Document Version: 2.0

GAME DESIGN DOCUMENT

CRAB JOURNEY

A small crab's **BIG** journey

Contents

1 Game Concept Document

1.1 Introduction

Crab Journey is a 2D, pixel-art, puzzle, platformer game that focuses on the **big** adventure of a *small* crab. The crab grew out of his shell and ventured into the sea looking for a new one. The player plays as this crab character from a side-scroller, 3rd person perspective, exploring the underwater depths—from collecting pieces of trash, cutting your way through the trash to climbing around the map. To progress deeper into the sea, you will have to solve puzzles. Solving the puzzles requires collecting trash and exploring the map. What does the crab do? Main actions that gameplay consists of involve puzzle solving...

Crab Journey is a short game which implements fun and unique mechanics. The game will be an original idea with a lot of other media influences and concepts merged.

From the warmth of the shallow beach to the loneliness of the ocean depths, climb corals, explore underwater environments, and cut your way through to the end. Highlighting the reality of ocean life today, highly polluted with plastic, the game gradually introduces the problem into the gameplay.

For players of all ages who enjoy a relaxing experience when playing games.

1.2 Game Analysis

Game Description	
Genre	• Platformer • Puzzle • Adventure
Theme	• Nature • Ocean • Pollution
Content	• Casual • Exploration • Discovery
Style	• Cartoon • Pixel Art • Stylized

Game Elements	• World Interactions
	  ○ Collecting
	  ○ Obstacles
	  ○ Destroying
	• Exploring
	• Seeking
	• Movement
	  ○ Left-right

	○ Jumping ○ Climbing
Game Sequence	• Mostly Linear
Players	• 1

Game Reference	
Game Taxonomy	• Crab Journey is a Fictional game with Game/Narrative elements.
Player Engagement	• Emotional • Narrative • Mental
Reference	• Ori • Hollow Knight • Celeste • Another crab's treasure • WALL-E • A Short Hike

Game Technical	
Technical Form	• 2D pixel
View	• 3rd person side scroller
Device	• PC
Game Engine	• Unity

Game Sales	
Consumer Group	• Casual players • All age groups • Players who enjoy a 'cutesy' art style with an inspirational message
Payment	• One-time purchase
Estimated Price	• 3-5 euros
Marketing Ideas	• Sending free copies to game review channels • TikTok promotion • Communication with activism groups connected to ocean pollution and ecology • Potential strategies and creative concepts to promote the game.

Unique Selling Points	• Cute main character • Spreads awareness about pollution in a fun way • Crab

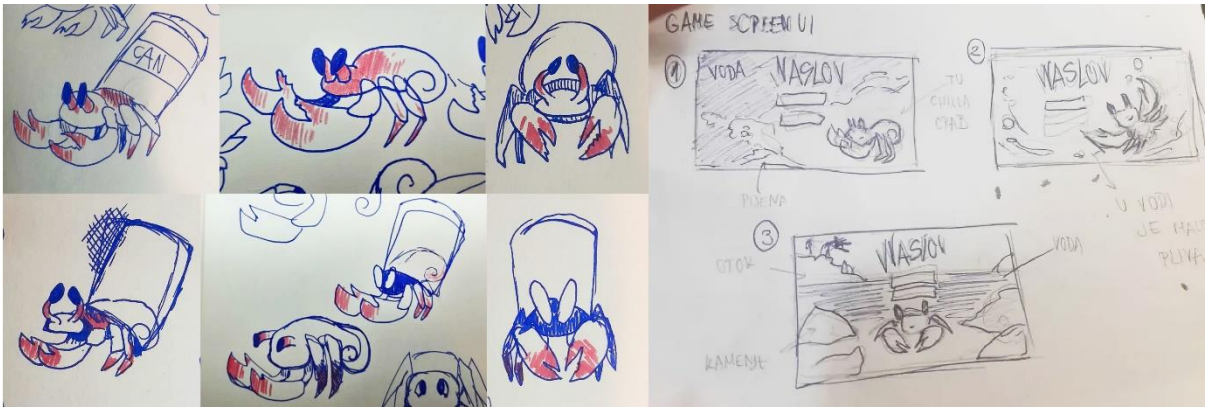
1.3 Game Atmosphere

- The game's atmosphere is characterized by a gradual change in atmosphere from a cheerful, relaxing and laid-back theme, that grows darker as the player travels to deeper ocean levels. This change will reflect different aspects of the game from motive and asset choices to music and colors.
- Most of the game consists of ambiental music with satisfying and loony background sound effects as well as movement sounds. The music sounds reflect the game atmosphere, which is in the beginning relaxing and cheerful. As the game progresses and as the crab travels to deeper levels of the ocean, the water becomes more polluted, and the music becomes more eerie.

These are examples of images that we used as inspiration for art style:

Document Version: 2.0

This moodboard shows sketches for game character and scenes we have made ourselves:

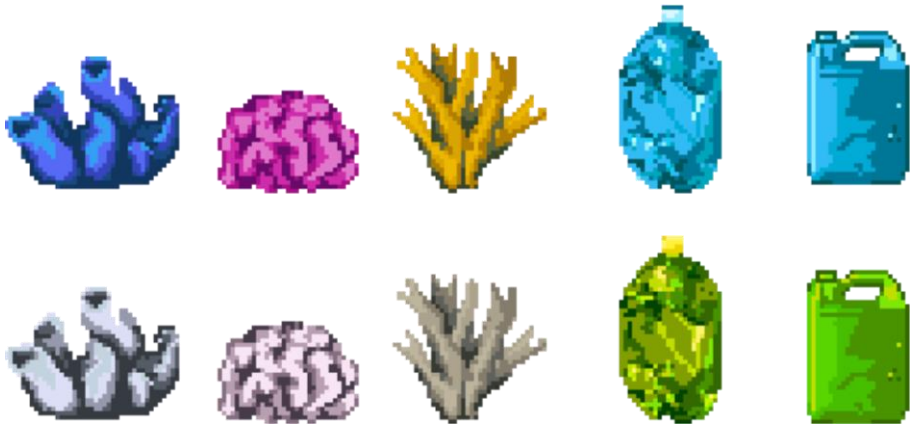


- Main character is a small hermit crab whose claws are big for trash collecting. The crab wears a shell. His eyes are big showing his cute and curious nature.

Character design of the small hermit crab wearing a tin bucket as a shell:

- Overall, the color palette is vibrant creating noticeable visual language of the game that catches the attention of players. Bright orange and red color choice of a small crab creates accent with the mainly blue background and nature of the sea in complementary way. Through the game, as the more polluted elements appear in the game, the color choices adapt to them, referring to, for example, toxicity of the plastic or the color disappearance in the coral bleaching caused by global warming.

The following are examples of indicating coral bleaching and toxicity in the game:



- Assets mostly consist of parallax 480x270 pixel backgrounds, tiles created in 24-pixel grid, and assets like ocean life motives and trash that vary in size. Backgrounds graduate from the seashore and deep ocean, all the way to ocean caves.

These are examples of assets made for the game in a 24x24 pixel grid:

1.4 Gameplay

- When you open the game, you'll be warmly welcomed with a splash screen with the team's name and then progress to an animated title screen and main menu.
- When the game starts, a short intro will display the character's motivation of going into the depths in search of a new shell.
- Moving the crab character from left to right (with the keyboard) in the usual platformer way and by climbing around the map, the player will move through the ocean environment and progress further into the ocean depth.
- The player will encounter pieces of trash floating through the ocean and will be able to collect them when the crab gets close enough to the pieces.
- In between different ocean depths where the player can cut through bigger trash objects and collect smaller trash objects directly into the crab's tin can, there will be puzzle rooms which will contain puzzles that will use the smaller pieces of trash collected.
- These puzzles also involve collecting *special* pieces of trash (like levers) and using these pieces to solve the puzzles.
- The player is encouraged to achieve 100% game completion and to do so can return to previous parts of the game. This will be implemented through keeping track of how many pieces of trash the player has collected in different depths, and which puzzle rooms they've managed to solve. This completion percentage is not directly displayed to the player.

- Additionally, to the player, the completion will be showcased through dynamic changes of the environment (like less opaque color overlays and small environmental changes in details). 100% completion will be showcased to the player through a small scene at the end of the game story.

1.5 Key Features

- One level, but with multiple visual scenes (exploratory platforming), and a puzzle room.
- There is a possibility of multiple levels if development time allows.
- One character.
- About 20 minutes, but there is a possibility of more if development time allows.
- There will be a soundtrack of songs and ambience sound effects alongside sound effects to immerse the player into the game.
- The player is encouraged to achieve 100% game completion.
- The game will be singleplayer based.
- There is only one game mode, a story mode.
- There is no applicable customization option.
- There is only one difficulty for this game.

2 Design Document

2.1 The Story

The small little crab finds itself in a tough spot: it has overgrown its shell and needs a new one! As there are no new shells on the beach, it finds a replacement tin can for its back. This is not acceptable!

The brave crab ventures under the sea to find his new shell which it can call home. Along the way it gets into many puzzling adventures and finds itself in a predicament: the ocean floor is in a polluted state, which the crab has never seen before. It's now surrounded with many objects it's not familiar with and collects them as it likes collecting things. After traveling to the bottom of the sea, it finds a new and beautiful shell for its back.

Its journey is not easy, as it is a Crab's Journey™.

2.2 Game Narrative

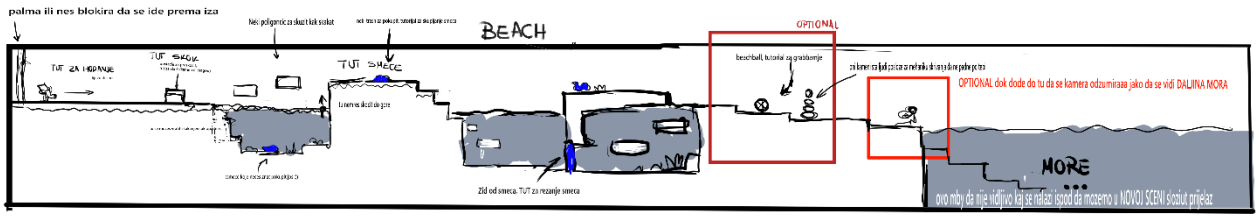
- Visual storyboard

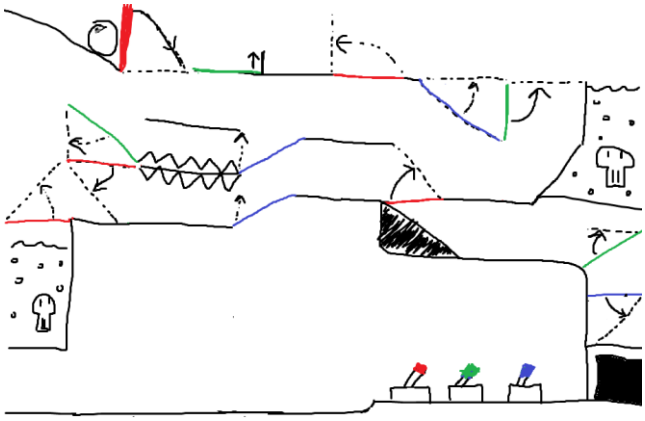
In-game environments (beach, shallow sea, deep waters):

- Level design

The player will start on a beach scene where they will be faced with a tutorial in which they will learn the mechanics of the game. After the beach scene comes a shallow water part where the player needs to use these mechanics to explore the environment and go deeper into the sea. The deeper the player goes, the more polluted the water will be. The access to deeper waters will be blocked by a mandatory puzzle room. After the puzzle room, the player needs to keep exploring the deeper part of the sea to find the shell and end the game. The first puzzle room will contain a set of podiums for levers and rotatable walls. In the puzzle room, there will be a cage with the first lever you can unlock with a certain amount of collected trash. This will motivate the player to backtrack and find the other levers for the puzzle spread across the level before. The player's job, after finding all of the levers, is to figure out in which order to interchange the positions of the walls to lead the pearl outside of the puzzle. When the puzzle is solved, the player will pick the pearl up and throw it into the clam (shown as a blacked-out box) to grant him access out of the puzzle room. There will be some sort of punishment if the player doesn't move the pearl to the right spots, for instance liquids or spikes that destroy the pearl and force the player to start the puzzle over. Restarting the puzzle will mean putting the pearl back into the starting position of the puzzle.

Sketches of the level designs for the beach scene and the puzzle room:





- Scripted content

  At the start of a new game:

  [Black screen - underwater sounds.]

  Text fades in, line by line:

  > Crustaceans like hermit crabs outgrow their shells,
  > As they age and grow, their old home no longer gels.
  >
  > Sometimes, when the perfect shell isn't found,
  > They adapt to whatever is lying around.
  >
  > Even if that something is just a rusty tin can,
  > They make do, the best they can.
  >
  > So, to avoid finding themselves in a ***pinch*** once more,
  > They set out on a journey, to find a new door.

2.3 User Experience Design

2.3.1 User Control

It has been decided that the game will be controlled only by using the mouse and keyboard.
For controlling the player, the default key layouts will be (susceptible to change if testing the game proves other keys to be more useful):

- **WASD** for moving the character *(WS – Y-axis climbing, AD – X-axis walking);*
- **E** for interacting with objects (picking up objects, pulling levers, cutting trash);
- **G** for grabbing;

- **ESC** for opening the pause menu;
- **SPACEBAR** for jumping;

Furthermore, interacting with the UI will be possible by moving the cursor and pressing the buttons using the left mouse button.
The camera is not controlled by the player but instead simply follows the player. In certain areas it can transition to a view of the whole area.
The settings that the user will have control over are as follows:

- Options to **change audio settings** (overall volume).
- Options to **change video settings** (resolution for window-mode and toggling full-screen mode)

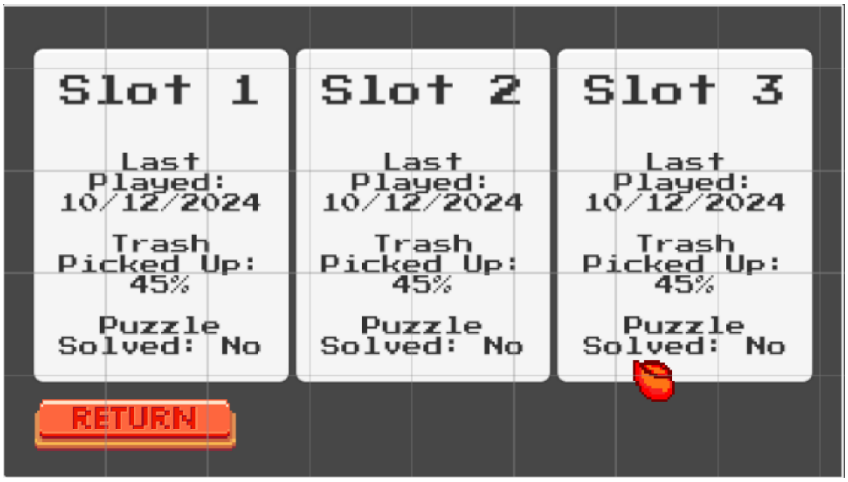
2.3.2 User Interface

Visual Hierarchy:

- Visual components are organized in a grid layout of the screen and are aligned with the game visual elements.
    - The layout of the game screen is a 480x270 grid of pixels (scalable to different resolutions), and that grid is subdivided into a 24x24 pixel square grid.
- **The Main Menu Screen** contains the following: "Continue", "Load Game", "New Game", "Settings", "Quit Game" buttons.
    - Buttons on the Main Menu Screen are aligned with the game title and the title font is larger compared to other UI elements.

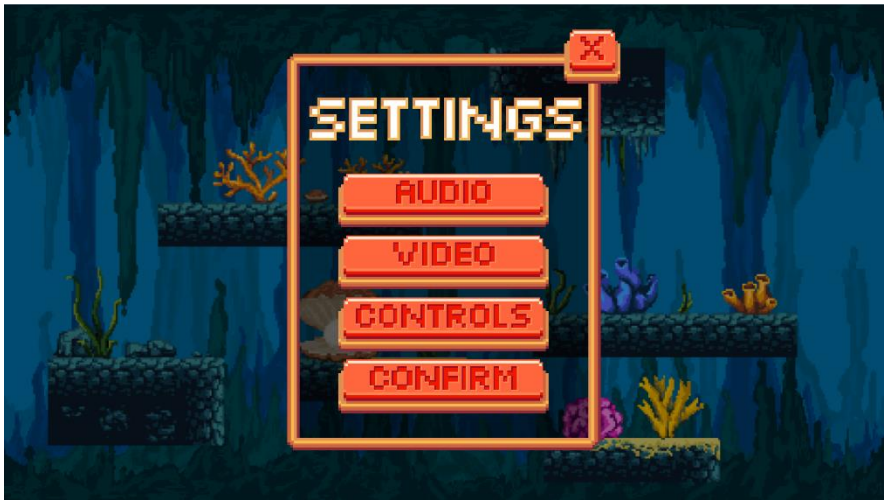
  This is a sketch of the main menu screen layout:

- **The Choose-a-Save Screen** contains the following: "Save Slot 1", "Save Slot 2", "Save Slot 3", "Return" buttons.
    - The save slots are three larger button objects which contain information on the save inside themselves. They are selectable depending on what the function for them is (from which screen the player arrived at the Choose-a-Save screen).



- **The Pause Menu Screen** contains the following: “Resume”, “Settings”, “Menu”, “Quit” buttons. This is a sketch of a layout for pause menu:

- **The Settings Menu Screen** contains the following: “Audio Settings”, “Video Settings”, “Control Settings” submenus and “Confirm” button.
  This is sketch of the settings menu layout:



Font sizes, colors, visual design:

- Font is recreated in pixel grid to create harmony with pixel-perfect layout of the game assets.

- The font of the main buttons is 80-point sizes, while the font size of game title is noticeably larger. In all game components, the font sizes are readable and often slightly bigger than the standard font sizes due to pixelized letter style.
- UI design of the components is visually related to game theme adding to the game atmosphere, it's consistent with the art style and orange referring to the main character. Consistency is also present in components one to another to create intuitive experience where the main actions are visually highlighted.
- Colors of the components are chosen to contrast the background in order for them to be visible and inclusive for players.

Feedback mechanisms:

- Game buttons in UI Menus will have enabled, hovered and pressed states that will be indicated in color changes and theme animations.

HUD:

- HUD will consist of the indication for the amount of trash collected and a pause button.

These are sketches of what HUD will look like:

In-Game Tutorials/Tooltips:

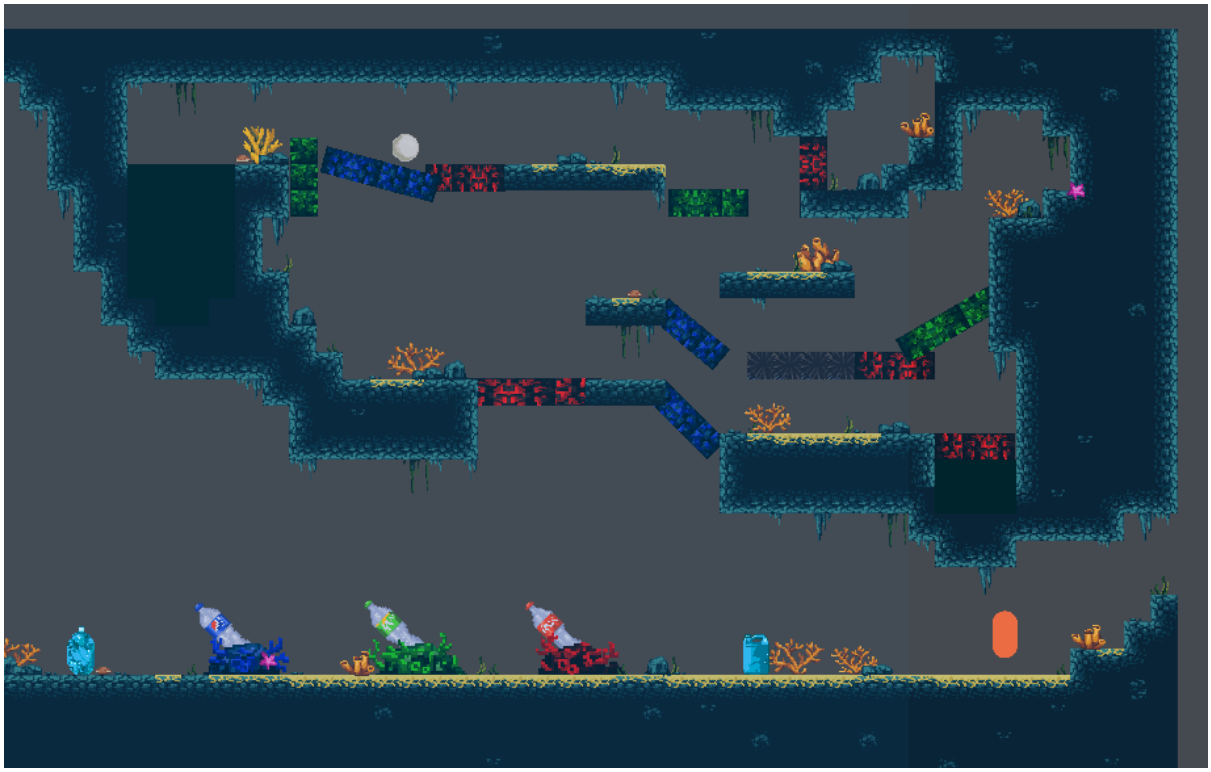
At the start of the game there will be a short tutorial for user controls and mechanics in the game

- Left-right movement keys
- Climbing
- Collecting trash
- Destroying a trash wall

2.3.3 Camera View

The camera will be of a 3rd person side view perspective. It usually follows the player object (crab). In the puzzle rooms, the camera will change into a full view side perspective of the scene.
While following the player, the camera smoothly adapts to the player's movement to display the area where the player is heading and moves back to center the player when necessary.

Camera in normal play mode



Camera view in the puzzle cave

2.3.4 Audio & Sound F/X

Background music

- Changes depending on the scene
- Puzzle room music
- Background music for main menu

SFX

- Menu interactions
- Picking up objects
- Grabbing and throwing objects
- Cutting trash
- Interacting with levers and similar things
- Movement
  - Walking
  - Jumping and landing
  - Climbing
  - Entering and exiting water

Changes in game state

- Solving puzzles
- Progressing deeper

There will be constant ambient background music. At the start of the game, it's a playful, chill tune and as the player goes deeper into the sea the music will change into more eerie music.
Collecting items, entering and exiting the water, cutting trash, picking up items, throwing items, interacting with levers and similar things will all have sound effects to signal to the player that the action was completed.
Solving puzzles will have sound effects indicating their completion.
Likewise, movement will be accompanied by appropriate sound effects to allow the player to more strongly feel the execution of the actions. For that, walking, jumping, landing, climbing and such, will all have appropriate sound effects.

2.4 Player

2.4.1 Player Definition

There is only a single player, the crab character, which the player controls and uses to interact with the world.
The crab can:

- Move
  - Walk
  - Jump
  - Climb specific materials (corals, …)
- Interact with the environment
  - Collect trash
  - Grab certain objects (pearls, …)
  - Pull levers
  - Cut bigger pieces of trash

The player can see the environment and the amount of collected trash which is displayed in the HUD.

2.4.2 Player Properties

Property	Description	Default Value	Effect on Gameplay
Collected levers	Resource for solving puzzles.	None	By correctly placing the lever, the player will be able to solve the puzzle.
Collected trash	The amount of trash the player collected.	0	You can unlock the cage to get a lever to help you solve the puzzle, if u collect all the trash, you will get a secret ending. The look of the environment changes depending on the amount of collected trash.

2.5 Other Game Characters

There will be no other game characters.

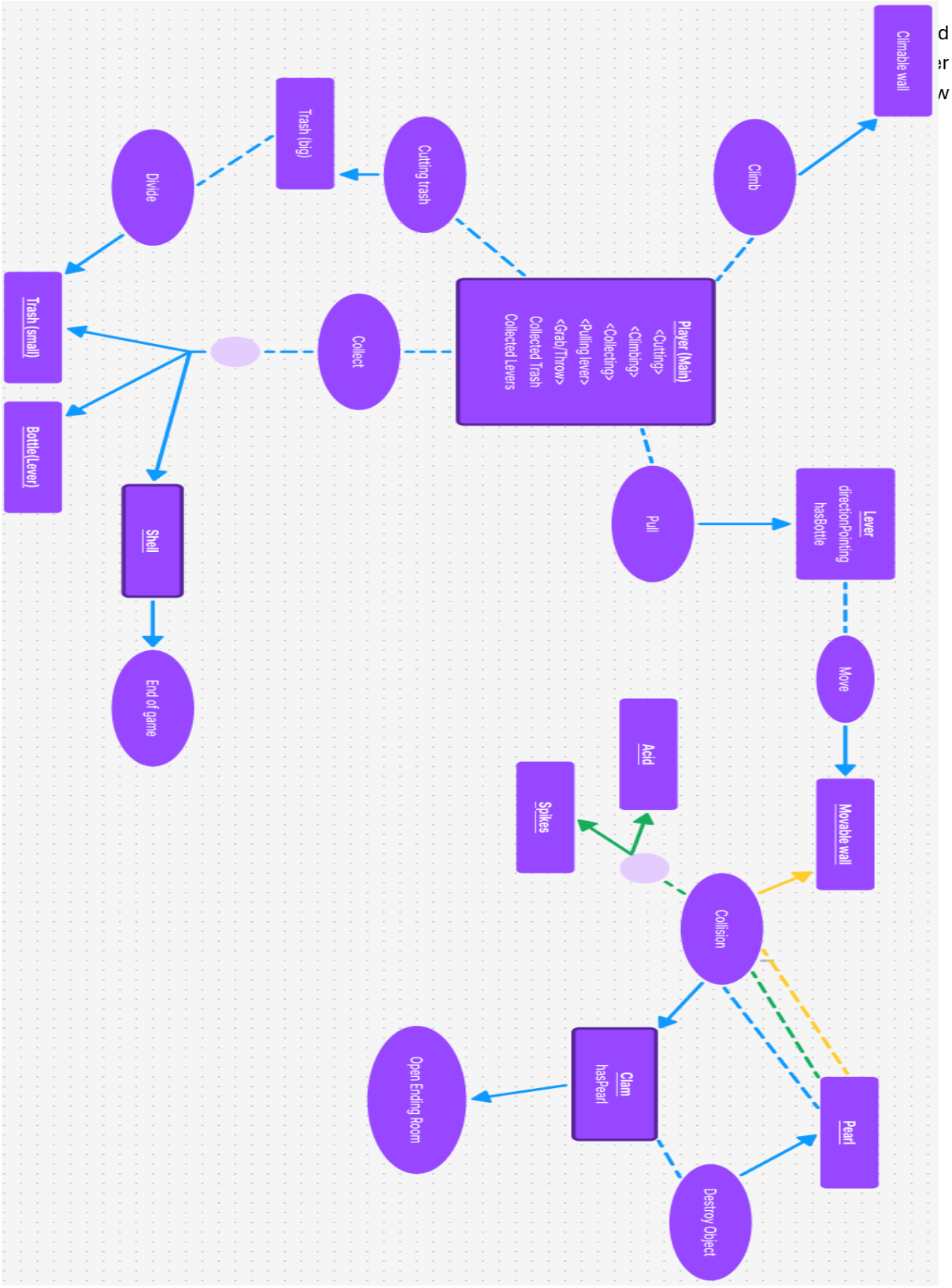
2.6 Gameplay Flow

2.6.1 Definitions

- *Gameplay*
    - Movement
    - Collecting trash and levers
    - Picking up and throwing pearls
    - Exploring the levels
    - Solving puzzles
    - Progressing through the levels
- Player control
    - Moving the crab character
    - Using levers to move walls
    - Grabbing and throwing the pearl

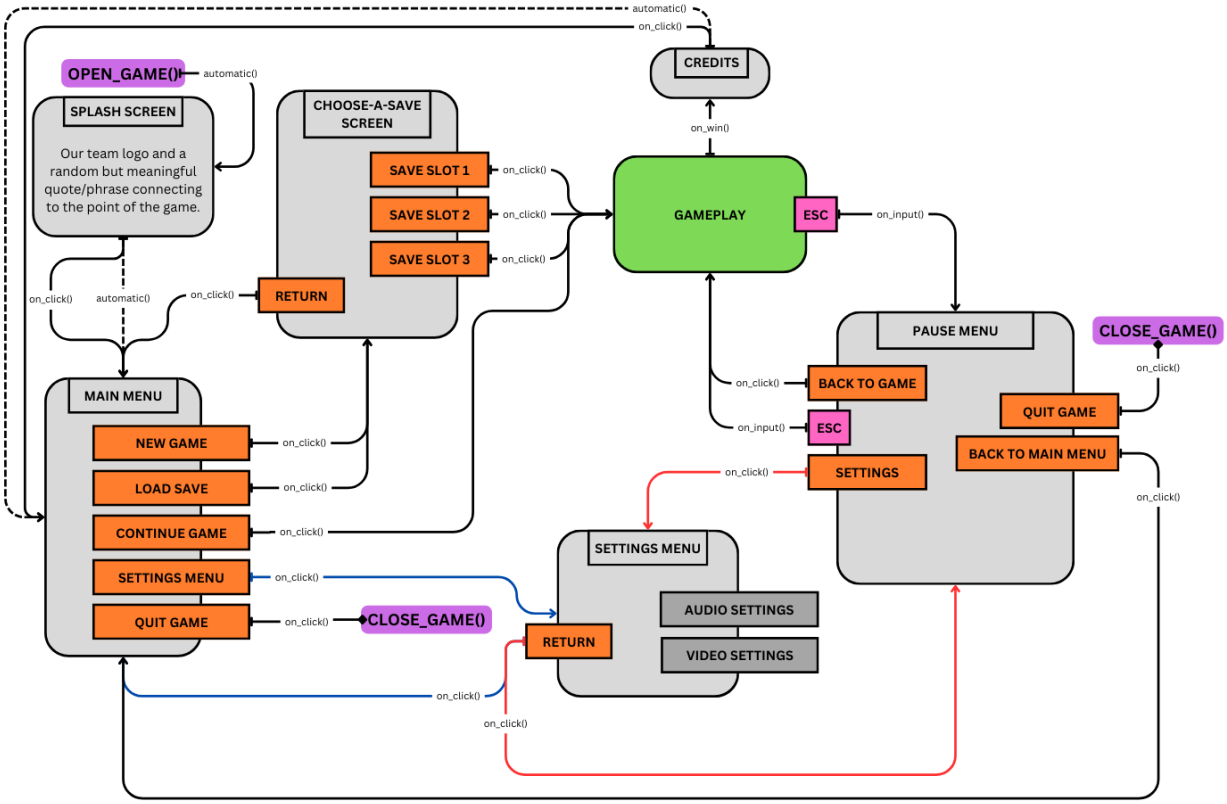
- Game over conditions
    - Victory conditions
        - Collecting a certain amount of trash to progress
        - Solving all the puzzles and moving to the end
        - Collecting the shell
    - Retry
        - If the players fail to “guide” the pearl in the puzzle room, the pearl resets its position to the starting point and the player can try again
    - Conditions for special ending
        - Collecting all the trash in the game

2.6.2 Game Flow Diagram



2.7 Game Architecture

Since the player first interacts with the UI Menus, and they/their functionalities surround the gameplay, the following is the User Interface Menu Diagram:



User Interface Menu Diagram elements description:

- **Splash Screen:** A small animation with our team logo, team name, and a quote dynamically gotten from a random list of quotes inside Unity (think like how Minecraft or Terraria generate their start-menu quotes). The splash screen disappears after a few seconds, but if the player doesn't want to wait for the animation to finish, they can click on the screen to speed up the animation and disappearing of the splash screen.
- **Main Menu:** Menu with an animated background. It has 5 buttons.
    - If there isn't any saved data from previous games, the Continue Game button and Load Save button will be disabled both functionally and visibly.
    - If Continue Game button is enabled and clicked, it takes the player to the gameplay with the most recent save file.
- **Choose-a-Save Screen:** Screen which contains three save slots which can be clicked on, and which displays information about the save slot.
    - If the player gets to this screen by clicking the New Game button, clicking on a save slot will rewrite the save slot data and start the gameplay on that now-empty slot data.
    - If the player gets to this screen by clicking the Load Save button, clicking on a save slot will only be possible if there is an actual save there. When the player clicks on the save slot, they will start the gameplay on the slot's data.
- **Pause Menu:** The player gets to this menu by inputting the PAUSE capture and can leave it in the same way. By quitting or going to the main menu, the player leaves the gameplay, and it's saved to the save slot they're playing on.

- **Settings Menu:** The player can edit the most general settings in this menu, or access one of the submenus to edit more specific settings. When the player clicks return, they return to the screen they got to the Settings Menu from (indicated on the diagram via blue and red arrows)
    - **Audio Settings**
        - Master Volume
        - Music Volume
        - Sound Effect Volume
    - **Video Settings**
        - Set Resolution
        - Fullscreen Toggle

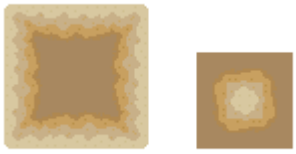
Document Version: 2.0

3 Technical Document

3.1 Visual Content

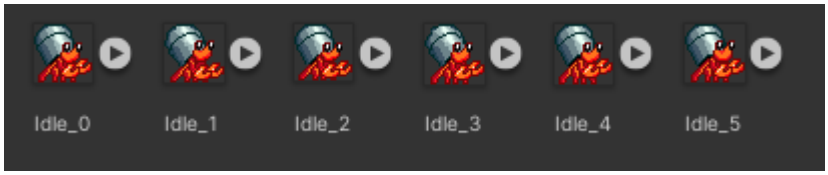
Tile Sets – 24x24px used for TIlemaps

Name	*Count of different tiles*
Beach Sand Tiles	*13*
Beach Water Tiles	*3*
Cave Tiles	*19*
Deep Cave Tiles	*13*
Shallow Water Stone Tiles	*19*
Shallow Water Sand Tiles	*14*
Climbable Planks	*2*
Cave Coral Tiles	*3*
Oil Spill	*3*



Player – 30x30px

Name of Animation	*Number of frames*
Climb	*5*
Climb Idle	*6*
Climb From Ground	*4*
Idle	*6*
Walk	*5*



In this section some of the assets meant for one level appear on others as well, but they are listed for what level they are originally meant to be in.

Beach assets - all the background assets for the Beach

Name of Sprite	*Size (px)*
Beach Rock	*55x35*
Beach Ball	*37x34*
Beach Algae	*46x25*

Beach Sign	*42x34*
Bubble	*59x47*
Shell ground1	*24x24*
Shell ground2	*24x24*
Shell ground3	*24x24*
Beach grass1	*24x24*
Beach grass2	*24x24*
Beach grass3	*24x24*
Beach rock1	*24x24*
Beach rock2	*24x24*
Beach rock3	*24x24*
Beach toy	*24x24*
Palm Tree	*203x294*

Shallow Water assets - all the background assets for the Shallow Water

Name of Sprite	*Size (px)*
Shell ground1	*18x18*
Shell ground2	*20x19*
Star fish ground	*14x16*
Shallow Water rock1	*39x21*
Shallow Water rock2	*10x9*
Shallow Water rock3	*26x15*
Shallow Water algae top1	*19x20*
Shallow Water algae top2	*20x23*
Shallow Water algae top3	*29x44*
Shallow Water algae top4	*18x26*
Shallow Water algae top5	*21x66*
Shallow Water algae ground1	*26x25*
Shallow Water algae ground2	*24x23*
Shallow Water algae ground1	*32x33*
Coral climable1	*20x22*
Coral climable2	*23x22*
Coral climable3	*22x16*
Coral climable4	*28x10*
Coral climable5	*25x22*
Coral climable6	*27x14*
Coral climable7	*36x16*
Coral climable8	*25x18*

Document Version: 2.0

Cave assets - all the background assets for Cave

Name of Sprite	*Size (px)*
Cave Algae1	*36x69*
Coral1	*18x16*
Star Fish	*14x16*
Lever1 rock	*45x19*
Lever2 rock	*41x19*
Lever3 rock	*38x38*
Cave Bottle Cage	*56x62*
Cave algae ground1	*17x15*
Cave algae ground1	*16x21*
Cave rocks top1	*18x14*
Cave rocks top2	*17x26*
Cave rocks top3	*13x19*
Cave rocks ground1	*25x20*
Cave rocks ground2	*29x13*
Fish Bones1	*140x56*
Fish Bones2	*63x29*
Fish Bones3	*30x21*
Fish Bones4	*14x20*
Glowing Coral	*41x44*
Pearl	*19x17*
Cave rocks ground3	*15x7*
Coral2	*53x33*
Coral3	*44x27*
Little clam	*16x10*
Coral4	*44x32*
Coral5	*33x25*
Coral6	*40x27*
Coral7	*37x35*
Cave algae top1	*24x33*
Sea urchin	*35x30*

Document Version: 2.0

Deep Sea assets - all the background assets for Deep Sea

Name of Sprite	*Size (px)*
Arch	*132x137*
Deep Water rock1	*38x19*
Deep Water rock2	*12x11*
Deep Water rock3	*21x11*
Deep Water algae ground1	*25x26*
Rib bones1	*49x46*
Rib bones2	*57X43*
Coral bleached1	*36x28*
Coral bleached2	*27x24*
Coral bleached3	*29x30*
Anchor	*42x54*
Deep Water algae ground2	*14x14*
Skull	*41x33*
Toxic Waste1	*35x13*
Toxic Waste2	*31x14*
Shell Normal	*26x21*
Shell Gold	*26x21*

Trash - listed sprites for collectable and cuttable trash

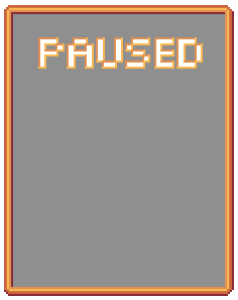
Name of Sprite	*Size (px)*
Net Cuttable	*75x109*
Net Collectable1	*28x35*
Net Collectable2	*32x30*
Net Collectable3	*23x35*

Net Collectable4	*35x33*
Tire Cuttable	*78x98*
Tire Collectable1	*30x26*
Tire Collectable2	*32x34*
Tire Collectable3	*36x33*
Tire Collectable4	*37x39*
Zagreb Trash Bag Cuttable	*74x84*
Zagreb Trash Bag Collectable1	*29x31*
Zagreb Trash Bag Collectable2	*30x33*
Zagreb Trash Bag Collectable3	*57x29*
Castle Cuttable	*72x75*
Castle Collectable1	*26x36*
Castle Collectable2	*26x39*
Castle Collectable3	*23x27*
Trash bag1	*22x24*
Trash bag2	*24x30*
Canister	*25x31*
Bottle1	*21x35*
Glove	*19x35*
Toxic Bottle1	*22x37*
Toxic Canister	*20x30*
Bottle2	*27x29*
Wine Bottle	*22x64*
Spoon	*23x24*
Needle	*17x45*
Red cup	*19x27*

UI elements - listed sprites for UI elements

Name of Sprite	*Sprites Variations*	*Size (px)*
Check	*-*	*4x3*
Check box	*-*	*8x8*
Cursor	*Clicked, Unclicked*	*16x16*
Down	*-*	*5x3*
Drop down	*-*	*168x34*
Drop down bigger	*-*	*168x90*
Load game button	*Clicked, Unclicked, Hover*	*152x39*
Menu button	*Clicked, Unclicked, Hover*	*153x39*
New game button	*Clicked, Unclicked, Hover*	*153x39*
Quit game button	*Clicked, Unclicked, Hover*	*153x39*
Quit button	*Clicked, Unclicked, Hover*	*153x39*
Resume button	*Clicked, Unclicked, Hover*	*153x39*

Settings button	*Clicked, Unclicked, Hover*	*153x39*
Paused menu	*-*	*228x288*
Settings menu	*-*	*215x150*
Slider button	*-*	*8x8*
Slider	*Slided, Unslided*	*54x2*
Title	*-*	*236x73*

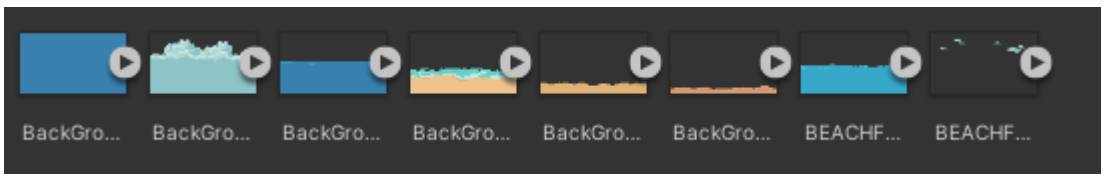


Animated elements - listed sprites for animated elements

Name of Sprite	*Number of frames*	*Size (px)*
UI background	*5*	*320x180*
Turtle	*5*	*480x270*
Algae ground1	*4*	*27x27*
Algae ground2	*4*	*25x29*
Algae ground3	*4*	*26x71*
Clam	*7*	*87x92*

Backgrounds - listed background and it's parallax layers

Level of background	*Number of parallax layers*	*Size (px)*
Beach	*8*	*480x270*
Cave	*5*	*1159x652*
Deep Sea	*5*	*480x270*
Shallow water	*8*	*480x270*



3.2 Audio Content

File Name	File Format Type	Audio Type	Looped	Compressed
Beach_Theme1	mp3	background music	yes	
Beach_Theme2	mp3	background music	yes	
Level1_Theme1	mp3	background music	yes	
Level1_Theme2	mp3	background music	yes	
Cave1_Theme1	mp3	background music	yes	
Cave1_Theme2	mp3	background music	yes	
DeepWater_Theme1	mp3	background music	yes	
DeepWater_Theme2	mp3	background music	yes	
Menu_Theme	mp3	background music	yes	
Menu_Click	mp3	UI sound	no	
Seagull_Sound	mp3	ambient sound	no	
Whale_Sound	mp3	ambient sound	no	
Walk	mp3	character audio	yes	
Climb	mp3	character audio	yes	
Jump	mp3	character audio	no	
Jump_Into_Water	mp3	character audio	no	
Trash_Pickup1	mp3	character audio	no	
Trash_Pickup2	mp3	character audio	no	
Pearl_Pickup	mp3	character audio	no	
Pearl_Throw	mp3	character audio	no	
Pearl_Roll	mp3	SFX audio	yes	
Trash_Cut	wav	character audio	yes	
Lever_Interact	mp3	SFX audio	no	
Wall_Movement	mp3	SFX audio	no	
Puzzle_Solve	mp3	SFX audio	no	
Cage_Open	mp3	SFX audio	no	
Clam_Close	mp3	SFX audio	no	

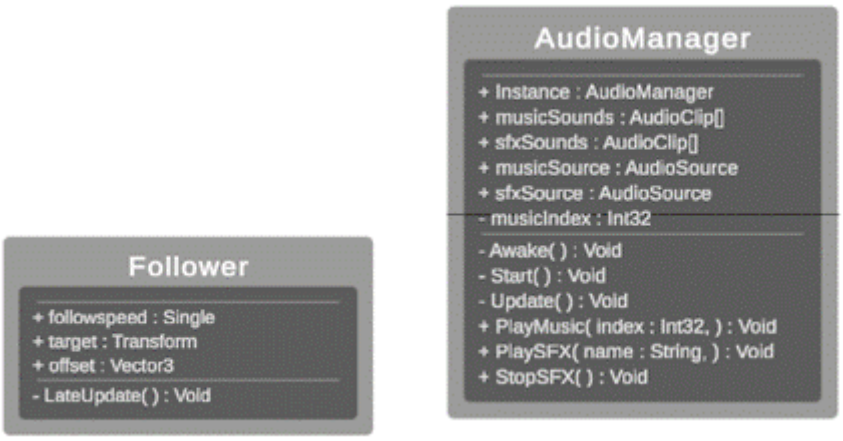
3.3 Programming Content

The game is made using the Unity game engine version 2022.3.28f1 and therefore, the programing language used is C#. The code editors which were primarily used are Visual Studio Code and Visual Studio. Since the game is made with Unity, there are certain additions to C#, such as the base class MonoBehaviour, that are implemented in Unity and allow the code to interact with parts of the game more easily. The game was made and tested on a modern Windows operating system, but most, if not all, of the functionalities should be available on other operating systems with the help of Unity. Github and Git were used for version control and shared access to code during development. No special error handling systems were used, other than the ones that are a part of Unity.

3.4 Code Structure

3.4.1 UML Diagram

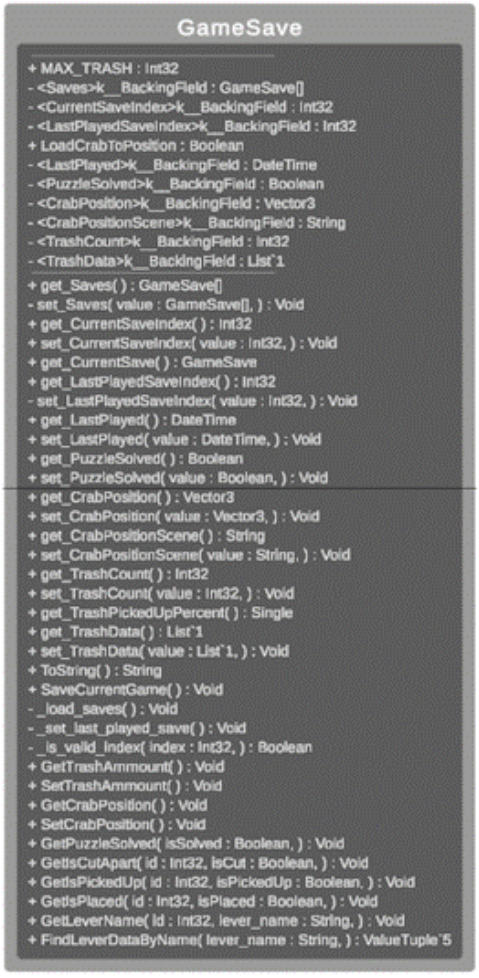
AudioManager is the main script handling game music and sfx. The Follower script is used for camera management.



Document Version: 2.0

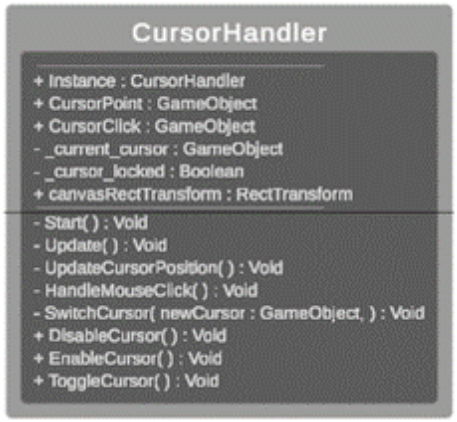
This is the structure of all character based scripts that are used for the player character mechanics.

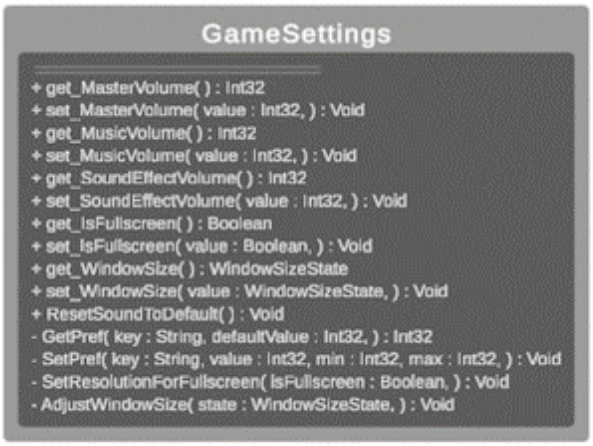
These 2 classes are used for the game save management.

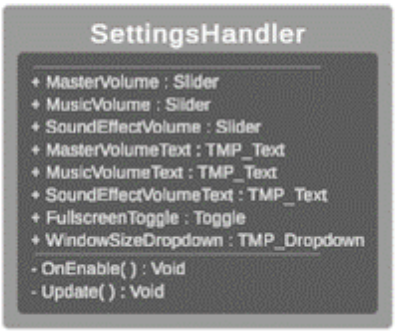


Document Version: 2.0

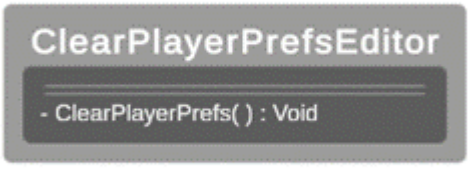
These are all the scripts used for UI handling and customizing settings.

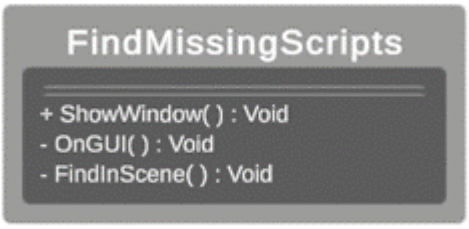




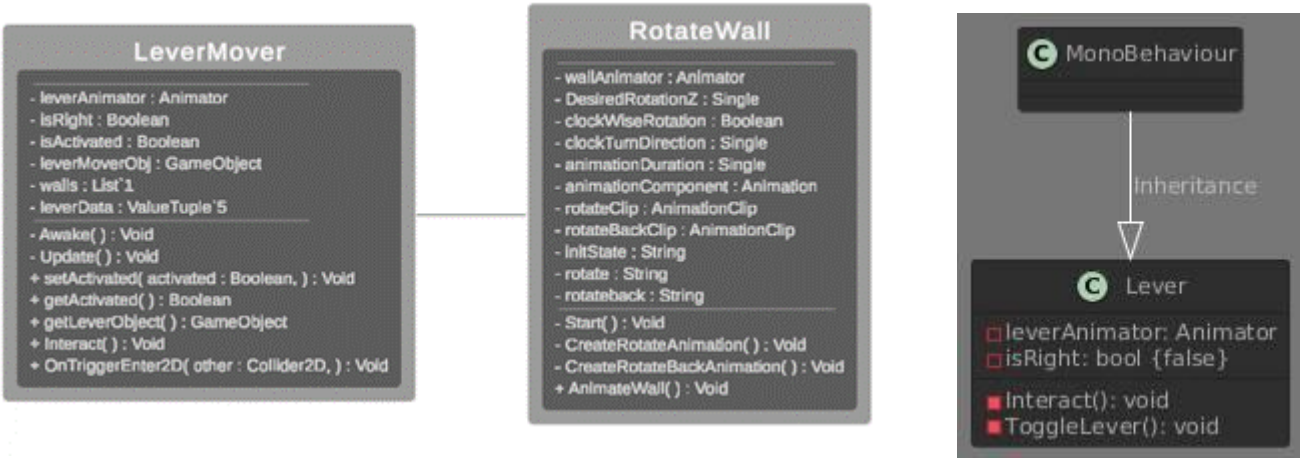


These 2 methods are used for debugging associated with the game save management.

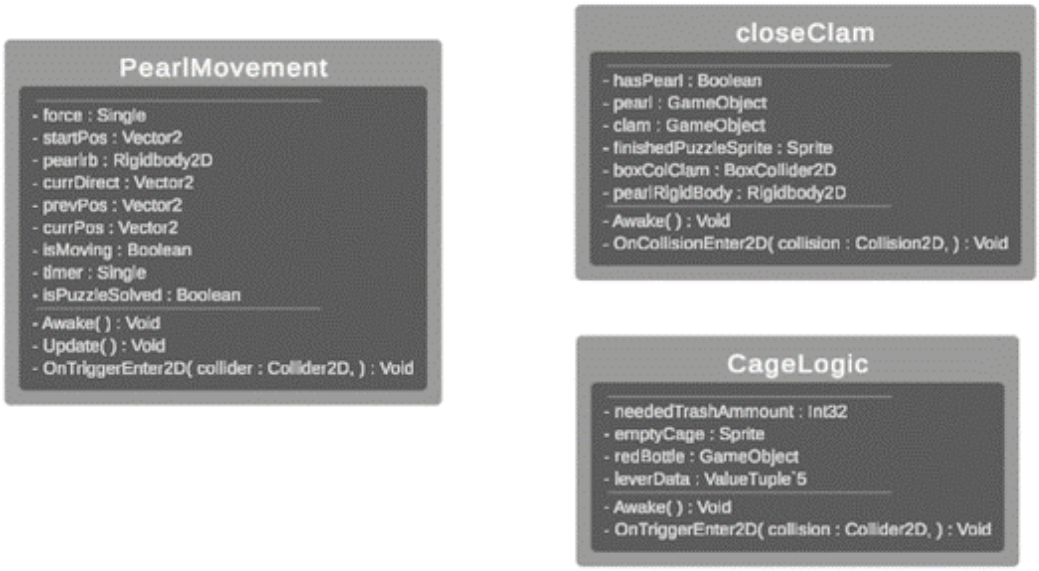




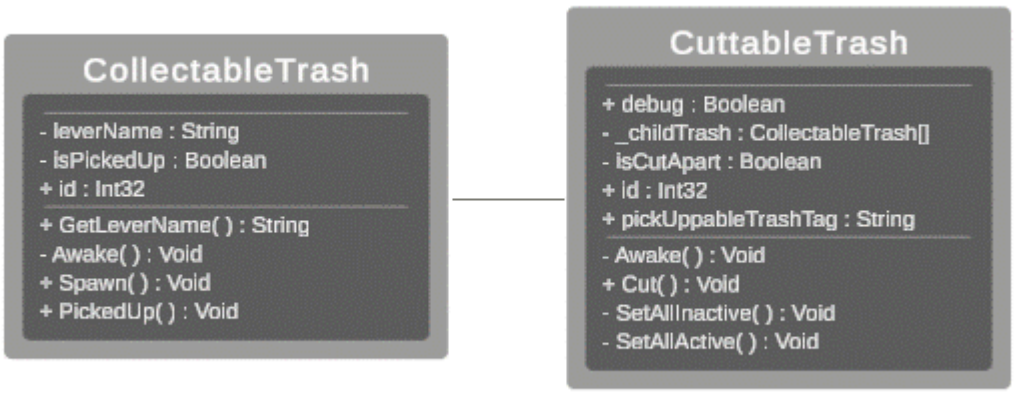
These scripts are used for handling the lever gameobject and the wall gameobject.



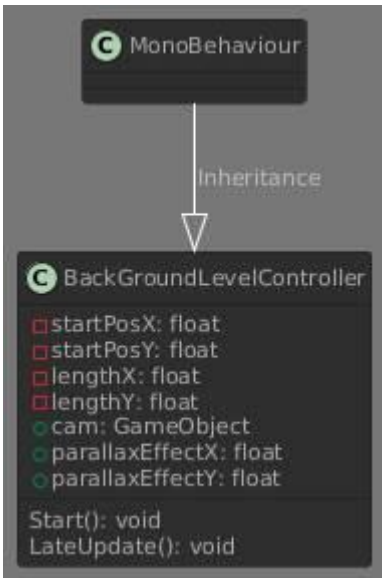
These scripts are used for the puzzle solving part of the scene Cave 1.



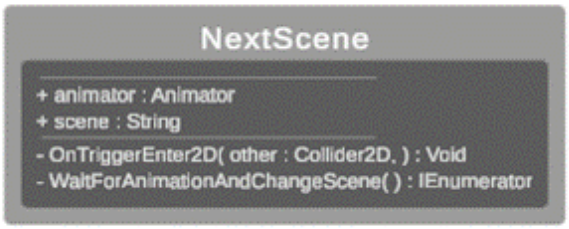
These are the scripts for the trash gameobjects.

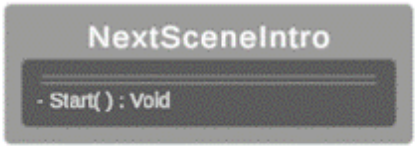


These are the scripts used for handling the background parallax of the scenes.



These are the scripts for scene management.





3.4.2 Additional Descriptions

Table of key classes and methods.

Class Name	Method Name	Purpose	Order of Operations
characterMovement	checkCollision	Sets the onGround and onWall variables for later used.	Called on each update.
	setGravity	Sets the character's gravity based on the current gravMultiplayer. Changes during jumping and climbing.	Called at different parts of the jump, when landing and climbing.
	updateHorizontal	Reads the horizontal input.	Called on each update.
	getHorizontal	Sets the horizontal vector.	Called after updateHorizontal.
	Update	Decides what the next action should be depending on the inputs and location of the character.	Called on each update.
	FixedUpdate	Changes the characters position by updating its gravity and movement vectors based on the Update function.	Called on each game tick, not synchronized with the Update function.
	calculateGravity	Sets gravMultiplier based on character status.	Called during FixedUpdate.
	Jump	Used to make the character jump. Sets and resets additional status variables and counters.	Called during FixedUpdate.
	StrongWallJump	Has the same function as Jump, but for when the character is on a wall.	Called during FixedUpdate.
	moveWall	Moves the character vertically along a wall.	Called duing FixedUpdate.
	moveHorizontal	Moves the character horizontally when on the ground or in the air.	Called during FixedUpdate.
CharacterTrashHandler	Update	Reads input and tries to pick up or cut trash.	Called on each update.
	FixedUpdate	Calls functions to pick up trash or activate cutting trash on objects.	Called on each game tick, not synchronized with the Update function.
GrabObjects	Update	Handles player input and grabs and releases objects as well as activates levers.	Called on each update.

	HandleMovement	Checks direction and calls FlipDirection	Called on each update.
	FlipDirection	Changes direction of held object.	Called on each update.
	HandleGrabbableObject	Used to grab object.	Called when attempting to grab an object.
	HandleLever	Used to pull lever. Calles it's interaction function.	Called when attemping to pull lever.
	ReleaseGrabbedObject	Used to stop holding grabbed object.	Called when attempting to release object.
TrashCut	Cut	Calls cut on closest object.	Called from CharacterTrashHandler.
	OnTriggerEnter2D	Adds collided with object to list of cuttable objects.	Called when a Trigger Enter event happens.
	OnTriggerExit2D	Removes collided with object from list of cuttable objects.	Called when a Trigger Exit event happens.
	FindClosest	Finds closest object, which will be cut next.	Called when cutting.
TrashPickup	OnDestroy	Saves trash amount to current save.	Called when exiting the scene.
	Collect	Calls PickedUp on collected object.	Called from CharacterTrashHandler.
	OnTriggerEnter2D	Adds collided with object to list of collectable objects.	Called when a Trigger Enter event happens.
	OnTriggerExit2D	Removes collided with object from list of collectable objects.	Called when a Trigger Exit event happens.
	FindClosest	Finds closest object, which will be collected next.	Called when collecting.
CuttableTrash	Awake	Deactivates this object if its cut in save file, else calls SetAllInactive.	Called when spawning object.
	Cut	Deactivates this object, calls SetAllActive and sets its state to the current save.	Called when cutting in TrashCut.
	SetAllInactive	Sets all child objects to inactive.	Called during Awake.
	SetAllActive	Sets all child objects to active.	Called when cutting this object.
CollectableTrash	Awake	Deactivates this object if already picked up in save file.	Called when spawning object.
	PickedUp	Deactivates this object and saves its state to the save file.	Called from TrashPickup.

GameSave	GameSave	Loads a GameSave object from a string.	Called when loading a save from the menu.
	GameSave	Creates a new GameSave object with the default values.	Called when creating new game from the menu.
	ToString	Creates string which holds the current game state.	Called when saving the game.
	SaveCurrentGame	Saves current game to PlayerPrefs.	Called when saving the game from the menu.
	_load_saves	Loads all saves from PlayerPrefs.	Called when interacting with save states in the menu.
	_set_last_played_save	Sets the last played save from the Saves array.	Called on game start.
	GetTrashAmmount	Reads the current trash amount from the TrashPickup object.	Called when saving the game.
	SetTrashAmmount	Sets the current trash amount from the TrashPickup object.	Called when exiting the scene.
	GetCrabPosition	Reads the characters current position.	Called when saving the game.
	SetCrabPosition	Sets the characters current position.	Called when loading save file.
	GetPuzzleSolved	Used for clam and pearl communication.	Called from closeClam.
	GetIsCutApart	Used for cuttable trash communication.	Called from cuttableTrash and CageLogic.
	GetIsPickedUp	Used for collectable trash communication.	Called from CollectableTrash.
	GetIsPlaced	Used for getting information of levers being placed down.	Called from LeverMover.
	GetLeverName	Used for storing the levers name.	Called from CollectableTrash and TrashPickup.
	FindLeverDataByName	Used for find data of a specific lever.	Called from LeverMover and CageLogic.
GameSaveEditorWindow	OnGUI	Displays available saves and their information.	Called when drawing GUI.
	RefreshSaves	Refreshes the displayed saves.	Called from OnGUI.
LeverMover	Awake	Loads and displays lever state from CurrentSave.	Called when spawning the object.
	Interact	Animates lever and connected walls.	Called from GrabObjects when interacting with lever.

	OnTriggerEnter2D	Enables lever when the character brings it the moving part.	Called when a Trigger Enter event happens.
RotateWall	Start	Sets up animations by calling CreateRotateAnimation and CreateRotateBackAnimation.	Called when creating object.
	CreateRotateAnimation	Creates basic rotation animation.	Called from Start.
	CreateRotateBackAnimation	Creates basic rotation animation for reversing animtaion.	Called from Start.
	AnimateWall	Animates the wall depending on current state.	Called from LeverMover.
CageLogic	Awake	Sets cage state from CurrentSave.	Called when loading the game.
	OnTriggerEnter2D	Activates a bottle object.	Called when a Trigger Enter event happens.
closeClam	Awake	Sets up clam from CurrentSave.	Called when spawning the object.
	OnCollisionEnter2D	Detects collision with pearl, closes clam and updates CurrentSave.	Called when a Collision Enter event happens.
PearlMovement	Awake	Load and set pearl information from CurrentSave.	Called when spawning the object.
	Update	Handles the pearls movement.	Called on each update.
	OnTriggerEnter2D	Changes direction, stops the pearl from moving or resets the pearl depending on the collided with trigger.	Called when a Trigger Enter event happens.

The game uses direct method calls as a way of communication between classes and their objects. Many of the functions are triggered by game events such as OnTriggerEnter2D being called when something enters a collider which is set to be used as a trigger. Inputs are managed by the Input Manager and are read via their appropriate functions in code.