RETRO FPS KIT v1.6

DOCUMENTATION (shortcut version)

Thank You for buying this package. If you like it, please make sure to **rate and review** it here:

https://assetstore.unity.com/packages/slug/209391

Contact Support: mcgamestudios@gmail.com

Contact Youtube: https://www.youtube.com/channel/UCP-s_g2NMIhhugCJDc89Ifg

VIDEO Tutorials to every single chapter are going to be available under this link:

FPS RETRO TUTORIAL: https://www.youtube.com/watch?v=Xq44QmxSzjc

This Documentation is going to be based on chapters for each possible category. It is going to be written based on the ONLINE Video for better understanding of each functionality found in the project.

This Documentation is going to get more chapters for each coming update.

Table of Contents

Chapter 1: Directions and Scenes

Retro FPS Kit has been professionally sorted to find any needed component within matter of seconds. As you can see on the picture, each folder represents described files.

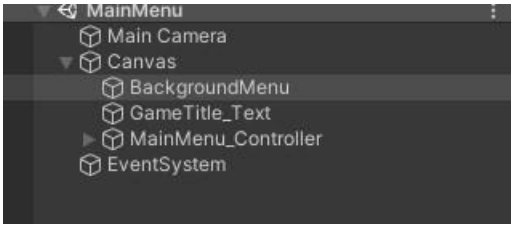
To start the game please go to: Scenes Folder and click on "MainMenu". This is the first scene that should be in Building Settings.

Level 1: This is the testable gameplay scene from the Video Trailer.

NewLevelExample: This is the "Next Level" scene. It is created only to describe you "How to load new level after defeating first one".

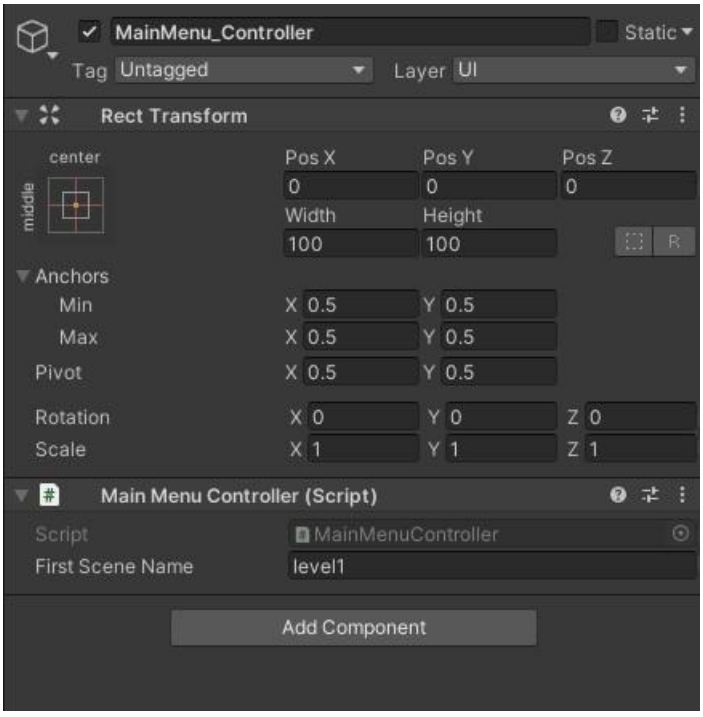
Chapter 2: Main Menu

MainMenu.Scene has animated (using Animator) Background. You can change that Animator within the “Canvas/BackgroundMenu” object in the Hierarchy.

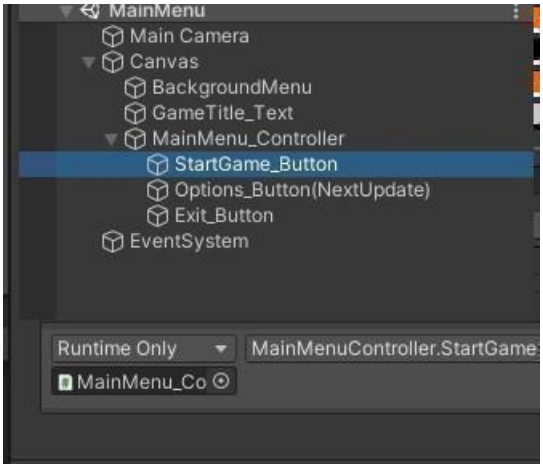


MainMenu_Controller Object contains all Buttons and core functionality for those buttons. It has script attached that loads the first level after clicking “Start Game”.

The MainMenuController Script can be edited for your needs to create: Options and other button functionalities. Options are going to be added with saving and loading in update 1.1.



To change the scene that is being loaded on “Start Game” button, simply write name of your scene within the MainMenu_Controller object in the Inspector. Now it loads level “level1”.

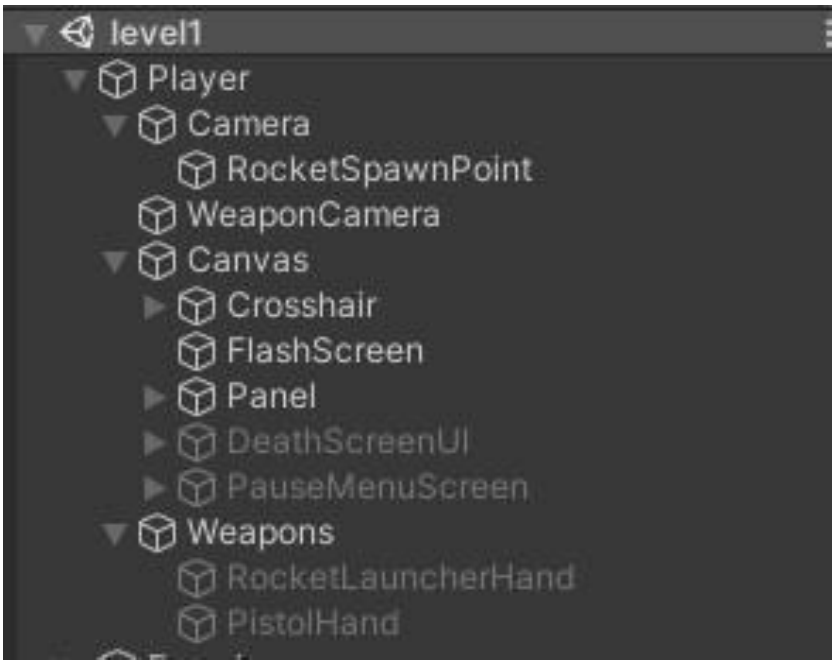


As you can see, Buttons are having “OnClick” event taken from the MainMenu_Controller object. So, every button is taking functions right from the Controller script.

Chapter 3: Player

On the scene “Level1” we can find quite few more important objects. One of them is “Player”. This is our Playable character. It’s simple collider with Canvas attached to it.

Player has several components. I will describe them one by one.



Player – Based Player Object.

Camera – Keeps the Camera for our Player. Camera contains "Head Bobbing" Script.

Changing its values will allow you to change how player "move its head" while walking.

RocketSpawnPoint – It's the place where Rocket Weapon should be placed and start shooting.

WeaponCamera – It's the camera for our weapons. It contains two scripts: Head Bobbing (this one allows us to change how weapons "jump" while moving) and "Dynamic Crosshair". Dynamic Crosshair allows us to set the crosshair for the Player with weapon attached.

Canvas – Canvas holds all UI for the player.

Crosshair is our cross so you could see where Player is aiming.

Flashscreen is the little "flash" effect that appears when we are being HIT (red) or when we collect some bonuses (blue). It flashes and fades for about 5 seconds then disappears.

Flashscreen contains script that allows you to change "how long" flash should be fading out.

Panel – This is our character statistics UI. It contains Health Text Values, Armour Text, Our Avatar & Ammo Values. You can freely change places of those by grabbing each component and relocating it on the scene window.

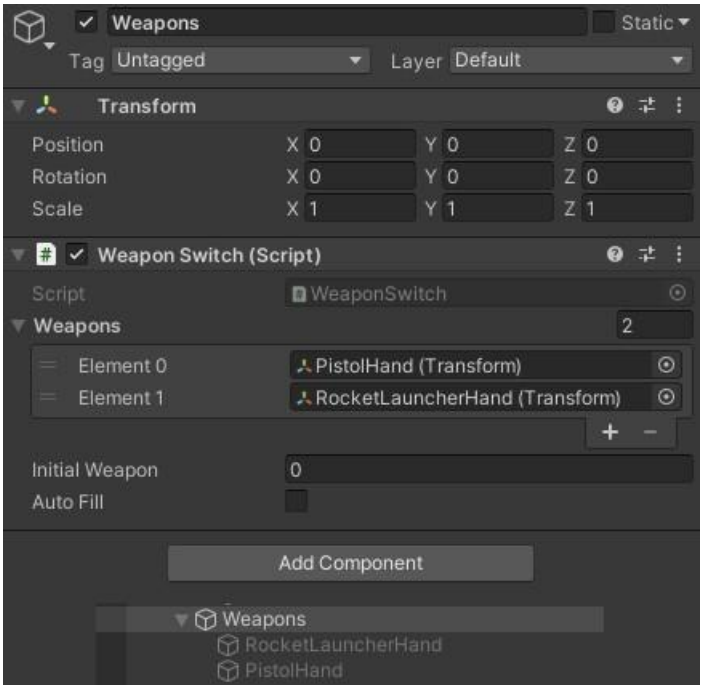
DeathScreenUI – It's our Death Screen. Once Player's health goes to 0, this UI window shows up.

PauseMenuScreen – It's our Pause Menu UI. Once Player clicks Escape, this window shows up.

Each of those UI windows contain MenuController. Pause and Death Screens are having their buttons functionality written in MainMenuController Script.

Chapter 3.1 – Weapons

Under Player Object we have "Weapons" Object in Hierarchy. This one contains all our weapons that Player could use.

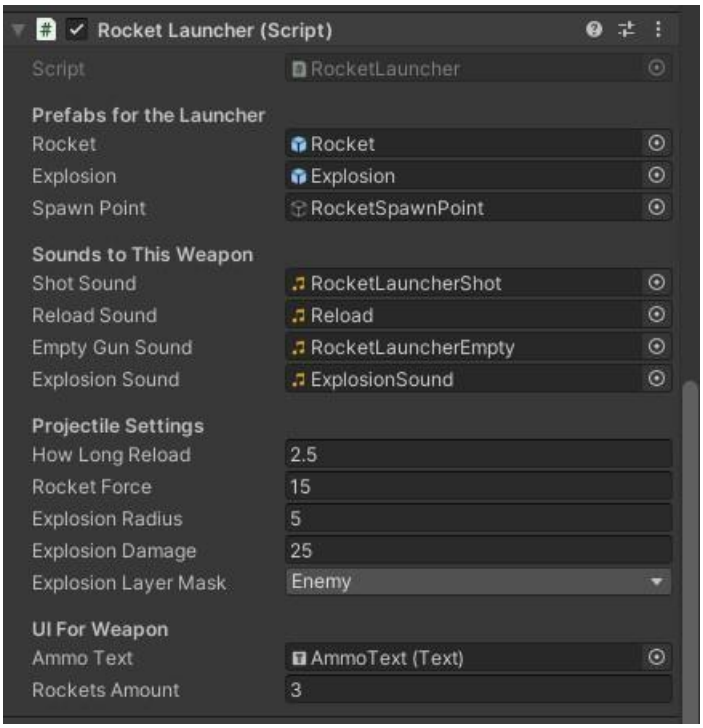


Weapons object has "WeaponSwitch" script that allows you to add more weapons to the selection. Player can change weapons in game by clicking Numeric Keys (1,2(…)) or by scrolling.

For now there are two weapon types: Pistol & RocketLauncher. Online Video shows how to add new weapon.

Chapter 3.1.1 – Rocket Launcher

Rocket Launcher has several options you can change. They all can be changed in the Inspector Window, so there are no needs to script anything.



Script however comments every single detail, so for your better understanding – please read the script comments.

Rocket – it's the Rocket Prefab (It's Graphic Prefab)

Explosion – It's the Rocket Explosion Prefab (Graphics on explode)

Spawn Point is previously described spawn for the Rocket to appear on screen.

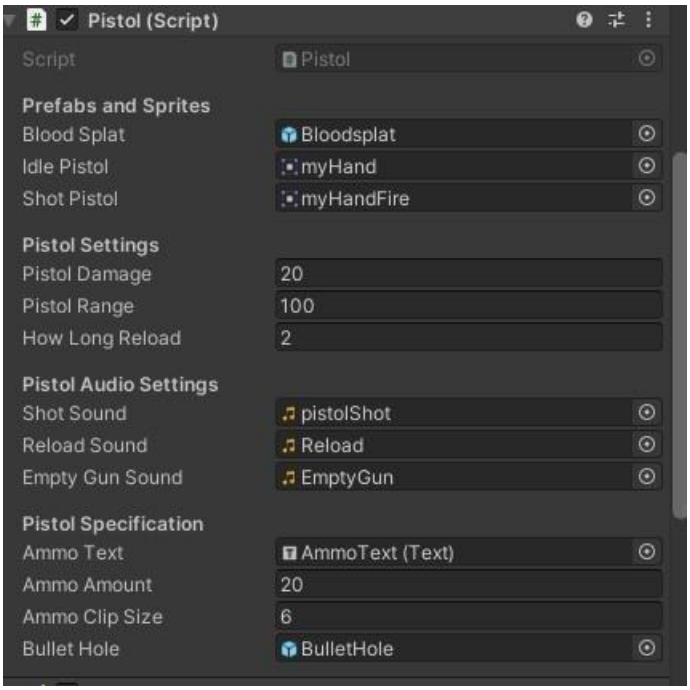
Sounds can be change for each described in the Inspector.

Projectile Settings – Those settings can change how Launcher behaves. How long it reloads, how strong it is, how far it should shoot and what is the explosion radius and damage.

Ammo Text & Amount – Those are for the UI described previously. You can find the AmmoText under the: Player/Canvas/Panel/AmmoText. This is where the Ammunition value is going to be displayed.

Chapter 3.1.2 – Pistol

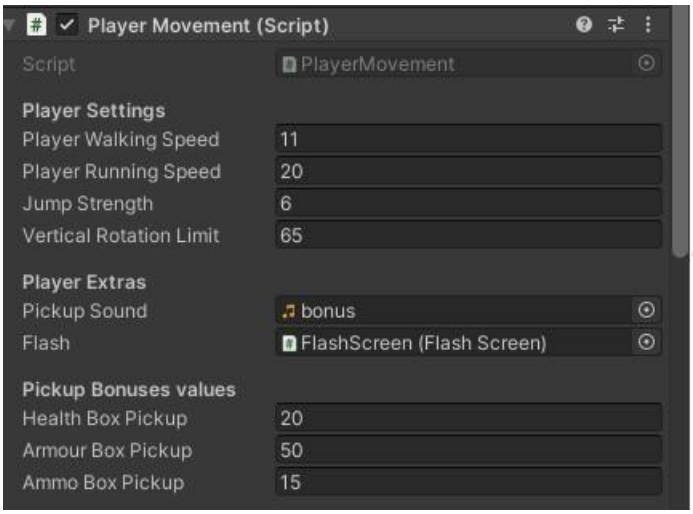
The same rules apply to the Pistol Weapon. It has its Sound and Weapon Settings. As well as maximum and loaded ammo.



MyHand Sprite is the graphic of your pistol (with hands). MyHandFire is the graphic that is being shown on “shot”. So when Player shoots – MyHandFire is being displayed for around 0.5 second.

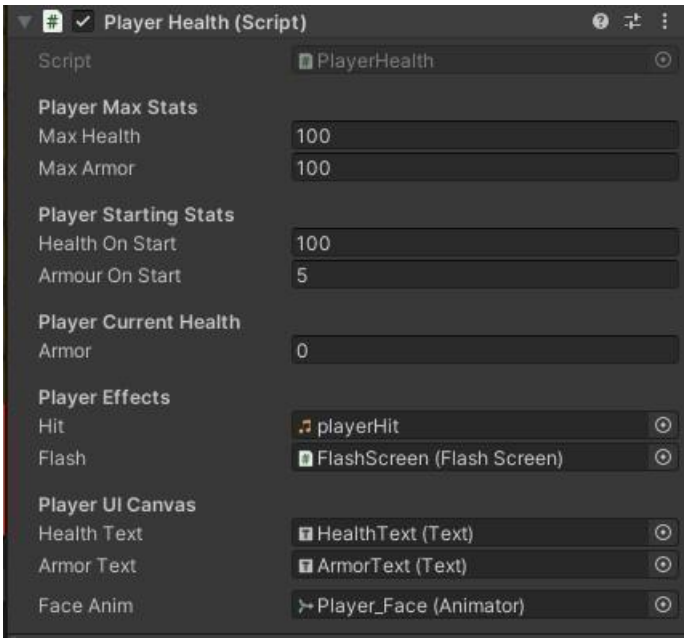
Chapter 3.2 – Player Settings

Player Object in the Hierarchy has its own scripts that allow you to change his values.



Player Movement script allows you to change his walking values. Also it contains how much of the bonuses are you getting by picking up: Health Box, Armour Box or Ammo Box.

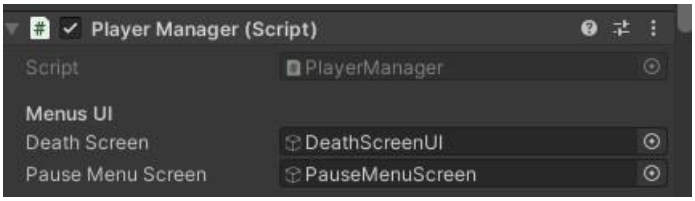
For example: If you pick up health – this script tells the player to add +20 to his health.



Player Health Script allows you to change starting and maximum values for the Player. Now he can have maximum of 100 health and armour. Also Player starts with 100 health and only 5 armour.

Thanks to that functionality you can create hardcore mode etc. You can let players to start with only 1 HP.

Player Current Health means how much health he has at this particular moment. It is visible only on gameplay.

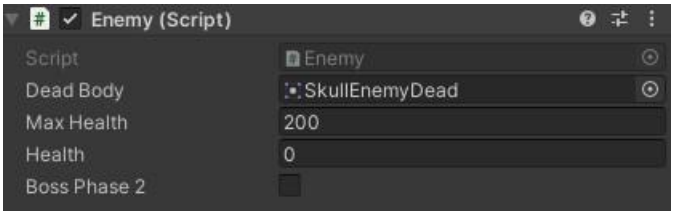


Player Manager contains all UI windows like: Pause Menu and Death Menu. Simply add new menu to the list (in the script) for example: Inventory and drag it to this script so you could use it in the game. Please read the script comments as those will tell you more of how to use it.

Chapter 4: Enemies

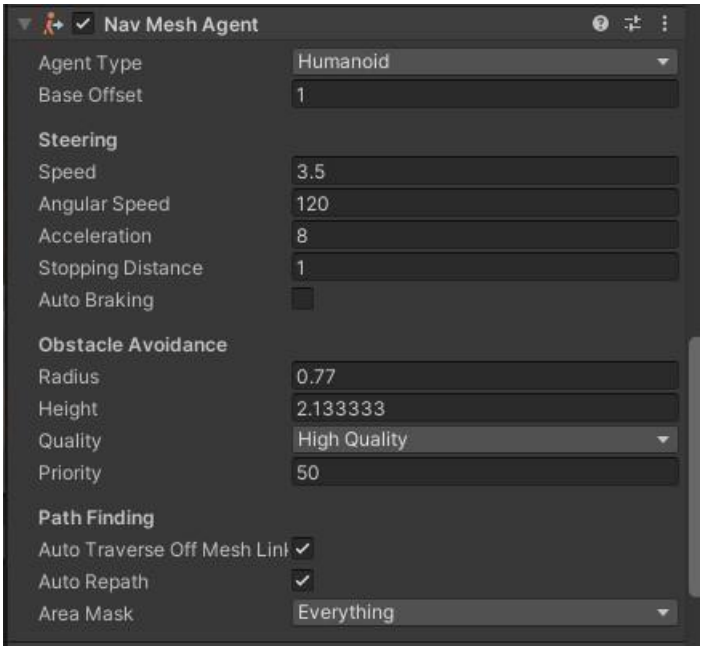
Enemy contains several scripts that I'd like to describe for your better project understanding.

Enemy has Sprite Renderer that allows us to see his graphics. Enemy is always looking to the Player's direction thanks to the "**FaceCamera**" script attached to him. This script is also attached to the bonuses.



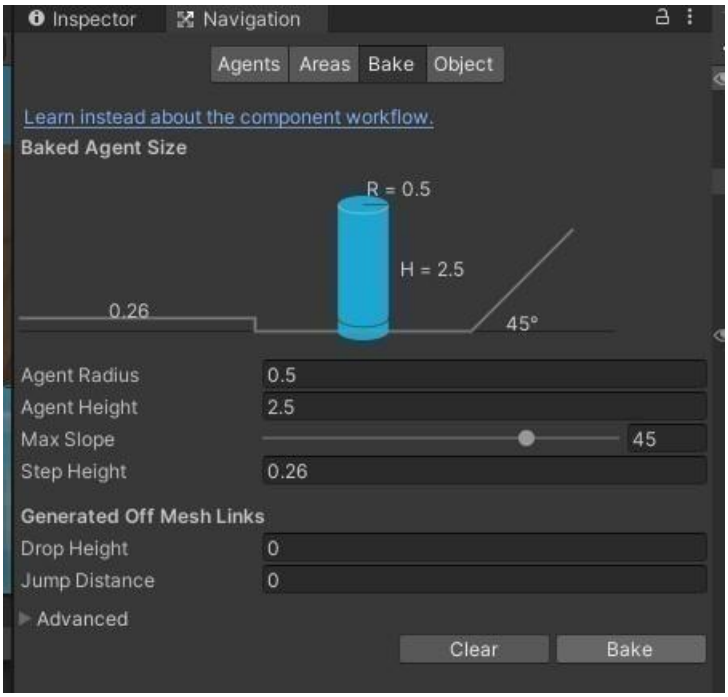
Enemy Script contains his Dead variant Graphic – his max health and current health (visible only during Gameplay).

Boss Phase 2 – if you mark it as "true", enemy will have second attack phase when being under 50% of its life. After we take him 50% of life he starts attacking faster and stronger. (Everything is described in the script).

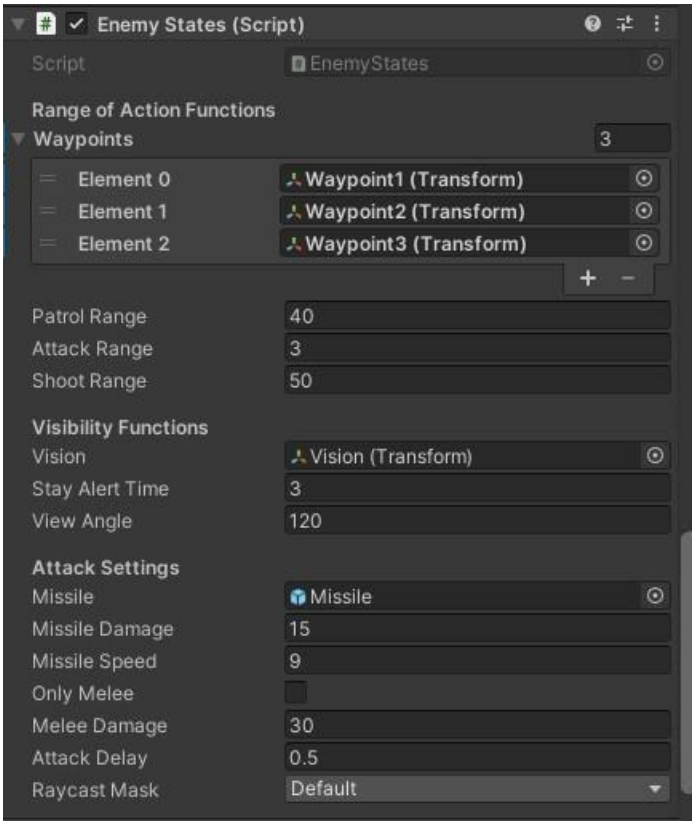


Nav Mesh agent allows the Enemy to move on the Level. We also need to bake the level before he can move. So if you add any other floor please go to:

(Upper Bar) Window -> AI -> Navigation. In the inspector of the Navigation just click on "Bake" and then "Bake". As seen below.



Enemy also contains states and waypoints.



Waypoints are just simple empty objects located on the level. This enemy will walk between those points until he activates new state. He has 4 states: Patrol, Attack, Chase and Alert.

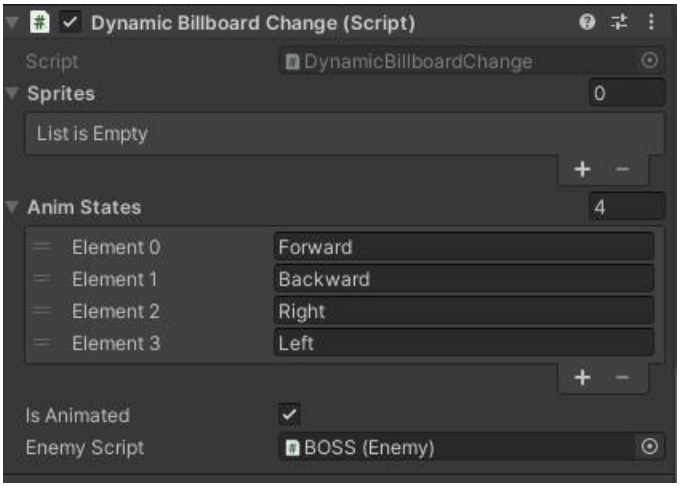
Patrol – Walking between waypoints

Attack – He sees player – he attacks him

Chase – He sees player and tries to catch him to start attacking

Alert – He heard the Player or seen him. Now he is waiting to Patrol or Chase.

All functionalities as seen on the photo could be changed in the Inspector of the particular Enemy.

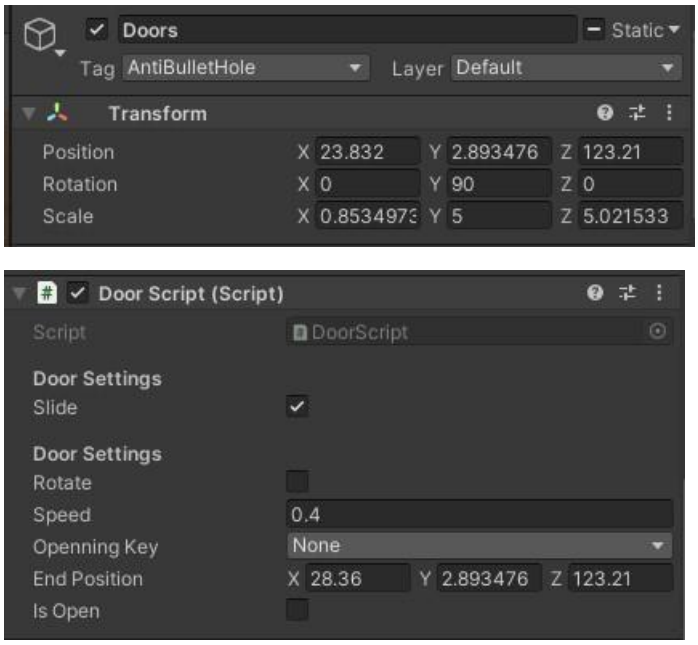


Billboard script allows us to change his graphics as the Enemy moves. If the enemy moves left – it changes its graphics to left rotation etc. All graphics should be made as shown below:

Then they are being put to Animator. For each movement (Backward, Forward, Left, Right) there is an animation created. So, this Animation then is being put to its Animator. States in the Animator should have the same name as those in the Billboard Script. Please check the project yourself so you could see it. Also feel free to watch the video where it's well-described.

Chapter 5 – Doors and Triggers

For now, I have added only one level trigger that is “Door”. Door contain one script that very simply allows you to make it move, animate or redirect you to the other level.



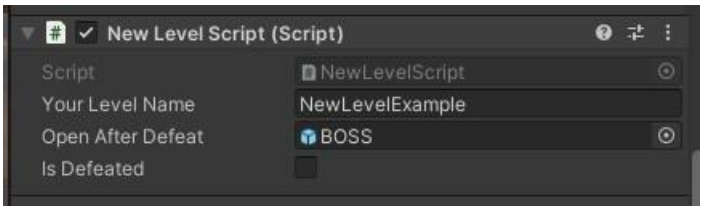
If you choose Slide, the door slides to the “End Position”. As you can see above, it’s starting X position is different than the End X Position. This is because, when Player is close to the door, door knows to slide from 23.832 to 28.36. This is how you could make all sliding in the game.

Speed is how fast the door opens.

Openning Key - This allows you to choose what Keyboard key player needs to click to open the door. Now it’s none – door opens when Player walks in. You can create that doors open only on the “E” keyboard press.

Chapter 6 – New Level / Defeating Boss

Doors can also contain “New Level Script”. This script is triggered when Player walks to the “Final Doors”. Script allows you to write Level Name of the next scene.



Also, script allows you to set “What Enemy” Player should defeat before allowing him to progress to the next level.

For now I have dragged from the Hierarchy the “Boss” Object. That means, Player can progress to the “NewLevelExample” scene, only after the “BOSS” is defeated. You can leave it empty if you want Player to progress without any execution.

Update v1.1: Main-Menu Settings

Main-Menu and in-game Settings can be found in **MainMenu.scene** under "**Canvas**".

All Settings Scripts and Options can be found under: **MainMenu_Controller object** with **MainMenuController.cs** script.

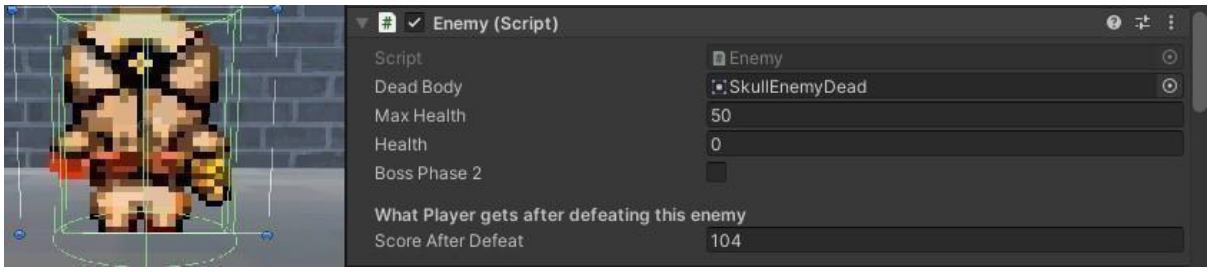
Update v1.2: Score and Leveling

Player contains “Score” parameter now that is being displayed among his health, ammo and armour information.

Score functionality can be found under two scripts:

Inside **Player** under **PlayerHealth.cs** Script & Inside **Enemy** under **Enemy.cs** script.

Player has new UI text that displays the Score on screen. This Score is being taken from the defeated enemies.



Enemy now contains new field under his Enemy script. “**Score After Defeat**”. You can set score separately to each enemy by changing the value “104” to your own. For example, “BOSS” can give you much more Score than normal Enemy.

If you want to change default score value, go to Enemy.cs script and change the value under scoreAfterDefeat float (as referred inside the script).

```
[Header("What Player gets after defeating this enemy")]
public float scoreAfterDefeat = 104f; //After defeating this monster, Player gets 104 score
bool scoreReceived = false;
```

There is also a “Check” that score has been given. Player can receive score only once and only if Enemy is “Defeated”.

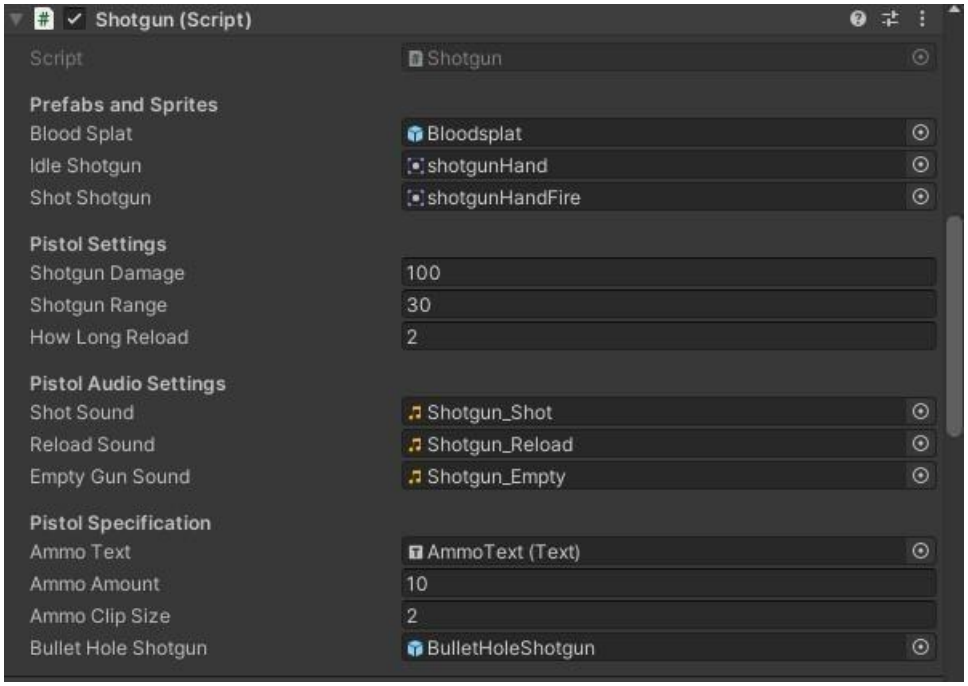
Important: PLAYER shouldn’t have parent object in Hierarchy. Refer to Enemy Script

Update v1.3: New Weapon SHOTGUN

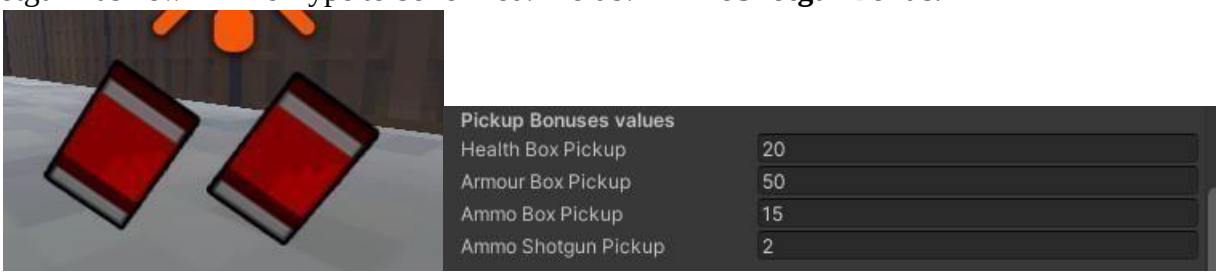
New weapon has been added. SHOTGUN shots 2 bullets at the same time and deals HIGH damage. Recommended use: Mini-Bosses. Its range is quite limited – better get closer to the enemy.

Shotgun has two states: Holding & Firing.

Shotgun has been added under Alpha Numeric “3” key in the Keyboard.



Shotgun has new Ammo Type to be refilled. Prefab: **AmmoShotgunBonus.**



It needs „**AmmoShotgunBonus**” **tag** under the object for Player to collect it properly.

Update v1.4: New Monsters

There are two monsters added to the game: Mini Zombie & Flying Dragon.

Mini Zombie: Walks fast & attacks only from very close distance.

Flying Dragon: Flies & Shoots with Fire. It's just an idea of how you can create your own monsters.

Monsters have no special powers by themselves because you can set them yourself within the Update v1.5 feature "Monster Attack Phases".

Update v1.5: Monsters Attack Phases

This update introduces several new functionalities, specially for AI. Now you can select special powers for each monster placed on your level. You can combine them, change every aspect of them and disable them at all.

Beginning: Each Monster can activate power when its health is below X value.

This value can be set by yourself in the Inspector within the Monsters Objects in Hierarchy under "**Enemy.cs**" script.

If you set 0 then it won't activate & if you set more than maxHealth of the enemy – it will activate straight away)

If enemy's health drops below this value (50 for example as seen above) then selected by yourself superpowers will be activated:

You can select (toggle) which power you would like this Monster to have after dropping the health to 50 value. You can also change the value of that superpower**. For example:**

If we choose "Monster Size" then we can select how much it should be scaled. In this example it multiplies the Monster Size by 2.

Health Regeneration works differently than other functionalities. If monster's health drops below 50 (as set in inspector) then it will start recovering its health for 2 seconds (Health Regeneration Time) by 2 health each second (Health Regeneration Amount).

There are tons and tons of Phases and superpowers you could create yourself. I am happy to make more of them in future updates. Feel free to give your ideas in the Unity FPS RETRO Thread or within the Reviews!

Coming Phases:

1) Changing sprite of the Monster (transformation).

2) Spawning more monsters when this one is being defeated.

3) Changing Enemy's missiles. So enemy shoots with explosions etc.

4) Separating Health to Activate the superpower for each superpower.

& and many many more!

CODE HAS BEEN WRITTEN SO YOU DON'T HAVE TO WRITE ANYTHING BUT TO BE SURE – THE CODE IS FULLY COMMENTED AND YOU WILL FIND AN EXAMPLE OF "HOW TO CREATE NEW PHASE".

Update v1.6: Level Select System & Save, Load System

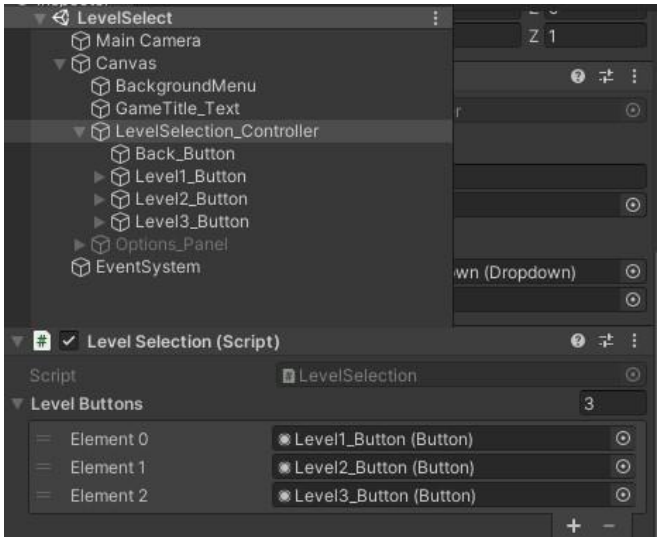
This update should introduce Saving and Loading the whole game and Player positions as introduced in newest games. However, before that – I would like to introduce Level Select System that is much closer to "smaller levels" like in Retro Games.

Now "**Start Game**" in **MainMenu.scene** loads "**LevelSelect.scene**".

LevelSelect.Scene contains levels your Players can unlock and play.

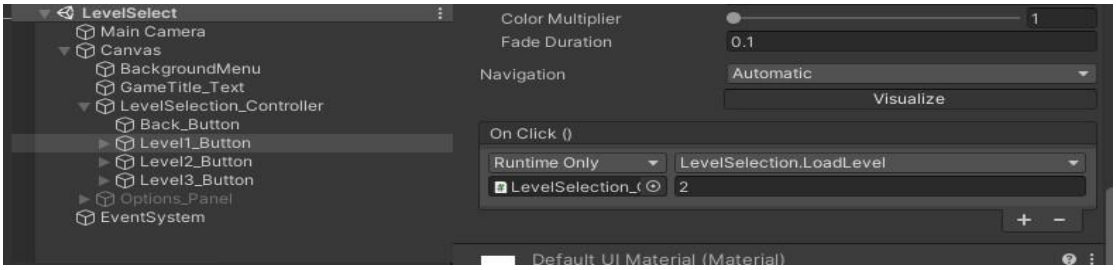
After completion of 1st level, Player is able to select level 2 in the same scene. It saves level completion as PlayerPrefs – so even after closing the game.

In the Hierarchy you have LevelSelection_Controller object that contains all buttons you might like to create:

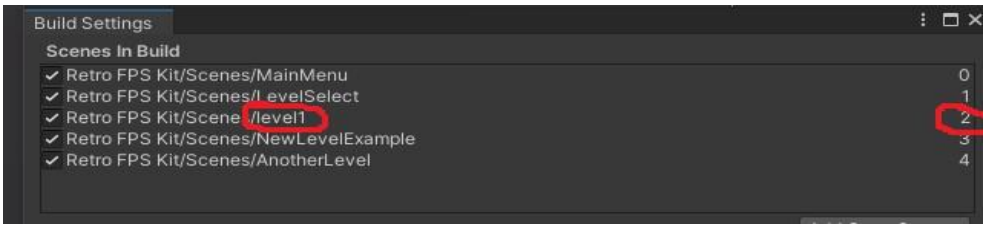


You can add as many buttons as you might like. Each Button represents the level.

Now in each Button you might need to set 1 thing: What level to load (from Build Setting Index):



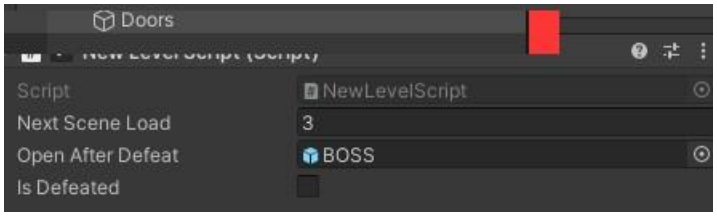
This example shows Level1_Button that LoadsLevel under Build Index 2 as below:



Do the same for each new level you'd like to load.

LEVEL COMPLETION TRIGGER:

We have added the "Level Completion" System to the last door in each level. Door that are being activated after defeating "THE BOSS" are now saving information, that player had passed this level.



NextSceneLoad is the Index to load the scene from Build Settings (exactly as with buttons above). That means, Next Scene Load: 3 (as above) loads "NewLevelExample.scene" after defeating the Boss.

Also, if player leaves the Game – now the Level 2 is unlocked in the LevelSelection.scene.

To reset Saved Data go to: In unity editor: Window -> DeleteAll PlayerPrefs.