could deviate from the norm and let the ostrich reach a mushroom, but that should be rare.

6.4 Game Engine

The game is developed using the game engine Unity. Unity was our choice since the team members had used the engine before in previous projects and the engine is free to use if the gross annual revenue is less than $100,000 in the previous fiscal year, which the team falls under. The version used is the Unity LTS 2019.4.9f1. Long time release version was used since a lot of changes can be applied in a new Unity version, and the team wanted a stable development environment with promised bug and stability fixes for two years. Another important factor with the engine is to be able to build the game for all targeted platforms. Unity supports most popular platforms, among them windows, mac and WebGL which is our prime targets for the demo and the finished game.

6.5 Network

A big focus on the game is high score. To motivate the player to take more risks and stay in one zone for longer. For this to be as efficient as possible it is important that the player can compare their personal best score with other players. A global high score leader board where all users high scores are uploaded will solve this issue. For the demo there will be no leader board. Instead the players would need to buy the full game to get that feature, or create their own self made leader boards and keep them updated. For the full game Steam leader boards will be used, that way we would not need to create a login system with a corresponding server and then create and keep a leader board updated on said server. Instead all the ground work are already done for us and we only need to call the Steam API for the features we want.

6.6 Scripting Language

Unity uses C# as the main language. C# is a compiled language which will lead to better performance, and it is well integrated in Unity. To use the scripting language on a game object you need to derive from the base class MonoBehaviour, with the important functions *Start* that works similar to a constructor in that it should initialises and find variables and components, *Update* and *FixedUpdate* that should handle the logic. One drawback with using C# as the scripting language is that compiled languages makes modding more complicated. Feed It has not been designed with modding in mind so we do not see that as a problem.

7. Section VII – Game Art

7.1 Concept Art

It did not take long for us to agree upon using pixel art as our style of choice. We set out to find ideas and inspiration from different games and sources which utilised this and gathered it all in a google document. Using this collection of thoughts we then tried to establish our ideal art style, going into details such as eyes, level of realism and outlining. We were also quick to settle upon which resolution we wanted to work with. One of our team members had experience working with pixel art games in the past and emphasised the importance of using the same resolution across all assets. The resolution we set was 32x32 pixels for each tile of the game, using a 16:9 ratio for the game itself. We felt that this resolution gave plenty of room for creativity while not offering too much room for details which would have required more time to create. Here is a chart of how our main character, the ostrich, evolved from concept art to the finished sprite:

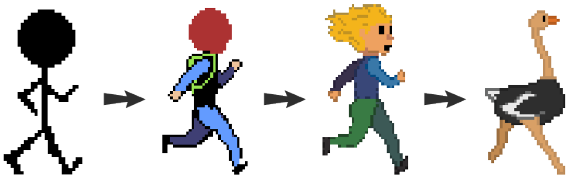


Figure 7.1: Main character progress

This is a similar chart of the evolution of our secondary character, the raccoon:

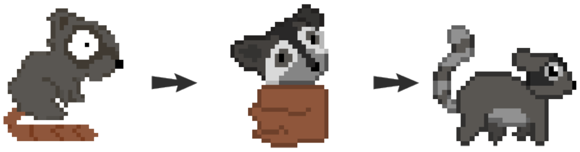
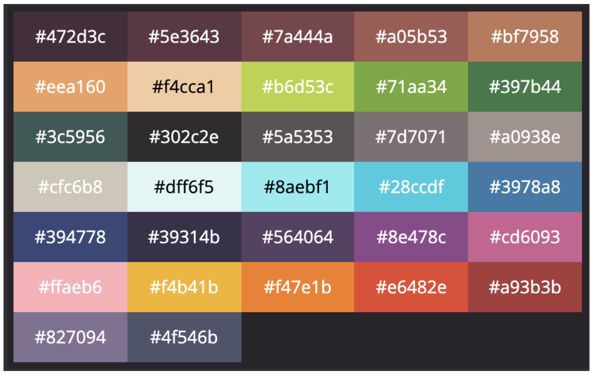


Figure 7.2: Second character progress

We did not spend an extensive amount of time developing different types of concept art however, but rather decided on a theme and then worked in an iterative process. For example, a keen eye can detect quit a large change during the last step of our main character. We decided on using an ostrich instead of human character to increase the ”cuteness” of the game, while still keeping the form factor of the character. Had we spent too much time on developing concepts for different human characters during the concept stage it would have been wasted.

7.2 Style Guides

As mentioned in the section above, we gathered all ideas and inspiration in a document called ”Art Style Guide”. This is where we set all the aesthetic features of the game. One of the major aspects we had to decide on was a common colour palette which we would try to stick with for all aspects of the game. For this we used a popular site called ”Lospec”, and the palette we decided on can be seen in figure 7.3. Some examples of the other features we decided on were:



#472d3c	#5e3643	#7a444a	#a05b53	#bf7958
#eea160	#f4cca1	#b6d53c	#71aa34	#397b44
#3c5956	#302c2e	#5a5353	#7d7071	#a0938e
#cfc6b8	#dff6f5	#8aebf1	#28ccdf	#3978a8
#394778	#39314b	#564064	#8e478c	#cd6093
#ffaeb6	#f4b41b	#f47e1b	#e6482e	#a93b3b
#827094	#4f546b			

Figure 7.3: Colour Palette

Eyes: We noticed that there were a large amount of different approaches to drawing eyes of characters but they seemed to rely on the size and resolution. A common theme when drawing smaller characters of limited resolution were black eyes, sometimes with a sole white pixel to create some expression. The pros of using this tactic was efficiency which led to us choosing to go with it. However it does restrict the amount of expression we can give the character.

Outlining: It is not unusual for graphical designers to use outlining when drawing pixel art, as it makes the sprite easier to distinguish from the background. We had a brief discussion on this topic and decided on using an outline of the same colour as it's adjacent pixels but with a darker hue. This was applied to all moving characters, but the static ingredients were made with a white outline to increase their salience in our fast paced game.

Size: The size of each character is very important as it heavily influences the game play. The sizes were therefore discussed with all members and those of us with more experience in pixel art games used it to guide the decision. We agreed on using a single tile (32x32 pixels) for both ingredients, dishes and the raccoon. For the ostrich however we went with two tiles in a column (32x64 pixels) as it had to be large enough to have the other player ride it without creating space issues.

7.3 Characters

The three characters involved in the game so far consists of the ostrich and the raccoon, controlled by players, and the Hippo, controlled by the computer. The choices of animals were somewhat arbitrary and could be replaced in a further development of the game. For example you could unlock additional characters after reaching certain scores or completing hidden achievements. Each character has several custom drawn animations for their different states, toggling between these animations is done by code. All characters and animations were drawn in a software called ”Aseprite”, which is a dedicated program for pixel art. Here is an example of how the design process looks:

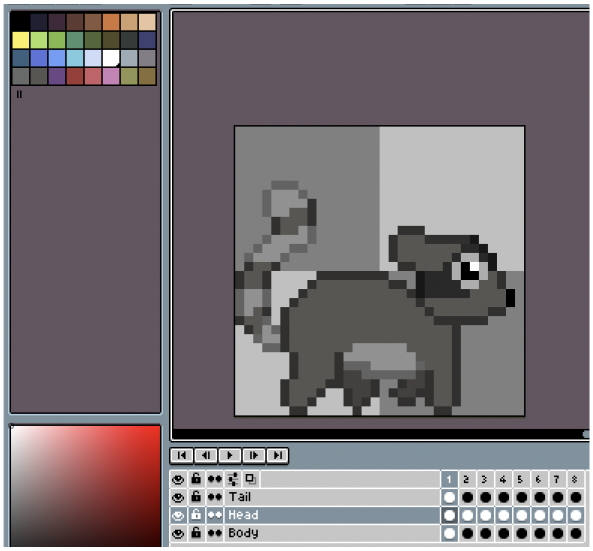


Figure 7.4: Drawing in Aseprite

The software supplies a smooth workflow for layering and animation through a timeline of frames, all of which is displayed in a matrix at the bottom. Here are the animations created for each character so far:

The Ostrich: Running, Jumping, Falling and Eating

The Raccoon: Running, Jumping, Falling, Riding, Start Cooking, Cooking and Stop Cooking

The Hippo: Running, Eating, Running Sleepily and Running Angrily

7.4 Environment

So far the only environment of the game is a jungle, but if the game would be further developed we would add several different environments which the players would travel through (See section 4.1). In that scenario we would switch the environment and the chasing animal each time the players manages to out run the hungry beast. Some of the other environments we have discussed are a dessert where you are chased by a lion or a an ice cave being chased by a polar bear etc. The implemented environment adheres to the same resolution standards as all other art and is displayed in the game in layers. By separating the environment art into different layers we are able to move them at different speeds in relation to each other, thus creating a parallax effect. To amplify this effect even more we faded the colours of the layers we wanted to appear more distant to the player. Unlike the character and miscellaneous art, all graphics for the environment and menu's was created in Adobe Photoshop. This was due to personal preference as we had two different team members working on environment and character art. This worked smoothly as we made sure we used the same colour palette and pixel resolution.

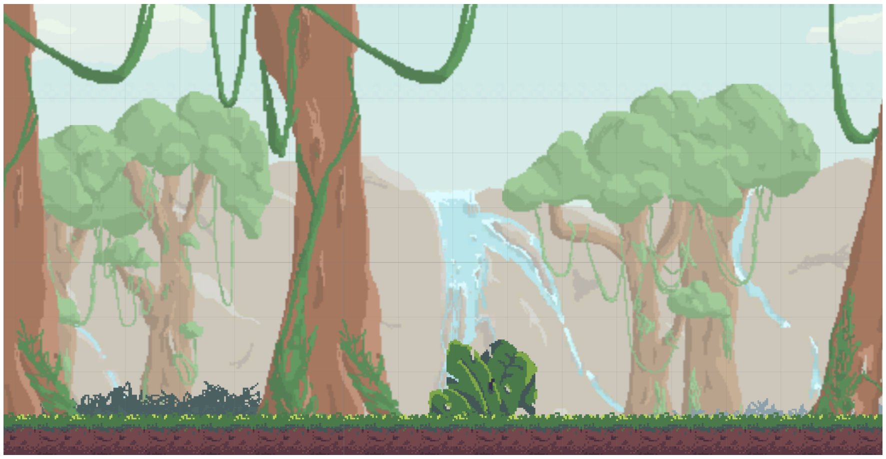


Figure 7.5: Environment

7.5 Cut scenes

We have not added any cut scenes to the game yet, but it is another thing that could be beneficial to add in further development. We have a vision for how we would like to implement the cut scene: The players slow down as the beast stop chasing them. They then cheer audibly as they've manages to escape the beast, and we could add graphical effects to reward the player for completing the level. After a brief celebration a loud roar sounds after which a new beast appears to start chasing the players again. This is where we would change the environment as well. This cut scene would be a great place to save user data and load necessary level data before initiating the chase again with a new beast.

7.6 Miscellaneous

Our miscellaneous assets consists of ingredients and dishes. As discussed in Section 2, the game circles around gathering ingredients and cooking dishes. So far we have added five different ingredients and six different dishes. The dishes are drawn with a slightly tilted angle to give some depth and are outlined by picking the colour of adjacent pixels and then lowering the hue value. All ingredients except the mushroom has a white outlining to make them more salient as previously mentioned. The reason why the mushroom carries a black outline is because it is the only asset drawn by one of the coders, we let it be in the game as a tribute to their hard work. However there are additional benefits of keeping it this way, the black outlining separates the mushroom from the other ingredients which indicates its difference to the players. The black outlining also make it look more ominous than the other ingredients, which again indicates it will have another effect.

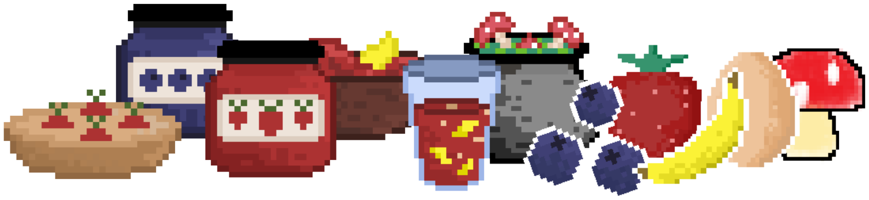


Figure 7.6: Miscellaneous

8. Section VIII - Management

8.1 Detailed Schedule

This section present what have been done in detail and a plan for the future from now until release. The future plane would be based on everyone working on this as a subproject, meaning that it will take a longer time. Therefore the future plan is not in timestamps but in likely order of execution. What has been done:

Work started on 06.11.2020 with the first meeting.

10.11.2020 - We decided on what idea to go for.

12.11.2020 - Prototyping in unity started.

15.11.2020 - Playable endless running with one character, random levels and simple actions implemented.

24.11.2020 - Two playable characters functioning as intended, ingredients now spawns, upgraded graphics and animation, and added background sound.

01.12.2020 - Beast moving when it eats, cooking logic is finished and main screen is created.

04.12.2020 - Pause and game over many implemented, added dishes for after cooking, essential animation finished, score display and website started.

13.12.2020 - Many minor changes in game-play, final graphics, animation and sound done, major bugs fixed.

14.12.2020 - Demo and website finished.

15.12.2020 - Final presentation.

Plan for the future

1. Finalising some art to use in marketing.
2. Start building avenues for marketing.
3. Finalise demo and video.
4. Conduct user test play sessions. To get first hand feedback on features.
5. Start looking for investors and or funds.
6. Implementing the remaining 7 zones.
7. Launch on steam.
8. If doing well look into other platform such as switch and mobile.

8.2 Budget

The project would in the start have a small budget where all of us would contribute, this would go to legal fees and other programs that we might need in the development phase. We plan to start making the game as a side project so for the first part there would be no salaries. In the later stages and after the trailer is made, we would start looking for grants and potential investors with the aim that at least one of us can start working on the game full time. When it comes to the game sales themselves we plan to sell the game for 7-9 Euro on Steam. We are putting the price point here to stick with the indie game pricing trend and to be able to take advantage of the Steam sales culture. If we price ourselves to low there will be little too gain form putting the game on sale, which is when a lot of players might buy our game[5].

8.3 Marketing

Because of the low starting budget of the project we will start by creating an online presence on free platforms. Relates to both our game and other indie game developer. There are several avenues and places we could do this and each of us will choose a platform like Reddit or Twitter to focus on. During this time, we will follow the 70/20/10 rule. 70% Organic content, 20% other articles and or other sources and 10% Direct promotion[9].

We will also aim to post once a day and interact more frequently with other content creator that does the same and answer questions posted to us. The post will mostly focus on using GIFs, videos or images to generate interest. Then we will start working on making a strong trailer that will hook people and make them even more interested[6]. When we have a small following of people, we will start approaching press.

8.4 Risk Analysis

What could happen	**To prevent**	**If it happens**
Nobody likes the game	Test with different people consistently	Find out why, we might have to scrap the idea
Technical difficulties with game or engine, that is too hard to fix	Test often so we know when, where and why the bug happens, making it easier to fix	Contact people that might be able to help us
Somebody cannot work anymore (quit, industry, +++)	There is little to do here	Find a replacement or put somebody else in charge of their responsibilities
Loose motivation	Check in on each other, make sure that everyone does what they are most motivated to do	Try to find other more interesting tasks
Marketing not effective	Make sure to follow guidelines and tips so that what we do is effective	Evaluate what we are doing wrong and then try other methods
Running out of money	Make sure to have a detailed budget and stick to it	Find new or other sources of revenue
People like the game but do not continue playing or engaging	Make sure in the development that people like and want to play	Find ways to motivate players to pick the game up again

8.5 Localization Plan

We will release the game first in the English speaking parts of the world. If it gains traction we might look at translating it to other languages. Because this is a game with a small amount of text and dialog it will hopefully not take too much time or money.

8.6 Test Plan

We want to aim at testing the game ourselves at least every time we make a change, both to catch bugs early and to have a good overview of how the game feels and is progressing. In addition, we also want to expand our testing to other people, starting with getting friends and family to test it. During the testing with players, we want to first and foremost use the methods *structured observation* and *think aloud.* This will hopefully give us a base for what to look for every time we have somebody test our game as well as gain insight into what the players are thinking. Because this game is intended as a multiplayer co-op game it is of utmost importance for us to test two players together.

Bibliography

[1] cartoon-throw.wav by copyc4t.

[2] cooking tomato sauce by trouby.

[3] Cutting$_{o}nion_{s}peedy.mp3bytairakomori$.

[4] Microwave.aif by jasonelrod.

[5] 2017, J. B. r. A. Indie game pricing is ”more art than science”.

[6] Lovato, N. Marketing your indie game with a 0 budget, Nov 2020.

[7] Technologies, U. Pixel perfect camera.

[8] Technologies, U. Spriterenderer.

[9] Wright, G., Hanlon, A., and Chaffey, D. 70:20:10 marketing, Sep 2020.