Space Frontier

Unity 3d Game Template

Requires **Unity V5.6+**

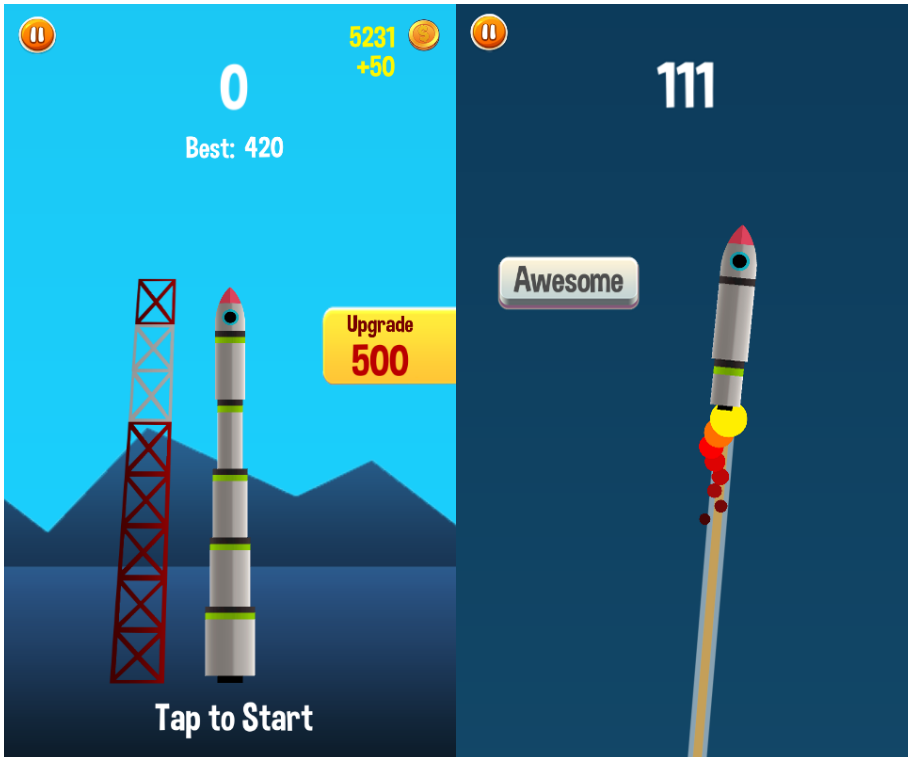
Supports Android, iOS, WebGL, Windows and Mac

Dear Customer,

Thank you so much for purchasing this game framework. Here you can find the most important information on how to use this project efficiently. All script and code assets are fully commented, but if you ever needed a hand on a block of code or anything else, feel free to contact us at www.finalbossgame.com . We'll try our best to support you with your questions as soon as possible.

Overview

Space Frontier is a fun and addictive physics rocket game template brought to you by Finalboss game studio. In this game, you are controlling a rocket which fly upward and you need to manage its fuel consumption to make it reach the highest possible height.



The game accepts both touch and mouse inputs, and thus, can be tested on **Android**, **iOS**, **WebGL** and **Stand-Alone** platforms.

This game kit needs no 3rd party plug-ins to works. It runs and builds out of the box. All you need to do is to load the kit inside Unity, set the project on the desired platform and hit "Build" to receive your game in no time!

Monetization

We have integrated AdMob ad system into the kit. You are free to set your own Admob IDs into the AdManager prefab which is also available from within the "Init" scene. AdManager is configured to show a banner ad at all time, while only showing an interstitial ad when the game is pauses or over. You are free to add more events for showing ads.

Game-Play

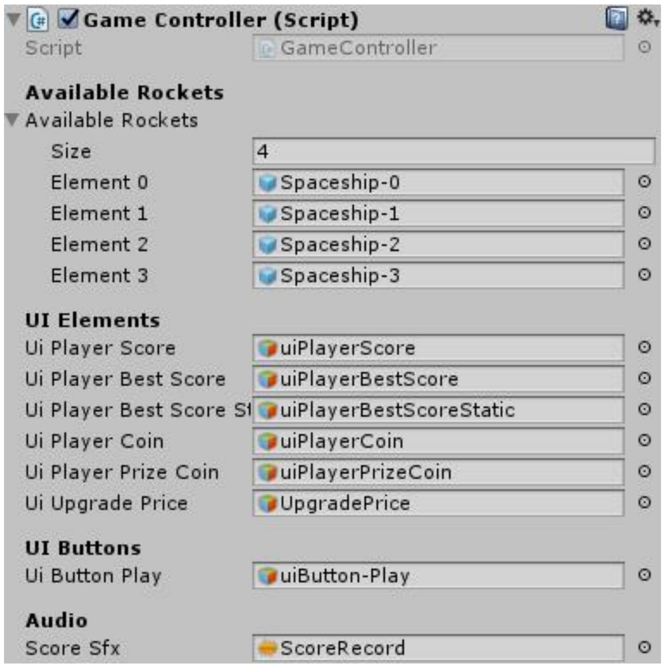
Your mission is to launch your rocket as high into orbit as possible. Simple touches control when you release each stage in your rocket. Earn in-game currency from successful launches and spend it to acquire new parts for the rocket.

Easy to play but hard to master gameplay means you'll be coming back again and again for one more boost.

Classes

This game framework uses a few separate classes to control the game's main routine. All these classes are fully commented and you can easily guess the dataflow. But we will try to introduce them here at a glance:

- **GameController:** This is the main game controller class. It is responsible for maintaining player coins, rocket upgrade level, current and best scores, monitoring game over event, enabling/disabling UI elements and saving game progress.



Game Controller (Script)

Script	GameController
Available Rockets	
Available Rockets	
Size	4
Element 0	Spaceship-0
Element 1	Spaceship-1
Element 2	Spaceship-2
Element 3	Spaceship-3
UI Elements	
Ui Player Score	uiPlayerScore
Ui Player Best Score	uiPlayerBestScore
Ui Player Best Score St	uiPlayerBestScoreStatic
Ui Player Coin	uiPlayerCoin
Ui Player Prize Coin	uiPlayerPrizeCoin
Ui Upgrade Price	UpgradePrice
UI Buttons	
Ui Button Play	uiButton-Play
Audio	
Score Sfx	ScoreRecord

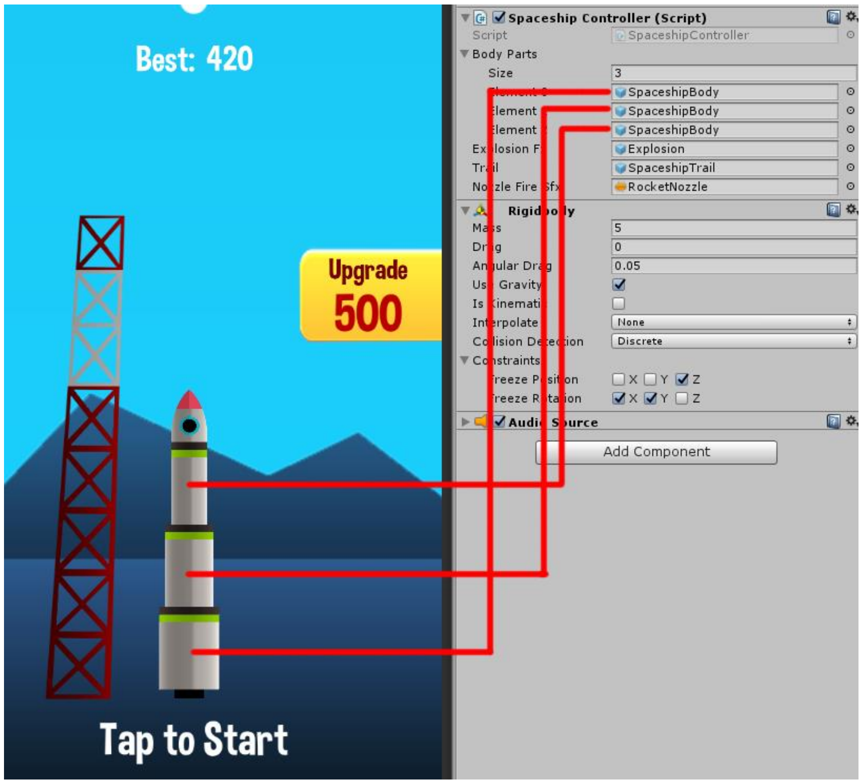
- **UserInputManager:** This class manages pause and un-pause states. Please note that the pause scene is the best place to show your full screen ads. It is also responsible for handling all input/touch events on ui buttons and elements.

- **CameraController:** Main camera manager. Handles camera movement, smooth follow (main rocket), and bounding by movement limiters. Note that this is the game-play camera. All UI rendering is done by UICamera in another thread.

- **CounterController:** It is used in the first run of the game. counts from 3 to 1 and then disappears.

- **StatusController:** We need to show a message after each successful body detach. This status controller receives the ID and displays a message accordingly.

- **SpaceshipController:** This is the main spaceship controller class which handles bodypart activation, fuel consumption, applying force & torque to make the rocket fly upwards, and explosion sequence when player ran out of fuel or clicks too late. One important thing to note when editing bodyparts inside the bodyParts array. You need to respect the order of parts and add them from the bottom to top, meaning that the index 0 of array should be filled with the first part (the bottom part) and so one. There is an image in the docs that describe this with more detail.



Spaceship Controller (Script)
Script: SpaceshipController
Body Parts
Size: 3
Element 0: SpaceshipBody
Element 1: SpaceshipBody
Element 2: SpaceshipBody
Explosion Fx: Explosion
Trail: SpaceshipTrail
Nozzle Fire Sfx: RocketNozzle

Rigidbody
Mass: 5
Drag: 0
Angular Drag: 0.05
Use Gravity: ✓
Is Kinematic: ☐
Interpolate: None
Collision Detection: Discrete
Constraints
Freeze Position: ☐ X ☐ Y ✓ Z
Freeze Rotation: ✓ X ✓ Y ☐ Z

Audio Source

- **SpaceshipBodyController:** Each body part has its own controller. Please remember that we need to carefully set each part's position inside the main rocket and then add it to bodypart array of the rocket to make it work. We also need to add the parts in a correct order -> index 0 is the bottom part and with last index is the top part. look at the structure of on spaceship prefab to find out more about this setup.

Add a new spaceship skin

To add a new skin, please follow these steps:

1. Create 3 new skin images "body, head and divider" and place them in a folder "Skin-n" inside textures.
2. Add a new field to the "Skins" array of the " GameController" class, and insert these 3 new images into the respected fields.
3. Add a new skin button inside " SkinCell" game object in hierarchy and place it under "BtnSkin-2". Rename this new cloned button to " BtnSkin-3" and also set its ID to 3 via inspector. Place it on an appropriate position inside UI.
4. Add this new button to the "SkinButtons" array of the "GetNewSkin" class.
5. You are done!

Have any questions?

If you have any questions, feel free to ask us at http://www.finalbossgame.com and we will get back to you as soon as possible.

Credits

The beautiful explosion system used inside the space frontier game kit is a courtesy of "Red Shark" game studio and can be found here: 2d Flat Explosion