**Game Concept & Design Document**

**The Return Of Laika**

**A Series Of Pawful Events**

Contents

[**1**](#_3tbugp1) **TEMPLATE INTRODUCTION 2**

[**2**](#_28h4qwu) **CONCEPT DOCUMENT 3**

[**2.1**](#_nmf14n) **Title Page 3**

[**2.2**](#_37m2jsg) **Introduction 3**

[**2.3**](#_1mrcu09) **Game Analysis 3**

[**2.4**](#_46r0co2) **Game Atmosphere 4**

[**2.5**](#_2lwamvv) **Gameplay 5**

[**2.6**](#_111kx3o) **Key Features 6**

[**2.7**](#_3l18frh) **Selling Features 6**

[**3**](#_17dp8vu) **DESIGN DOCUMENT 7**

[**3.1**](#_206ipza) **Game Design Definitions 7**

[**3.2**](#_4k668n3) **Player Elements 7**

[3.2.1](#_2zbgiuw) Player Definition 7

[3.2.2](#_1egqt2p) Player Properties 7

[3.2.3](#_3ygebqi) Player Rewards (Power-ups & Pick-ups) 7

[3.2.4](#_2dlolyb) User Interface (UI) 8

[3.2.5](#_sqyw64) Player View 8

[**3.3**](#_z337ya) **Characters 8**

[3.3.1](#_3cqmetx) Character Definitions 8

[3.3.2](#_1rvwp1q) Character List and Properties 8

[**3.4**](#_2r0uhxc) **Game Flow Chart 9**

[**3.5**](#_1664s55) **The Story 9**

[**3.6**](#_3q5sasy) **Concept Art 9**

[**3.7**](#_25b2l0r) **Level Design 9**

[**3.8**](#_kgcv8k) **Audio & Sound F/X 9**

[**3.9**](#_34g0dwd) **Game Architecture 10**

[**4**](#_1jlao46) **TECHNICAL DOCUMENT 11**

[**4.1**](#_43ky6rz) **Visual Content 11**

[**4.2**](#_2iq8gzs) **Audio Content 11**

[**4.3**](#_xvir7l) **Programming Content 12**

[**4.4**](#_3hv69ve) **Code Structure 12**

[**5**](#_1x0gk37) **APPENDIX A 13**

[**6**](#_2u6wntf) **APPENDIX B 15**

[**7**](#_19c6y18) **APPENDIX C 16**

1. **Concept Document**
   1. **Title Page**

The Return of Laika, A Series of Pawful Events, Game Concept by *One of the teams from Ordi*, Version 1.0

* 1. **Introduction**

The following document specifies gameplay design of a 2D pixel art adventure from a top-down perspective. The main goal of The Return of Laika is to produce a simple yet entertaining set of mechanical puzzles with elements of action followed by educational and humorous storytelling.  
  
The game is set in 1957, somewhere in outer space, where the player is supposed to control movement, shooting and gripping mechanics of Laika the dog, whose main task is to return home safely.

* 1. **Game Analysis**

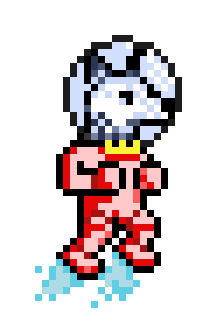
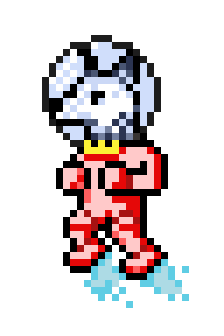
This is a general overview of the game.

| **Game Description** |  |
| --- | --- |
| Genre: | * Adventure with elements of comedy |
| Game Elements: | * Dashing * Collecting * Dodging * Shooting * Obstacles * Seeking |
| Game Content: | * Humor |
| Theme: | * Sci-Fi |
| Style: | * Pixel art |
| Game Sequence: | * Linear |
| Player: | * 1 |
| **Game Reference** |  |
| Game Taxonomy: | * Play   + Story     - Fiction |
| Player Immersion: | * Tactical * Narrative * Emotional |
| Reference: | * Story of Laika (based on true story) * A Series of Unfortunate Events * Scooby Doo * Undertale |
| **Game Technical** |  |
| Technical From: | * 2D graphics |
| View: | * Top down |
| Platform: | * Unity (C#) |
| Device: | * PC |
| **Game Sales** |  |
| Consumer Group: | * Children and adults |
| Payment: | * Free |
| Estimated Price: | * Free |

* 1. **Game Atmosphere**

The Return of Laika’s key atmospheric element is tranquility achieved by soothing weightless floating caused by outer space gravity. Although, in the vastness of empty space, the player can be faced with moments of danger caused by cosmic phenomena such as meteor showers, black holes, and burning planets.

Music has a big impact on the game atmosphere. Slow ambient melodies can be heard at times when the player is feeling safe, while unsettling and dramatic ones indicate that danger is near.



*Early sketches of the main character Laika.*



*Simple empty background prototype which represents space.*

* 1. **Gameplay**

After launching the game, the screen displays the main menu with the game’s logo and only a few standard options such as *New Game*, *Continue*, *Options* and *Exit*. The background of the main menu features the starry vastness of outer space with floating elements of cosmic phenomena. The options menu offers only toggling music and sound effects on and off.



*The game’s main menu.*

When *New Game* is selected, the story begins and it is presented in the form of an animated cut scene: A spaceship, with Laika and her monologues in it, travels through space and, while approaching planet Earth, suddenly breaks down and is afterwards blown to pieces.

The animations are accompanied by Laika’s humorous monologues, which are displayed in a dialogue box at the bottom of the screen. After the ship is blown up, the camera focuses on the parts of the ship that scattered across space, and Laika realizes that she must collect all those parts to repair her ship in order to continue the journey.

The player can move using W, A, S, and D keys. Such movement is very slow and inefficient, but, fortunately, Laika gets an opportunity to collect some residue of her rocket engine’s thrust power. That residue has a power-up function and allows the player to move using thrust that is activated by holding the space bar key.

The goal of the entire first level is to teach movement patterns that the player will use while avoiding dangers such as black holes, burning planets and meteor showers. If Laika gets hit by a meteor or gets too close to a burning planet, her health is reduced. If her health reaches 0, the game is over and Laika gets spawned at the beginning of the level. However, while exploring the space, Laika can also come across blue planets, which are pushable. These planets can have a dog cookie hidden behind them, and eating it gives Laika one health point. If her health is full, the cookie is not eaten. If Laika gets sucked into a black hole, though, the game progress is reset and Laika spawns back where she started.



*A cookie hidden behind a blue planet. Laika can eat it if her health is below maximum to replenish 1 point.*



*Otherwise, Laika notifies the player that she is not hungry.*

By avoiding the mentioned elements, the player should collect all parts of the ship that are on certain parts of the map. Once a part is collected, the weight of the part slows down the player, and it is the player’s duty to take the part back to the wreck of the ship that continues to float in space. While carrying a ship part, the player can also protect themselves with it from meteors in meteor showers.



*Laika carrying a ship part.*

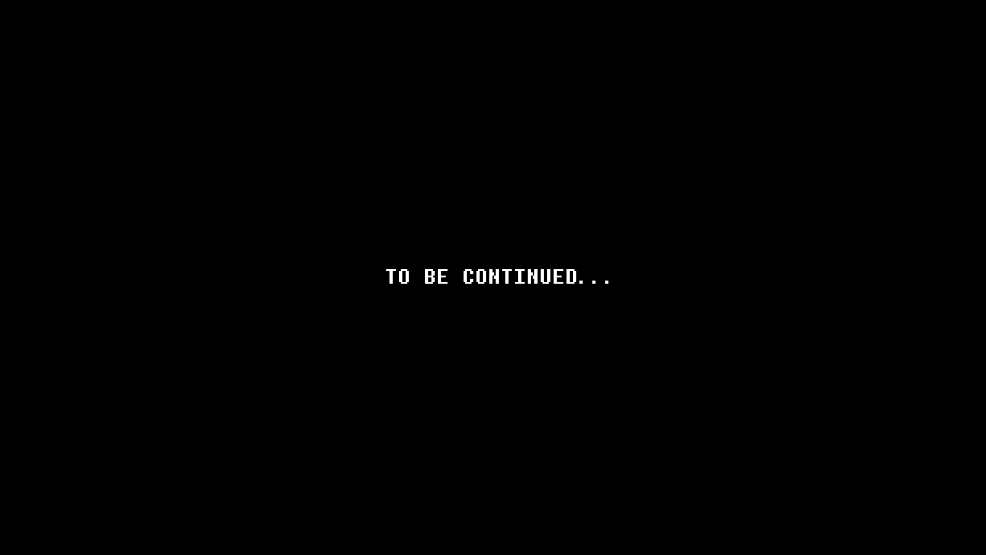
Once all the parts have been collected, the ship regains its function. After a while, a new accident occurs. Laika and her ship are attacked by extra-terrestrials. The player's duty is to use the gun she finds in her restored ship to help Laika defend her ship from extra-terrestrials and keep moving. New accidents can be added subsequently.



*Laika finds her gun after fixing her ship.*

The mentioned game mechanics and player controls are balanced between simple and challenging. There are no special game modes. The game is completed once the player overcomes all events that have befallen Laika and her spaceship. The number of accidents represents the number of levels.

The Return of Laika is meant to be beautiful as a simple pixel art adventure reinforced by funny monologues and simple, yet enjoyable mechanics. These are the points which are supposed to make the game fun.



*Ending screen of the game prototype.*

**Key Features**

Key features are a list of game elements that are attractive to the player.

* Animated 2D pixel art
* Thoughts of the main character expressed through monologues
* A reference to the actual (real life) story of the main character
* Calming audio elements
* The mechanics of thrust motion

* 1. **Selling Features**

This is a list of features that could be potentially helpful to marketing.

* Kickstarter campaign
* If someone backs the project on Kickstarter, their name will be in the credits (special thanks) at the end of the game
* Merchandising (such as T-shirts and mugs)
* Finding the right publisher



EVERYONE

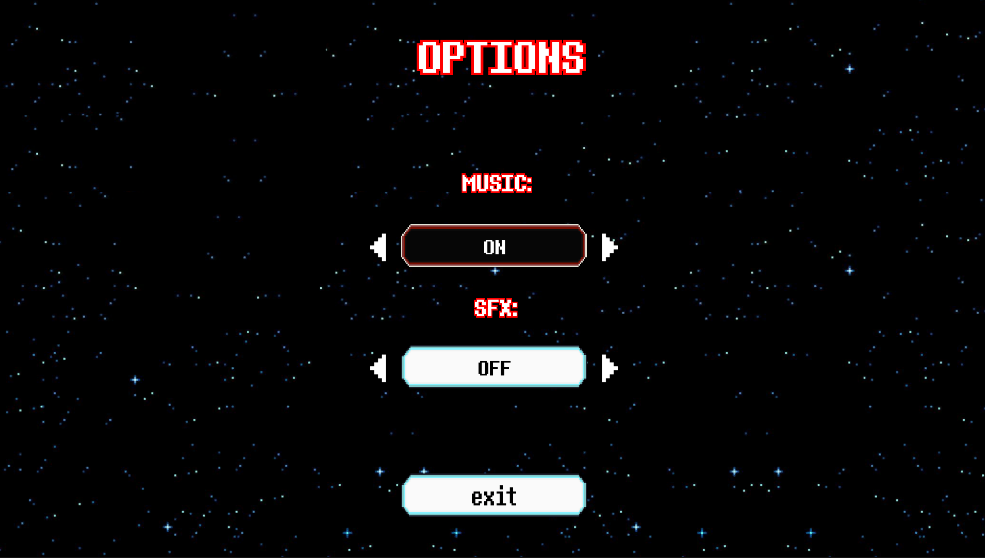
Titles rated E (Everyone) have content that may be suitable for persons ages 6 and older. Titles in this category may contain minimal cartoon, fantasy or mild violence and/or infrequent use of mild language.

1. **Design Document**

This part of the document describes how game objects behave, how are they controlled and what properties they have. This documentation is primarily concerned with the game itself.

* 1. **Game Design Definitions**

The main menu of the game has four buttons: *New Game*, *Continue*, *Options*, and *Exit*. The options menu allows the toggling of music or sound effects between off and on. Along with those, the menu includes the logo of the game, which is the game’s stylised title, and a simple background, both of which match the atmosphere with their design.

**

*The options menu with toggles for background music and sound effects.*

When *New Game* is pressed, a skippable cutscene begins. It serves as an introduction to the story and precedes the first level. The cutscene is filled with Laika’s humorous monologue as she travels through space in her spaceship, but it ends with an accident - Laika’s ship falls apart into several pieces. The first level begins soon afterwards at the same location where the cutscene ends, and is marked by another short cutscene whose goal is to show the locations of the scattered ship parts and hint that the player is supposed to gather them.

Laika’s first goal is to gather residual thrust power that enables faster movement through space. Without it, her movement, achieved with the keys W, A, S, and D, is extremely slow. After the device is picked up, W, A, S, and D are still used for movement in their respective directions, but now the player can press the Space key while holding any of the four movement keys to activate the usage of her thrust power. The movement through space is restricted by invisible borders and a warning message is displayed that it’s dangerous to go further.



*Laika discovers some thrust power residue that can help her move faster.*

The ship parts that have to be gathered are all in different dangerous or hidden areas - close to black holes, between burning planets, or hidden behind planets. Laika can interact with those uncovered ship parts with the help of grabbing mechanics so she can carry them back to her ship. The ship parts she gathers slow her down depending on the ship part’s mass, which in turn makes the movement and avoidance of danger harder. However, if Laika is carrying a ship part when a meteor shower starts, it is raised up and used as a shield.



*Meteor shower while Laika is carrying a ship part.*

*Game over* can happen in two different ways:

1. Laika has a health system displayed in the upper left corner of the scene. It counts eight lives, and she loses one or more in the following cases:
   * 1. If Laika stays within the glowing rim of a burning planet, she loses a life every 2 seconds.
     2. If she is hit by a meteor during a meteor shower, she loses a life. These meteor showers appear at random and move diagonally across the scene from the upper border of the scene.
     3. If her number of lives reaches 0, the player is spawned at the beginning of the game - not the level where they died. The player is also given an explanation that, fortunately, there was an anomaly in time and space which saved Laika from dying, but returned the space to the state it was in just before Laika started gathering the ship parts.
2. If Laika is sucked in by a black hole, the level is reloaded without saving the player’s progress.

If Laika approaches the shipwreck while carrying a ship part, she can assemble the current ship part onto the wreck and rebuild her ship part by part that way. The goal of the level is to gather all the scattered parts and assemble them onto the wreck to be able to get the ship back up and running. After that is completed, an end cutscene plays and the next level begins.

* 1. **Player Elements**
     1. **Player Definition**

The player can control Laika’s movement in two ways: with just the keys W, A, S, and D for slow movement, or the keys W, A, S, and D combined with the Space key for faster movement with thrust power. The player can also protect themselves from the danger of meteors by using a carried ship part as a shield, which is automatically placed in a defensive position over the player. Moreover, the player can gather the ship parts in an arbitrary order - any route or path through objects in space is available as long as the player is within the invisible borders on the edges of the map which prevent the player from leaving the available area and warn him of the impossibility.

The information about the flow of the story is visible through the skippable cutscene before the first level or from Laika’s occasional comments that appear randomly throughout the duration of the level. It is assumed that the player can intuitively understand the mechanics needed to reach the goal by using combinations of input or through hints displayed above Laika’s head, for example, when interaction with a specific object is possible and in which way. Additionally, the player can anticipate the danger of a meteor shower with the help of sound information - the intensity of the background music changes to announce the incoming danger.

The default settings include sound effects and music turned on, which can be turned off through the game’s options menu. However, the player cannot change the controls used for the gameplay mechanics.

At the beginning of each level, the player has all lives.

* + 1. **Player Properties**

Player properties are defined with the following list:

* Health – At the beginning of each level the health bar is full. It consists of eight lives, which are lost through interactions with dangers such as meteor showers or burning planets. How many are lost depends on the intensity of the interaction, for example, interaction with a black hole instantly reduces the lives to zero. The loss of all lives causes the loss of all progress in the level and restarts it.
* Thrust power – If Laika does not posses the thrust power, movement through space is extremely slow and makes the dangerous situations practically impossible to pass through. Collecting the thrust power enables faster, though more challenging movement and the player is expected to learn to control the thrust force, which is regulated by the duration of the Space key being held down.
* Shield – If Laika is currently carrying a ship part, it is automatically set to be used as a shield in meteor showers. Some of the meteors are destroyed upon impact with the shield and no longer present a danger to Laika. In case Laika is not carrying any ship part during a meteor shower, every meteor presents a danger to losing a life if it hits Laika and it is therefore necessary to find a ship part as a shield beforehand or maneuver to successfully avoid the meteors.
* Grabbing, carrying, and releasing ship parts – Laika has grabbing mechanics available which allow her to collect a ship part. She can carry the grabbed ship part with her, which slows her down depending on the part’s mass, assemble the part onto the wreck, or let it go by pressing the same key used for grabbing it and continue moving without it.
* Weapon – At the end of the first level, Laika finds some sort of gun near her spaceship. The gun can be used for combat during level two, however combat is not implemented.
* Moving planets – Laika can push away some planets in order to find cookies or ship parts that can be hidden behind them. Pushing the planets is enables by pressing a certain key when Laika is close enough to the planet, and the player is notified of the possibility of the interaction with a message above Laika’s head.
* Dog cookie – If Laika’s health is below the maximum and she collects a bone-shaped cookie, which can sometimes be found behind planets, she recovers one health point. Nothing happens if her health is full, the cookie remains where it was and can be collected later.

* + 1. **Player Rewards (Power-ups & Pick-ups)**

An item that can be considered a power-up is the thrust power residue as it enables faster movement through space. However, without it, Laika’s survival in space would not even be possible so this item is essential and unavoidable.

Another power-up would be the cookies that the player can occasionally find behind planets, and each cookie restores one life. At the end of the first level, Laika finds her gun which could be used for the planned combat in level two, which is not implemented.

What can be considered a reward for collecting a ship part is the fact that the collected part can be used as protection during meteor showers.

* + 1. **User Interface (UI)**

The user interface consists of controls that allow the player to use the game mechanics for movement, grabbing and assembling ship parts, pushing planets away, and using the shield.

Main Menu user interface:

Navigation through the menu’s buttons is done with the mouse or keyboard. With the mouse, the pointer combined with a left mouse button click is used to select a button. With a keyboard, Up Arrow and Down Arrow keys are used to focus the buttons, and then Enter is used to press the focused button.

Gameplay user interface:

The following keys are used for movement: W for upwards, A for leftwards, D for rightwards, and S for downwards movement. A combination of one of those keys with the Space key activates the thrust device if the player has collected it and moves the player faster in the chosen direction.

Grabbing and releasing ship parts - key E.

Assembling ship parts onto the wreck (requirement - the player is carrying a ship part) - key X.

Moving planets away – key P**.**

Shooting the space gun – key R**.**

Positioning the shield (requirement - the player is carrying a ship part during a meteor shower) – right arrow key and left arrow key move the shield to the right and left side of the player, respectively**.**

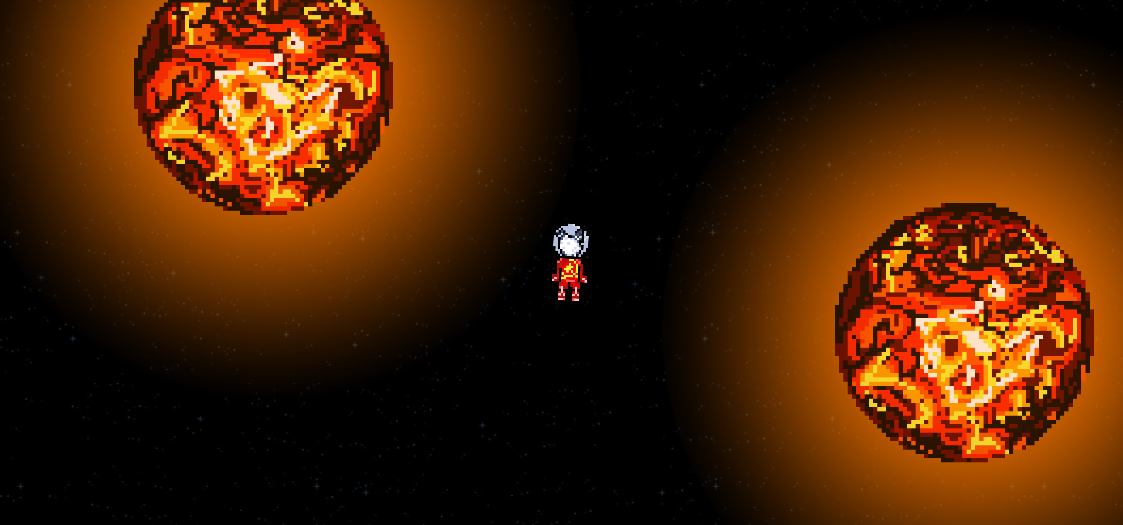
Skipping a cutscene (requirement - marked as skippable near the upper border of the scene) - Escape key.

Information about actions, i.e. keys that can be pressed to achieve something is presented to the player with texts displayed above Laika’s head at the moment when such actions are possible.

* + 1. **Player View**

The game has a top-down orientation, and the camera is always positioned above Laika’s head and follows the player character’s movement across the map.

During the cutscenes, the camera is animated to show objects of interest and the player can move neither the character nor the camera.

*Laika between two dangerous burning planets.*

*Example of Laika’s monologue.*

* 1. **Characters**
     1. **Character Definitions**

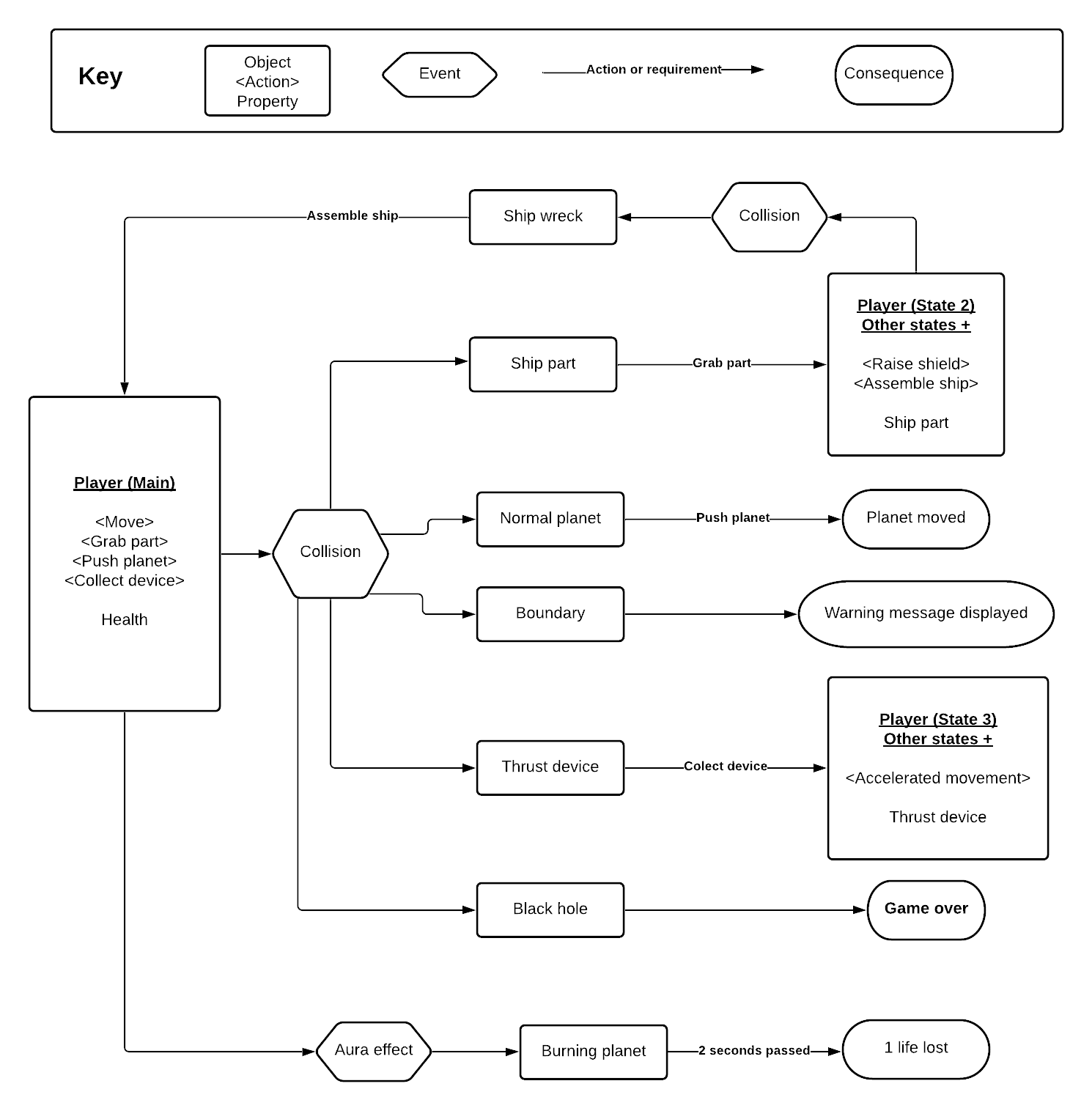
Laika is the protagonist of the game. There are no characters besides her in the first level.

* + 1. **Character List and Properties**

This is a list of all the characters and their properties:

| **Object** | **Properties** | |  |  |
| --- | --- | --- | --- | --- |
| *Unit type* | *Hit Points* | *Damage* | *Bullet speed* | *Movement* |
| *Protagonist* |  |  |  |  |
| Laika | 8 | - | Medium | Slow/Medium |

* 1. **Game Flow Chart**



* 1. **The Story**



*Opening frame of the game’s introductory cutscene.*

The story begins with the camera showing the vast open space and the current date is displayed over it - the date Laika launched into space. A spaceship slowly appears in the scene and Laika’s monologue begins:

“It feels like ten pawfuls of sunrises have come and gone already. When am I going to start returning home?”

“If only Albina or Mushka were here like when we trained! My tail is getting restless. I can't wait to play with them again, and run, run a lot!”

“And this weird gel food, yuck! I hope I never have to taste it again when I get back! Oh, I could do with some nice chicken. Or a treat.”

…

At one moment, Laika comments on the rising heat inside her ship by saying: “It's getting kind of hot, I could rip all my fur out! But that would hurt…”. After that, the ship starts shaking, and Laika shows interest in the strange sound her ship is now making, noting that it does not remind her of her training. Once the ship starts shaking, she expresses her distress by the movement, starting to guess that the ship might explode and almost asking a cut-off question which is meant to be “Am I going to be okay?”.

Fortunately, Laika does survive the explosion of her spaceship, but she is thrown outside and parts of the ship are torn off and scattered off into space, the locations displayed with the camera.

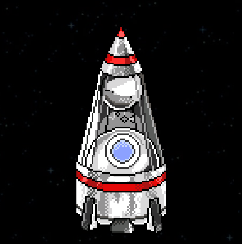
Laika is left at the wreck of her ship, realising through monologue that she has to try and fix her ship if she wants to go home, quickly coming to the conclusion that she also needs something to be able to move more easily through the vacuum. Near her and the wreck is also the thrust device which she can pick up, and once she does she is equipped to go and find the ship parts so she can fix her ship. And once she fixes the ship, she can get going home.

Throughout the game, Laika occasionally speaks to herself, saying some of the following:

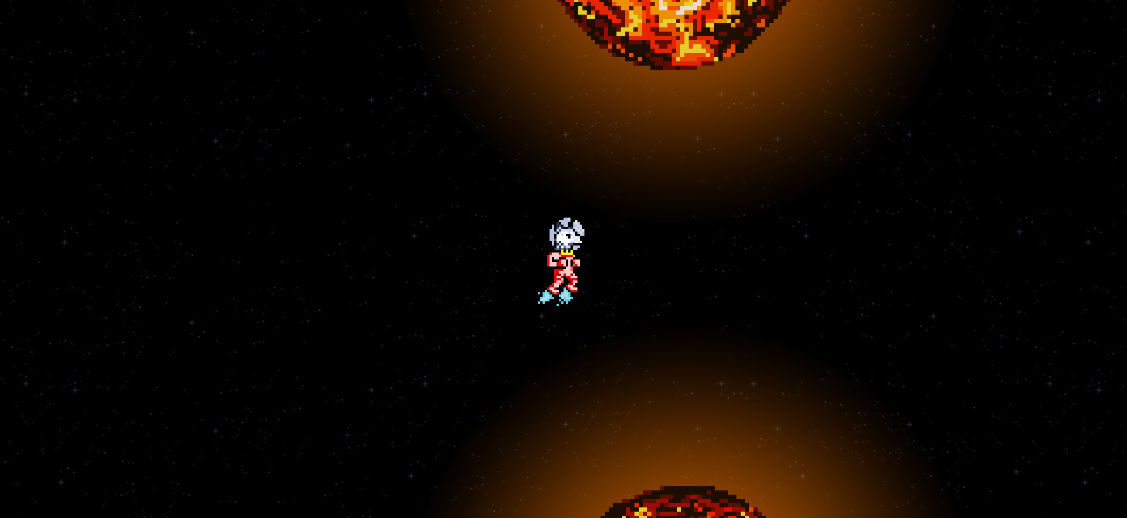
“Cats and rats! I prefer a different kind of walk.”

“I bet a hundred bones no one will believe me this happened.” and shortly after “No! A hundred treats.”

* 1. **Concept Art**

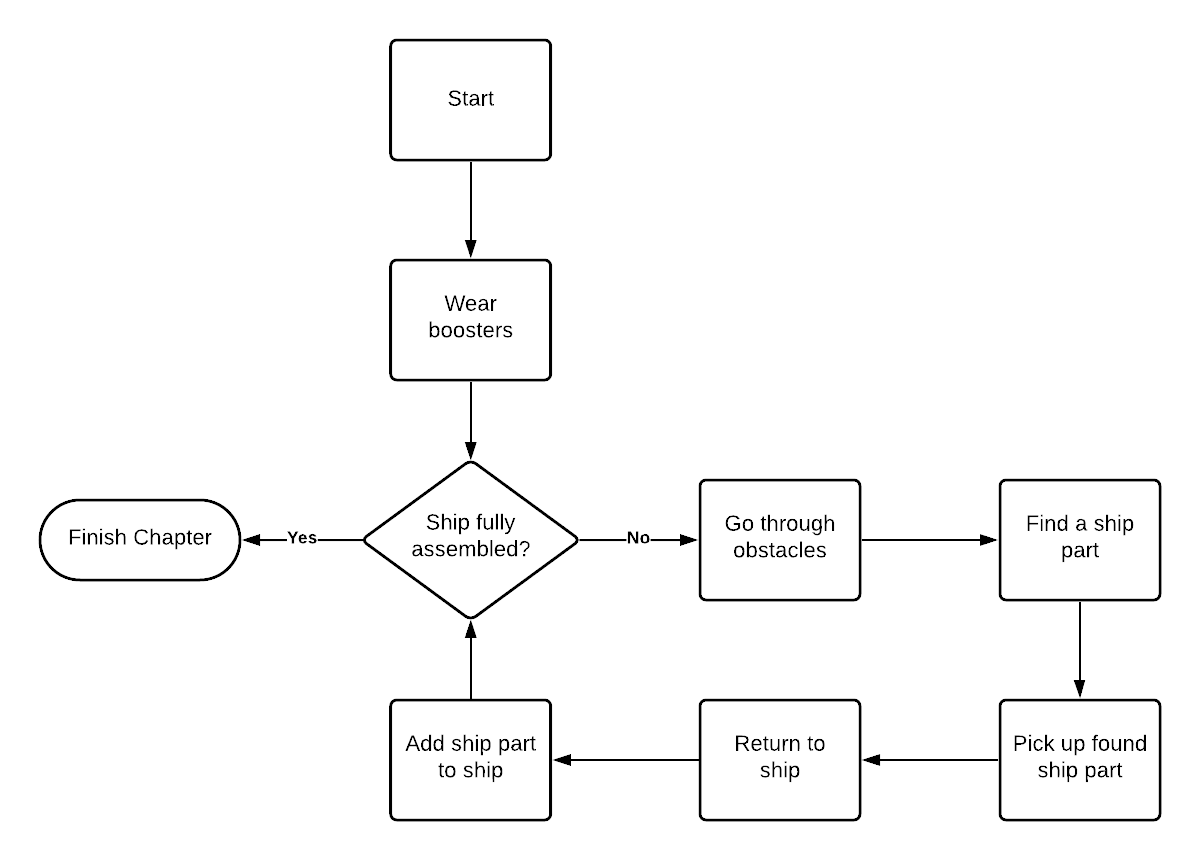
 





* 1. **Level Design**

Level design can best be shown as a flow chart:



This level design has a clear goal for the player - reassemble the exploded ship. The ship is assembled by completing smaller goals - picking up the scattered ship parts from various locations in space surrounding the starting point and bringing them back to the ship. The first goal, though, is collecting the residual thrust power and using it with Laika’s suit so the movement through space in search of ship parts is not extremely slow. All of these goals are communicated to the player through Laika’s monologues so they don’t have to guess at what they’re supposed to be doing, and hints are displayed as instructions for the player, explaining how to complete a goal so the player does not have to guess at the mechanics.

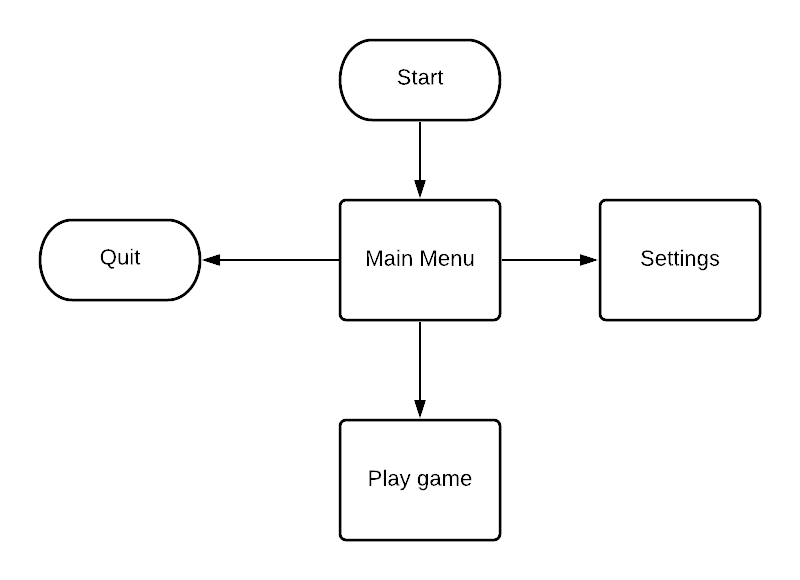
So as not to have the picking up of ship parts be too repetitive, there are obstacles between the player and every ship part such as meteor showers and burning planets they have to avoid. The ship parts themselves are hidden behind planets the player has to move away, and sometimes there is even a small reward - a dog cookie that restores a single life that might have been lost along the way. Obstacles are a good way to give the player some challenge, and the possibility of a reward in the form of restored lives lowers the stress of each lost life and gives the player a better chance to finish the game.

* 1. **Audio & Sound F/X**

Audio components of the game are music and special effects. Usage of ambiental music creates a peaceful atmosphere and simulates the tranquility of space environment. Special effects are used to describe specific behaviours such as activating the thrust power, being hurt by meteors or heat, firing Laika’s weapon, and others.  
Music takes special part in announcing danger. When a meteor shower begins, the calm music stops and the intensity of the more aggressive melody increases, which hints at some kind of danger.

* 1. [**Game Architecture**](about:blank)

**Main menu:**



Options screen consists of only audio base options and an exit button. There are no special game modes planned.

1. **Technical Document**
   1. **Visual Content**

A list of technical requirements concerned with the visual aspects of the game.

* General
  + 2D Sprites pixelated art
  + File Size Restrictions: none; files vary from 1KB to 79KB
  + File Format Type: PORTABLE NETWORK GRAPHIC (.png) or JOINT PHOTOGRAPHIC EXPERTS GROUP (.jpg)
* Player Elements (Laika)
  + Without power up
    - steady
    - upward idle
    - left idle
    - right idle
    - downward idle
  + With power up
    - upward moving
    - left moving
    - right moving
    - downward moving
* Player equipment
  + Bone power up
  + Moving power up
  + Weapon
    - Bullet
* Spaceship Elements
  + Spaceship
  + Explosion
  + Shipwreck
  + Spaceship parts
    - Astronaut cabine
    - Jet engine
    - Payload fairing
    - Radiation sensor
    - Transmitter sphere
* Space elements
  + Fireball with glow
  + Black hole
  + Planet
  + Meteors (size 1, size 2, size 3, size 4)
* Menu
  + Game logo
  + Background space picture
  + Buttons
    - New Game (selected, not selected)
    - Continue (selected, not selected)
    - Options (selected, not selected)
    - Exit (selected, not selected)
* Heads Up Display (HUD)
  + Health bar
    - Health empty
    - Health full
  + Display warning
* Global Elements
  + Background space picture
  1. **Audio Content**

A list of technical requirements concerned with the audio aspects of the game, containing types of audio content used in certain aspects of the game.

* General
  + File Size Restrictions: none; files vary from 6KB to 53MB
  + File Format Type: RIFF WAVE (.wav) or MP3 (.mp3)
* Menus
  + Main menu background music
  + Keyboard menu navigation
  + Choosing a menu option or changing sound settings
  + Opening pause menu
* Beginning cutscene
  + Background music
  + Ship beeping and movement sounds
  + Dialogue box typewriter effect
  + Ship shaking and rumbling
  + Ship explosion
* Ending cutscenes
  + Ship parts assembling with the base
  + Finished assembly flash
  + Ship flying away
* Player
  + Grabbing and releasing a ship part, equipping weapon
  + Successful assembly of a ship part
  + Jetpack movement
  + Pushing a planet
  + Shield positioning
  + Weapon shot
  + Enter ship
* Danger
  + Black hole gravitational pull
  + Meteor shower background music
* Health
  + Damage from meteor or burning planet
  + Eating a cookie
* Death
  + Death by black hole
  + Teleportation after death by meteors or burning planet
  + Game over
* Global Elements
  + Game background music
  + Dialogue box typewriter effect
  1. **Programming Content**

A list of game events that are implemented through the code presented as a loosely categorized and exhaustive list of scripts along with technical requirements concerned with the programming aspects of the game.

* General information and requirements
  + C# programming language
  + Built for desktop computers with keyboard controls
  + Fit for desktop monitor screens of any size
* Player Elements
  + mechanics and logic for grabbing, carrying and assembling ship parts
    - AssembleShipPart.cs, GrabShipPart.cs
    - AssembledShipPartList.cs, ShipPartID.cs
  + camera movement and following logic
    - AssignFollowPlayer.cs, CameraFollowPlayerScript.cs
  + entering the ship after assembly
    - EnterTheShip.cs
  + weapon picking up and usage
    - EquipWeapon.cs
    - Shooting.cs
  + the player’s health system, including logic for the canvas display update on changes
    - LaikaHealth.cs
  + movement mechanics and thrust device pickup
    - LaikaMovement.cs
    - PickupThrustPower.cs, ThrustMachinePickupController.cs
  + shield mechanics
    - Shield.cs
    - ShieldCollide.cs
* Obstacles, danger, power-ups
  + meteor shower spawning, movement, damage
    - activateMeteorShowers.cs
    - DestroyShower.cs, DestroyShowerMainParent.cs, DestroyShowerPiece.cs
    - MeteorMovementAndCollision.cs
    - MeteorShowerSpawner.cs
    - newMateorShowersAfterDeath.cs
  + logic for basic planets, push mechanics and trigger detection
    - BasicPlanetDirection.cs
    - BasicPlanetPush.cs, BasicPlanetPush[1-13].cs
    - BasicPlanetRange.cs
    - PushToGrab.cs, PushToGrab2.cs
    - PositionBasicPlanet.cs
  + black hole gravitational pull
    - BlackHoleMagnet.cs
  + bullet mechanics
    - BulletMovement.cs
    - DestroyBullet.cs
  + health power-up spawning
    - SpawNewCookie.cs
* Cutscene Elements
  + beginning cutscene control scripts for turning on/off animations and elements at the specific times
    - ActivateShaking.cs
    - AfterMoving.cs
    - BeginingCutsceneControl.cs
    - BeginningCutscene.cs
    - SpeedUpTheScene.cs
    - ShipMovement.cs
  + end cutscenes control script for turning on/off animations and elements at the specific times
    - EndingCutscene.cs
  + game credits for displaying text and smooth transition between scenes
    - CreditsController.cs
* Text and sound elements
  + Typewriter text restriction
    - CloseLastText.cs
  + displaying messages as hints or warnings for the player
    - DisplayCollisionWarning.cs
    - DisplayHint.cs, DisplayHintShipPart.cs
    - DangerTextRestriction.cs
  + dialogue box display logic and sound effects - mostly for typewriter effects
    - DialogueBoxMain.cs
    - DialogueBoxScript.cs, DialogueBoxScript2.cs
    - DestroyDialogueBox.cs
    - DisableDialogueBox.cs, EnableDialogueBox.cs
    - EndingSceneBoxClose.cs
    - TypeWriterEffect.cs, TypeWriterEffect2.cs, TypeWriterMain.cs
  + Laika’s monologues and logic to persist the knowledge about first encounters with specific objects
    - InsideMonologue.cs
    - EnableMonologueBasicPlanet.cs, EnableMonologueBeforePickup.cs, EnableMonologueBlackHole.cs, EnableMonologueFireball.cs, EnableMonologueInfrontOfMachine.cs, EnableMonologueInfrontOfWreck.cs, EnableMonologueShipPart.cs
    - InfrontOfBasicPlanetTriggerControl.cs, BeforeMachineTriggerControl.cs, InfrontOfBlackHoleTriggerControl.cs, InfrontOfFireballTriggerControl.cs, InfrontOfMachineTriggerControl.cs, InfrontOfWreckTriggerControl.cs, InfrontOfShipPartControl.cs
    - DetectBasicPlanetTrigger.cs, DetectBlackHoleTrigger.cs, DetectFireballTrigger.cs, DetectShipPartTrigger.cs
    - PlayerPrefsController.cs
  + transition between scenes
    - Chapter1TextControler.cs
  + logic to mute and unmute music and effects based on player settings
    - DisableMusic.cs, MenuMusic.cs
    - DisableSFX.cs, MenuSFX.cs
  + handling of sounds played on triggers
    - JetpackSoundManagerScript.cs
    - SoundManagerScript.cs
* Menus
  + main and options menu logic
    - ExitOptionsButton.cs
    - MenuBTNs.cs
    - MuiscButton.cs, SFXButton.cs
    - NewGameBTNScript.cs
    - OptionMenuButtonsControler.cs
  + pause menu logic
    - PauseMenu.cs
    - AutoSellect.cs
    - ContinueGame.cs
    - ExitToMenu.cs
    - PlayExitSound.cs
    - SelectMusicButton.cs
* Other
  + saving initial transform setups of, e.g., ship parts for after-death resetting purposes
    - InitialTransform.cs
  + some helper functions and enum definitions
    - Utils.cs
  + helper function to randomise Laika’s speech when hurt
    - RandomAnimGenerator.cs
  1. **Code Structure**

All of the project’s scripts use the System.Collections, System.Collections.Generic and UnityEngine namespaces. The scripts in charge of working with UI elements (most often text elements) also use UnityEngine.UI. The scripts in charge of switching between scenes use UnityEngine.SceneManagement for the SceneManager class.

Only the "Utils" script doesn't use the MonoBehaviour class. MonoBehaviour is the base class from which each Unity script is derived. In the mentioned "Utils" script, the enumeration data type is used to define the state of movements, messages to be displayed, and IDs of the collectible ship parts. It also contains these methods: "isGrabbable", "isGrabbableShip", "isGrabbableItem", "isBoundary", and "isShip", which are self-explanatory. All of them accept a string as an argument, representing the tag of a particular object, and return a boolean value. The remaining method is "getShipPartID", which accept a GameObject and returns the enum value attached to it through its "ShipPartID" script component.

The scripts "DisplayHint", "DisplayHintShipPart", and "ShipPartID" use the "Utils" script to identify the state of other elements. The "DisplayHint" script uses protected functions such as "isCollisionTag" and "playAnimation", which are also the only protected methods in the code. The only overriden method is protected void "onEvent" which, in the "DisplayHintShipPart" script, signals the Animator component attached to an object whether or not it is necessary to play the animation.

Scripts that extend the MonoBehaviour class can implement its functions "Start", "Update" (or "FixedUpdate" if used for any physics simulations, such as moving an object with speed) and "Awake" if needed. These are used for initializing variables or saving states before calling the "Start" function.

The way scripts are bound to objects depends on whether the objects use the scripts or the scripts use the objects. For some objects, scripts are linked directly through the editor and can call functions marked as public on some of the object’s events like OnMouseDown or OnSelect (direct user-object interaction via input). Objects used by scripts for the purpose of modifying any of their properties in any way (e.g., calling their Animator component, modifying their Transform component, or changing their current image or text), are fetched in scripts via their name or tag using, respectively, "GameObject.Find" or "GameObject.FindGameObjectWithTag". Some objects are manually associated with scripts, via the editor, if the script has at least one declared GameObject variable marked as public. A script component of an object can be accessed by using the "gameObject.GetComponent<ScriptName>" method. It can then be stored into a variable to allow later access to its public attributes and functions without calling the method again, or they can be directly accessed by chaining them to the “GetComponent” method call.

Commonly used MonoBehaviour methods are "OnTriggerEnter2D", " OnTriggerExit2D", "OnTriggerStay2D", "OnCollisionEnter2D", "OnCollisionExit2D", and "OnCollisionStay2D". These methods receive, respectively, a Collider2D or Collision2D object as an argument, which represents the object in collision with the object the script is attached to. Through them, the tag of the object can be accessed and, depending on the object the tag is related to, the script can call functions that perform actions defined for interactions with specific colliders.

"Start" and "Awake" are mostly used only for initializing variables at the beginning of a scene or when starting a co-routine. IEnumerator and "StartCouroutine" are mainly used to count down a certain number of seconds between two actions being performed.

All other code is executed in the "Update" method of MonoBehaviour scripts. It is called on every frame update, and is therefore used for logic such as motion mechanics, which requires constant updates of state.

The MonoBehaviour "Invoke" method is used to delay calling certain functions. Delays are useful for cutscenes where sets of actions depend on each other and have to be called at different times.

Time is manipulated by using the Time.timeScale variable, which, in our case, served to speed up the performance of all activities in the scene (to speed up the entire scene). The Time.deltaTime variable was used to create motion mechanics at a certain speed defined as float value.

All movement except thrust power movement is vector motion, while for thrust power "rb.AddForce (transform.up \* (-thrust) / AdditionalForce, ForceMode2D.Impulse)" is used. That expression adds an "Impulse" force to the Rigidbody2D component and, combined with the Linear Drag component, creates motion physics defined in the "FixedUpdate" method.

To save the state of the game, the "PlayerPrefs" class is used, which is a class that stores player settings between game sessions. They can be accessed by calling (most often) "GetInt" and "SetInt", and sometimes "GetString" and "SetString”. The stored numeric or string value represents a specific state in which some associated object should be when that value is checked again.