15-12-2020

Space Quest

Design Documentation

Luis Alberto Durán Chávez

Index

Image Index

1. Introduction

This is the Space Quest design document. A videogame created for the pc platform in order to complete the challenge imposed by Amber KO studio in Guadalajara. In this document you'll find all the information related to the videogame as diagrams, stats and playability.

1.1 Concept game

space quest is a video game in which we control a lonely spaceship in space with the current objective of surviving the wave of enemies that is coming and the oxygen that is ending

1.2 Genre

Genres may encompass a wide variety of games, leading to even more specific classifications called subgenres. For example, an action game can be classified into many subgenres such as platform games and fighting games. Some games, most notably browser and mobile games, are commonly classified into multiple genres.

My videogame is going to have the next Genres:

Adventure: is a video game in which the player assumes the role of a protagonist in an interactive story driven by exploration and puzzle-solving. In Space Quest we still working on the adventure mode where is going to include new levels, puzzles and more destruction;

Shoot 'em up: is a genre of computer games in which the main gameplay focus is on shooting, such that other aspects of the game are often simplified to facilitate this. In Space Quest we are going to shoot a <u>lot</u>, trust me.

1.3 Propose

The propose of this game is demonstrate my knowledge creating video games in the programing scope using free tools, create an acceptable product and be able to complete the Amber KO challenge.

1.4 Playability

1.4.1 Survivor

Space Quest only have one level called Survivor where you will have to do your best to stay alive as enemies will come from anywhere like a wave.

1.4.2 Updates

The player will have the option to upgrade the spaceship, upgrades like more speed, armor, more O2 or Turbo speed that can be purchased.

1.5 visual style

We decided to opt for a retro-style visual section, since the genre is a classic of old arcade mechanics, fulfilling the objective of giving the player the feeling of playing in a real arcade.

2 Game Mechanics

In this section we will analyze in depth the mechanics of the game, levels, difficulty as well as the resources available to fulfill our objective.

2.2 Playability

Space Quest only has one level for now, but that does not mean that it will be easy, you will have to upgrade your ship and to do so you must use coins to buy these improvements if you want to be able to survive the enemies that are coming.

Coins

This will be your main and only source of income, with this item you will be able to purchase updates for your spaceship but is not going to be easy to obtain.

Oxygen

This item is essential for survival, as picking up one of these will temporarily increase your oxygen bar.

Enemies

If you were asking you how to get the last items (coins and oxygen) well... if you kill one enemy their going to drop one of these, take it to survive and get more upgrades.

2.3 Updates

As we mentioned before, updates will be essential for a long battle in space, but what kind of updates do we have? Let's see next

2.3.1 Speed

Increase your base speed. This upgrade can be purchased 5 times and get the max level.

2.3.2 Hp

reduce projectile damage. This upgrade can be purchased 5 times and get the max level.

2.3.3 O2

reduces the amount of O2 (Oxygen) used per second. This upgrade can be purchased 5 times and get the max level.

2.3.4 Turbo

reduces the amount of Turbo used per second when is used. This upgrade can be purchased 5 times and get the max level.

3 Interface

In this section, each of the Space Quest screens will be specified in detail, as well as the diagrams that make up the game in order to publicize the flow of the game.

3.1 Diagrams and screeshots

The following flow chart shows us the tour of the game starting from the beginning (Start), going through the first screen and progressing to the main menu where we will find the options of:
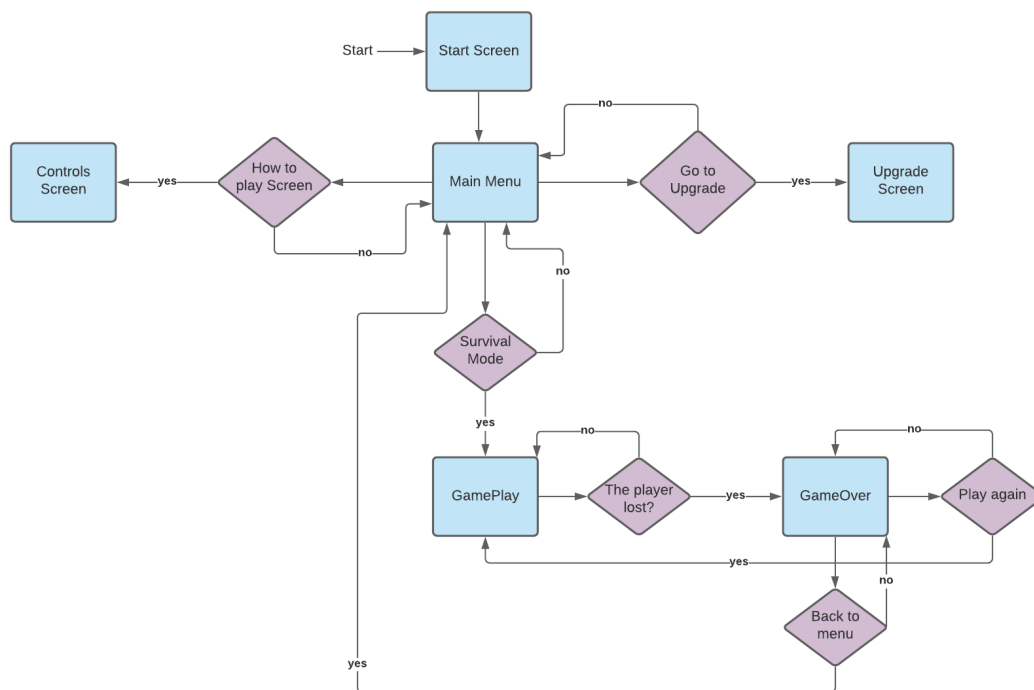

Figure 1 Flow Diagram

3.1.1 Menu:

In this section you can be able to see all your coins, your best score and the last one.

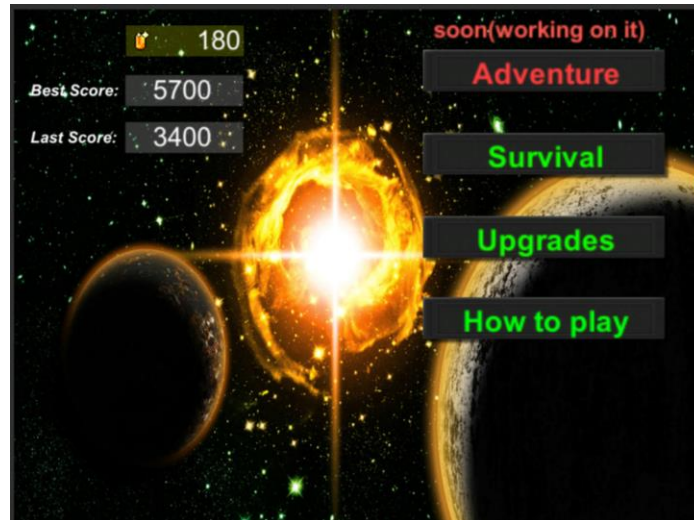Three option are going to be enable in the game, Survival, Upgrades and how to play.

Figure 2 Menu screeshot

3.1.2 Adventure:

This screen is disable for now but in a future will contain the adventure mode with new levels, enemies and puzzles.

3.1.3 How to play:

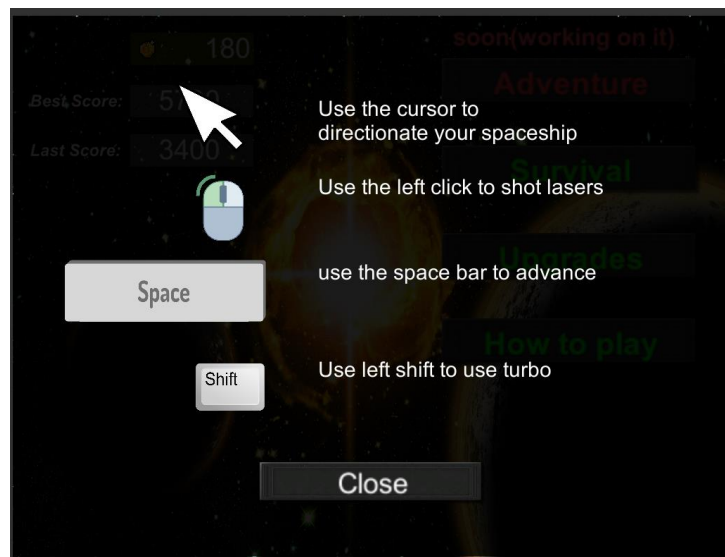This option shows us the controls of the game to play correctly.

Figure 3 Controls Screeshot

3.1.4 Update:
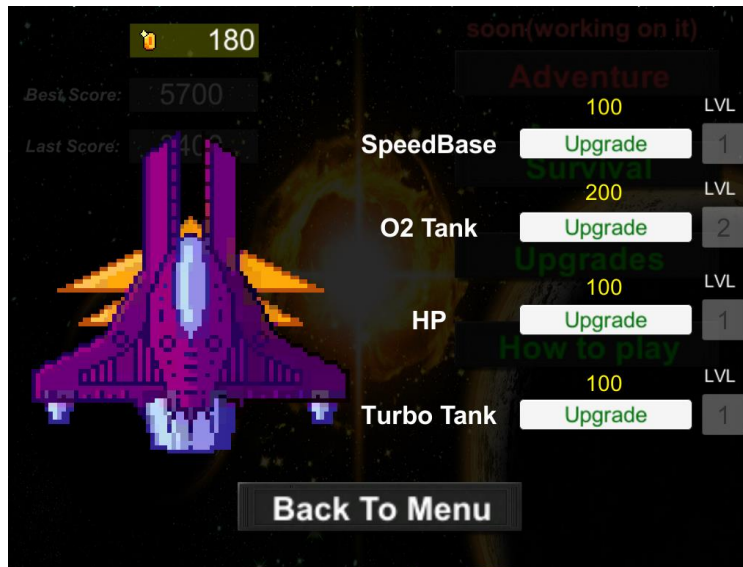This option shows a screen where is possible to update our spaceship.

Figure 4 Upgrade Screeshot

3.1.5 Survival:
This option starts the game and send you to the battle area.

Figure 5 Gameplay Screeshot

3.1.6 Game Over

This screen shows you are dead and lost the battle but don't worry you can play again or go back to the menu and get stronger with the upgrades.
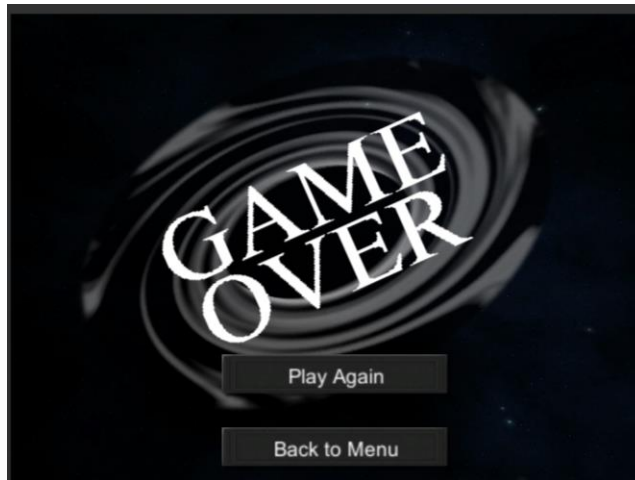
Figure 6 Game Over screeshot

3.1.7 UML Diagram

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

3.1.8 Scriptable Object

Scriptable Objects are amazing data containers. They don't need to be attached to a GameObject in a scene. They can be saved as assets in our project. Most often, they are used as assets which are only meant to store data, but can also be used to help serialize objects and can be instantiated in our scenes.

I create an scriptable object to obtain the information about the player and the Spaceship status to control the game, upgrades, coins and score stored in Savedata(ScriptableObject).
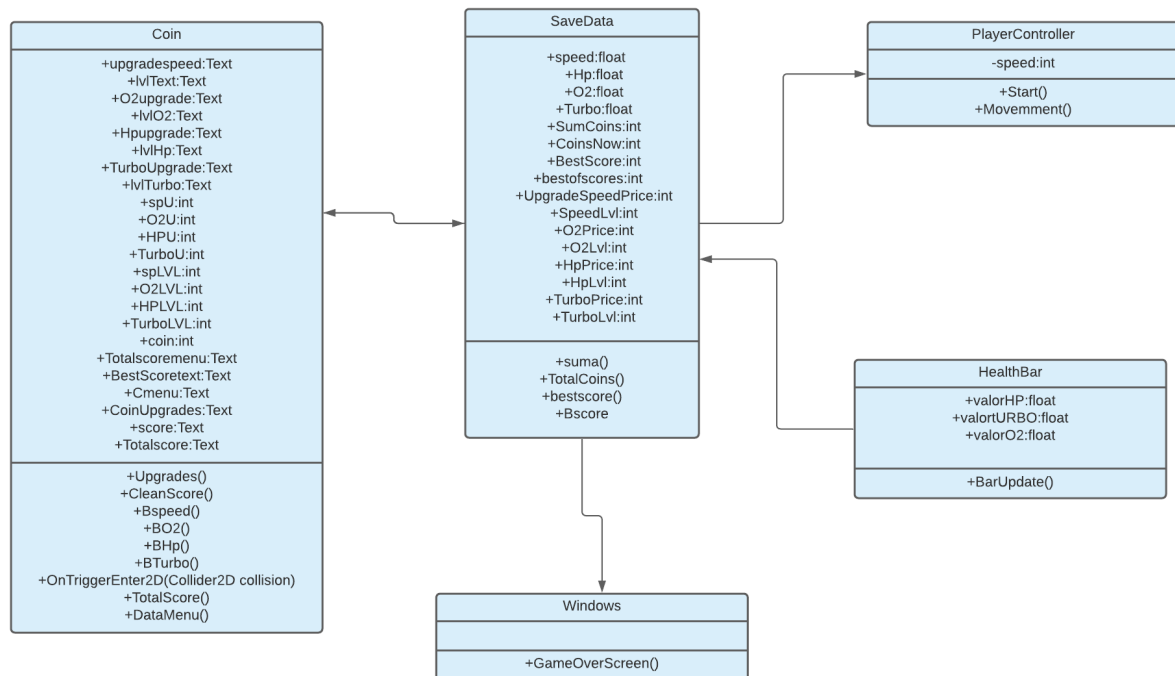

Coin

+upgradespeed:Text
+lvlText:Text
+O2upgrade:Text
+lvlO2:Text
+Hpupgrade:Text
+lvlHp:Text
+TurboUpgrade:Text
+lvlTurbo:Text
+spU:int
+O2U:int
+HPU:int
+TurboU:int
+spLVL:int
+O2LVL:int
+HPLVL:int
+TurboLVL:int
+coin:int
+Totalscoremenu:Text
+BestScoretext:Text
+Cmenu:Text
+CoinUpgrades:Text
+score:Text
+Totalscore:Text

+Upgrades()
+CleanScore()
+Bspeed()
+BO2()
+BHp()
+BTurbo()
+OnTriggerEnter2D(Collider2D collision)
+TotalScore()
+DataMenu()

SaveData

+speed:float
+Hp:float
+O2:float
+Turbo:float
+SumCoins:int
+CoinsNow:int
+BestScore:int
+bestofscores:int
+UpgradeSpeedPrice:int
+SpeedLvl:int
+O2Price:int
+O2Lvl:int
+HpPrice:int
+HpLvl:int
+TurboPrice:int
+TurboLvl:int

+suma()
+TotalCoins()
+bestscore()
+Bscore

PlayerController

-speed:int

+Start()
+Movemment()

HealthBar

+valorHP:float
+valortURBO:float
+valorO2:float

+BarUpdate()

Windows

+GameOverScreen()

Figure 7 UML diagram