Thank you for purchasing the Fusion Shooter Brawler asset. Let's make a competitive multiplayer game!

If you need help contact me at vauxlandsupp@gmail.com

Discord coming soon!

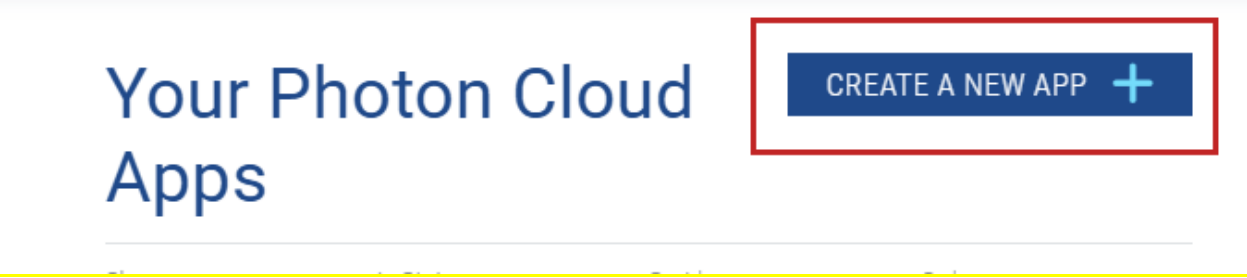
CONTENTS

SETTING UP

If you haven't already, download and install the Fusion 2 Asset from the asset store Photon Fusion | Network | Unity Asset Store

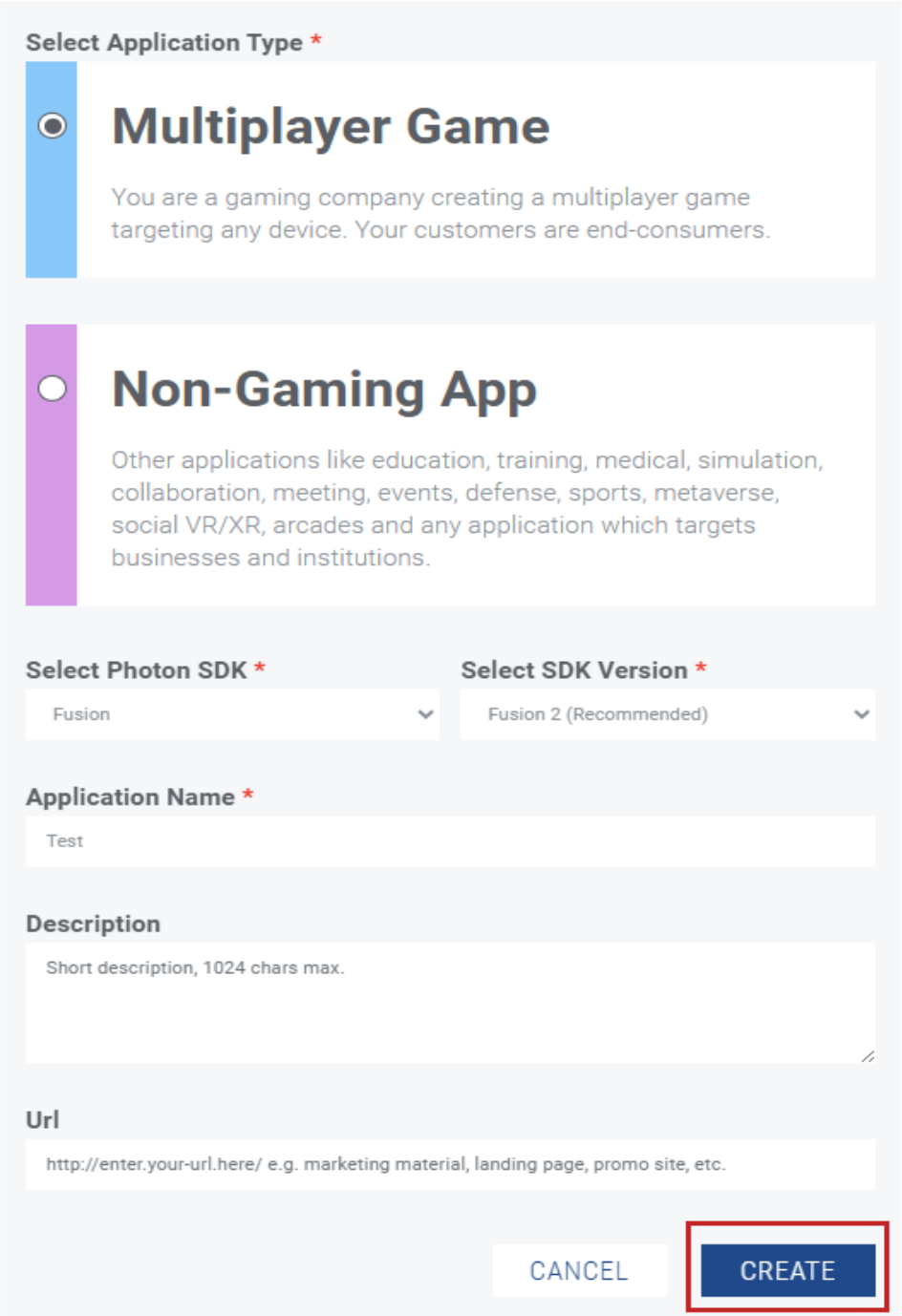
After its done installing you will need to set up your fusion app ID. If you don't already go to Your Photon Cloud | Photon Engine to set up an account and sign in. Or click the **Open the Photon Dashboard** button in the fusion hub.

On your dashboard click 'Create a new App'



Select "Multiplayer Game" and Sdk: Fusion then Sdk Version: Fusion 2

Call it whatever you want then click create.



Then right click on your app Id and copy.

Go back to your unity project and paste your app ID into the **Fusion 2 Setup Fusion App ID** section. If you don't have it pulled up its under Tools > Fusion > **Fusion Hub** then click Fusion 2 Setup

After you paste your ID you should get a little green checkmark that means you're all set and can close the fusion hub.

You will need to download and import Photons Physics addon from this link:

[Fusion 2 Download | Photon Engine]

ASSET FOLDER STRUCTURE

Alright now that we're set up let's unpack this asset. Open –

Assets > Vauxland > FusionShooterBrawler.

You will see 5 Folders: **Demo Resources, Game Data Configs, Prefabs, Scenes and Scripts.**

Demo Resources just contains all of the textures, materials, models and font used to make the demo scenes.

Game Data Configs contains 4 folders: Character Configs, Powerup Configs, Stat Effects and Weapon Configs.

This is where you can look at the current characters, weapons, stat effects and powerups and add more if you'd like by duplicating them and changing their configs. More on that in their respective sections

Prefabs folder contains the network runner prefab, the player prefab, pick up prefabs (which are power ups and weapon pick ups) projectile prefabs, for all of the different projectiles for the weapons, and Ui prefabs which contain prefabs such as the lobby menus, the match UI all of the instantiated UI pieces like the kill feed pop up and scoreboard entries. When you change or add your own UI if you get lost or mess something up you can revert back to these base prefabs of the demo.

Scenes contains the current scenes in the demo, Lobby Scene and GameMatchScene. You can duplicate these scenes and change the game rules and then add to build settings before you play. More explained in **Making a New Match Scene** section.

Scripts folder, you guessed it, contains all the games scripts in the asset.

Gameplay scripts: contains all the scripts that run in the match scenes.

Lobby scripts: contains all the scripts that run on the main menu in the lobby scene aside from the networked ones.

Network scripts: contains all of the core network scripts.

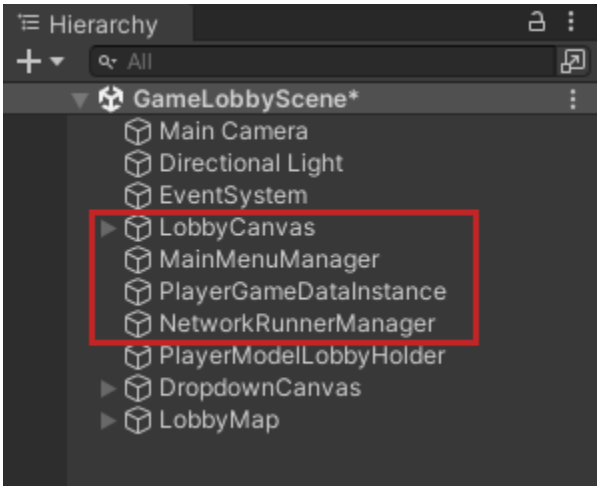
Player scripts:: contains all of the scripts that are on the player object.

Utility scripts contains scripts like the scriptable object scripts and joystick script etc.

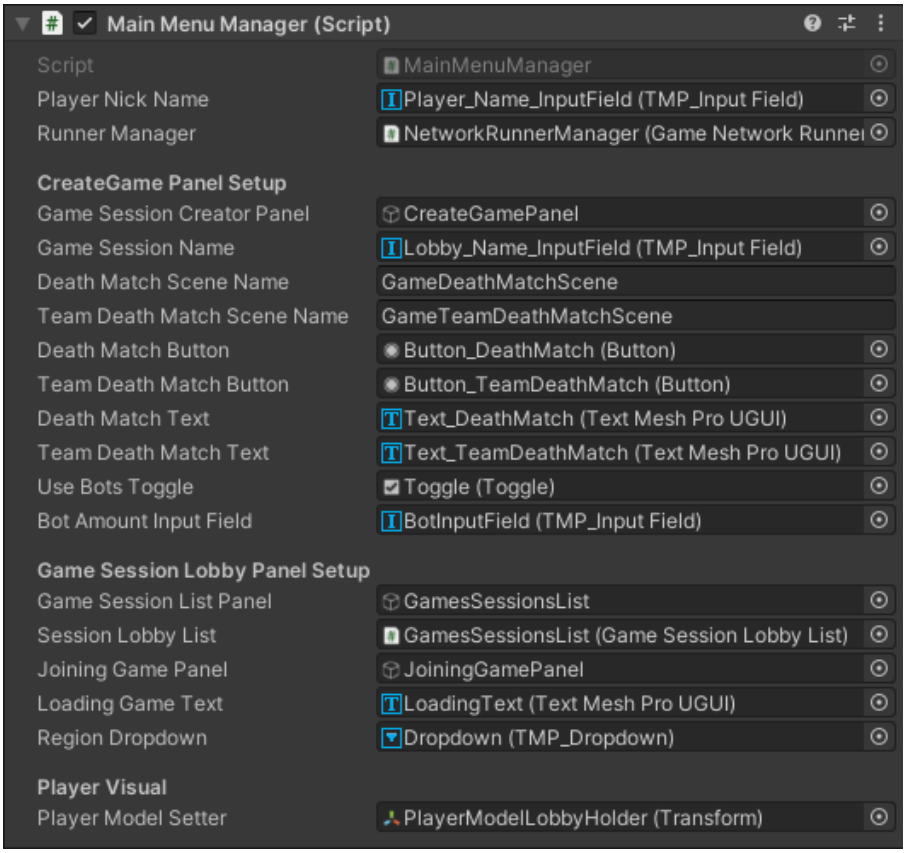
LOBBY SCENE

Load up the **Lobby Scene** under Assets > Vauxland > FusionShooterBrawler > Scenes

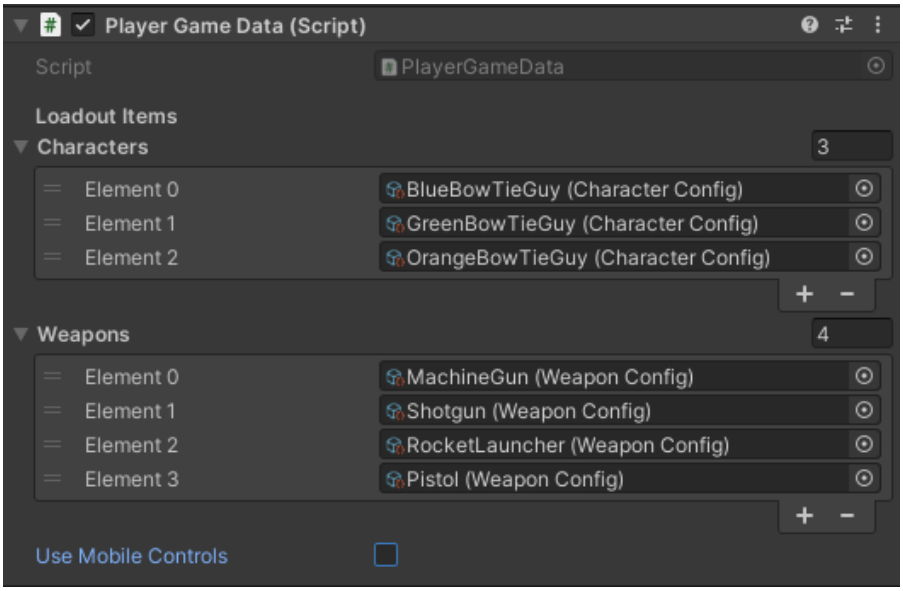
The Lobby scene consists of the **main menu UI lobby canvas**, the **main menu manager** , the **player game data instance** and the **network runner manager.**



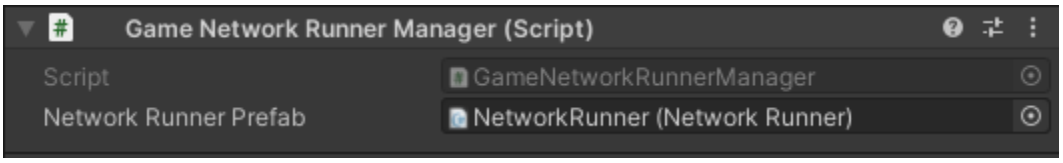
The **MainMenuManager** is where you will set all of your corresponding lobby canvas UI elements to the required fields in the main menu manager object inspector such as the **Create Game Panel,** which is set active when you click **create game** button to create your own match, and other UI elements of the canvas.



PlayerGameDataInstance is where we will be setting up any new weapons, characters and game data that the player uses in the matches and adding them to the **Player Game Data Instance inspectors** respective fields.



The **NetworkRunnerManager** object has the script **GameNetworkRunnerManager** attached to it and this handles spawning our runner (which is the heartbeat behind fusion simulation), creating and joining game sessions over the fusion network. This is static and carries over into every scene.



MAKING A NEW WEAPON

Lets make a new weapon.

Go to Vauxland > FusionShooterBrawler > GameDataConfigs > WeaponConfigs > Models.

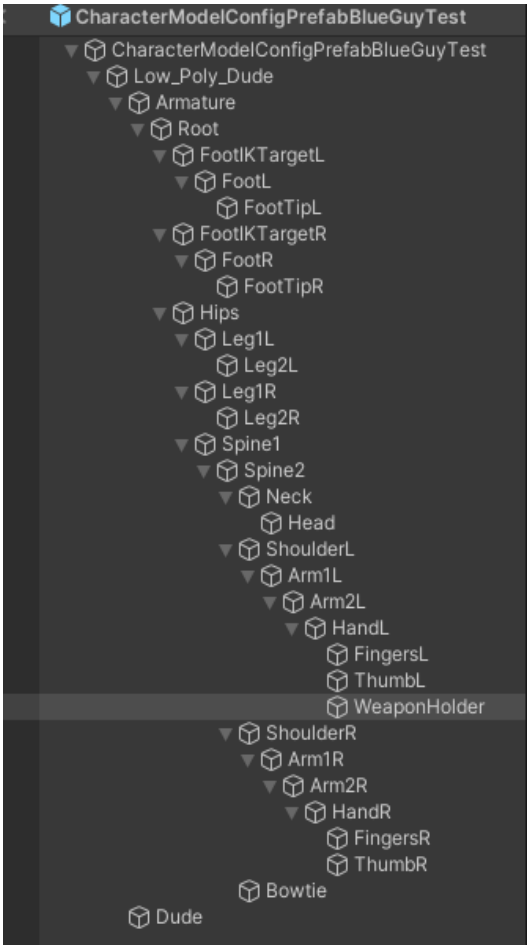
Here you will see the model prefabs of the weapons in the demo.

You can duplicate one and replace the mesh with one of your own models and then use that to set up the proper hand placement easier than creating one from scratch but lets make one from scratch.

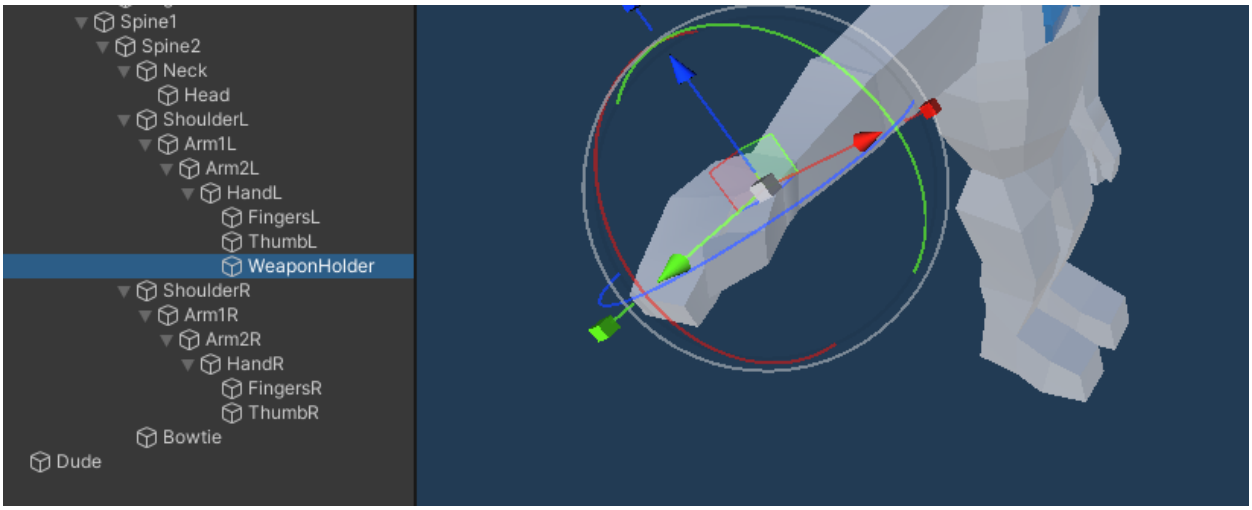
To set up your weapon model, place your rigged character model or one of the models in Vauxland > FusionShooterBrawler > GameDataConfigs > CharacterConfigs > Models. Just drag and drop one into the scene view or just open its prefab.

Once you have your character model out lets set up your Weapon Holder.

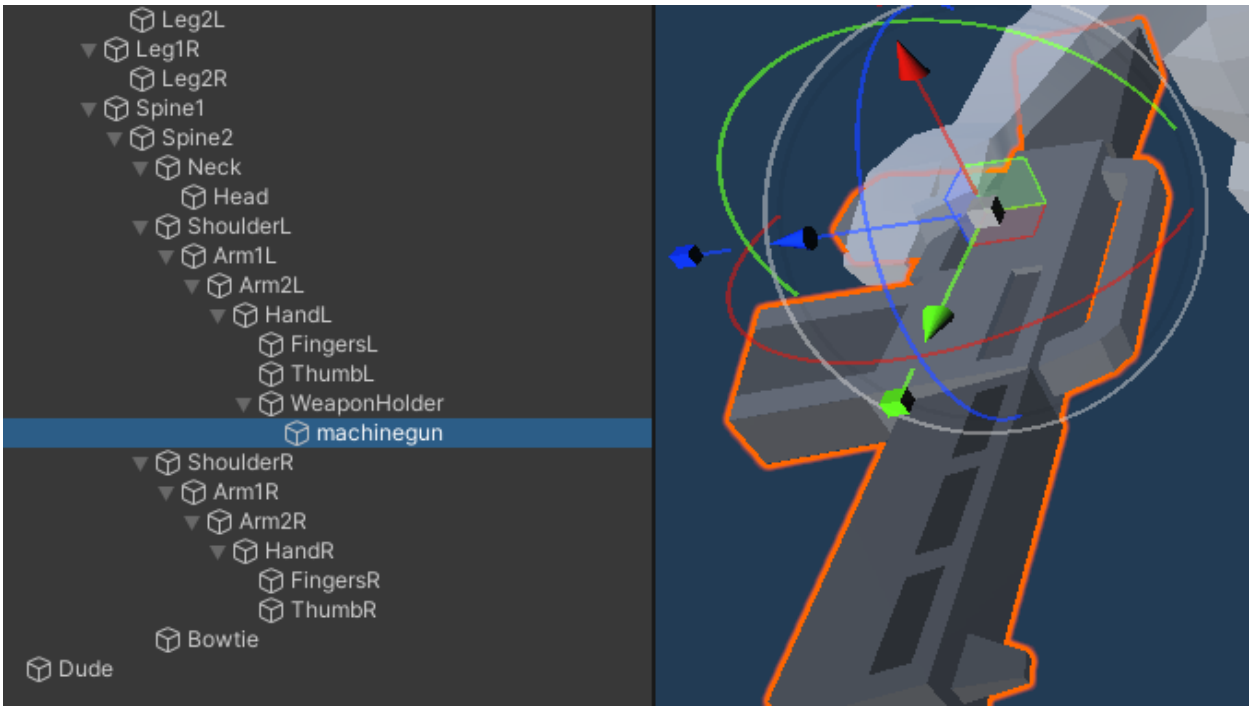
Drop down the armature of your character and create and empty game object on the right or left hand and call it “Weapon Holder” or whatever you want like so:



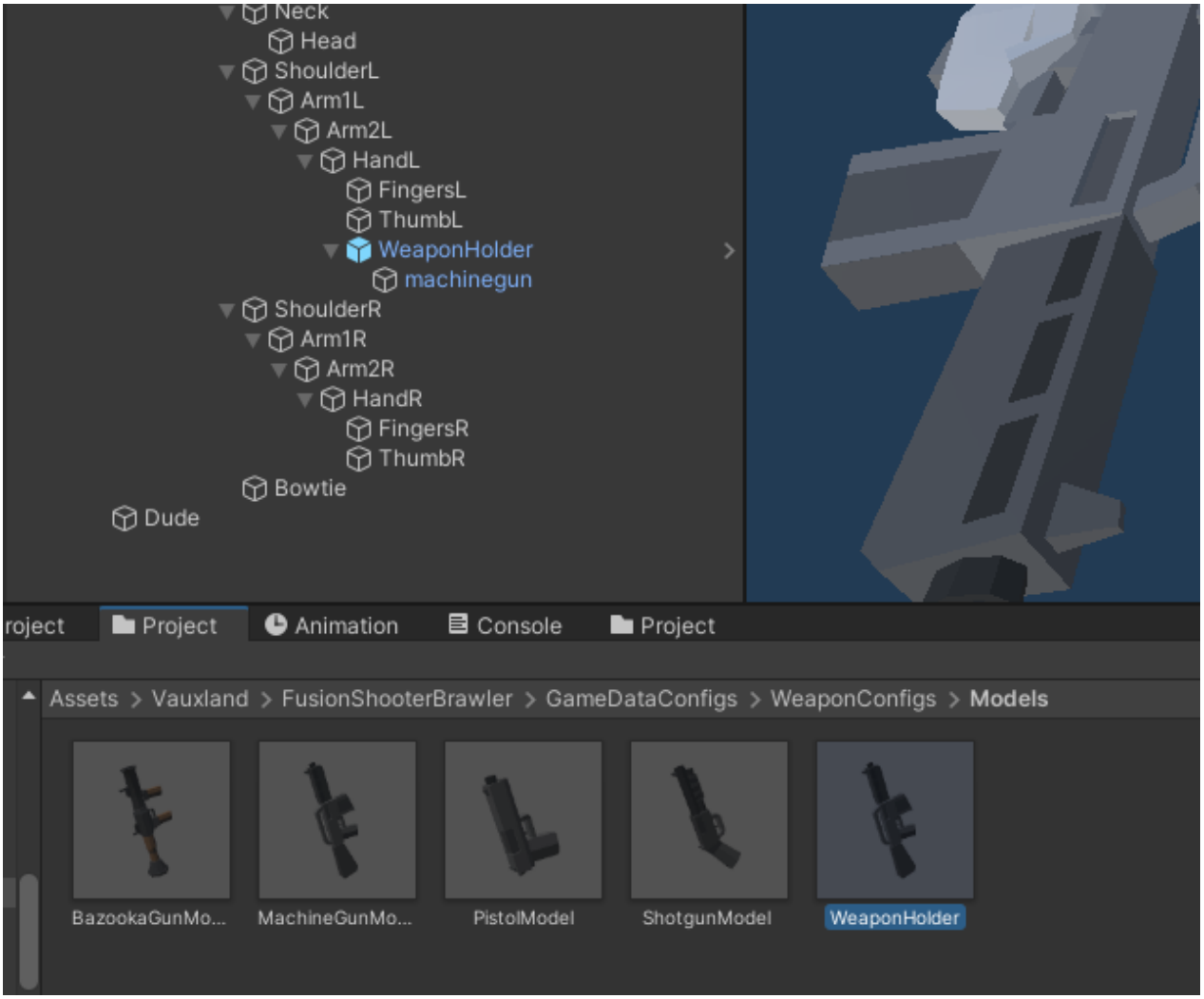
Position the weapon holder on to the hand, the rotation doesn't really matter on this part just have the position in the general area of the hand like so:



It’s best to leave the weapon holder transform at 0, 0, 0. Now drag and drop your weapon model onto the WeaponHolder. Adjust and position your weapon model as you see fit, do not adjust the weapon holder transform, only the weapon model itself.

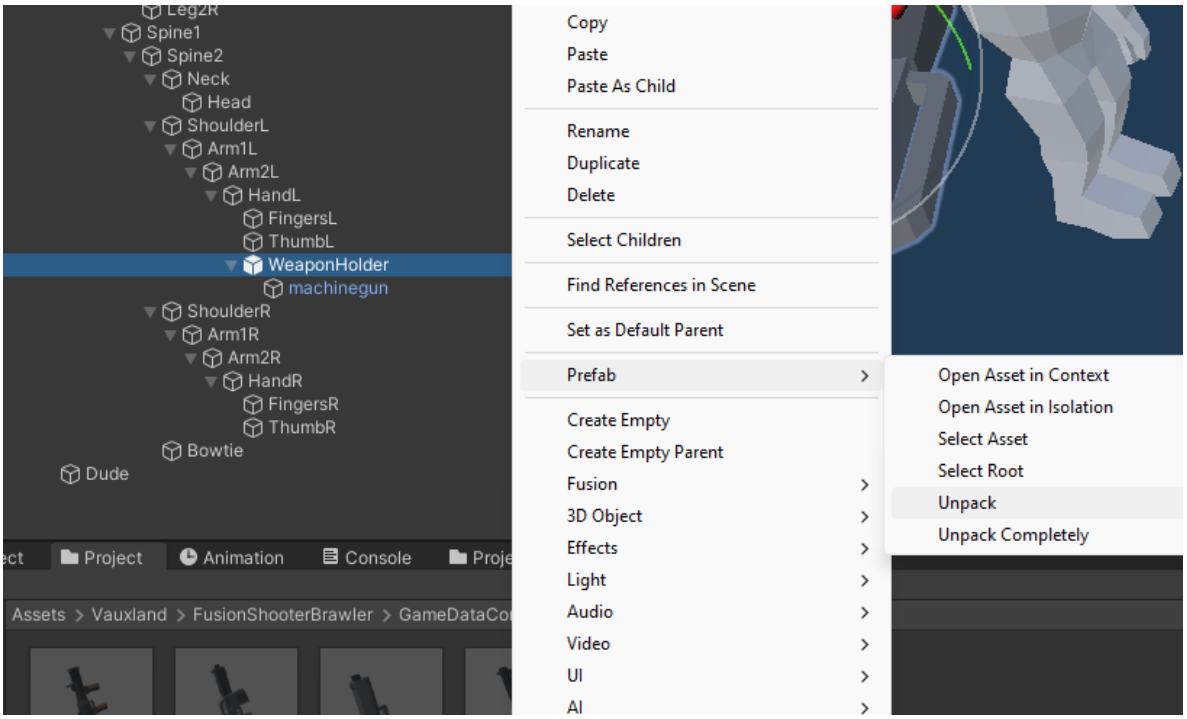


Once you’re satisfied click on the WeaponHolder transform and drag and drop it into your assets folder to create a prefab of the model

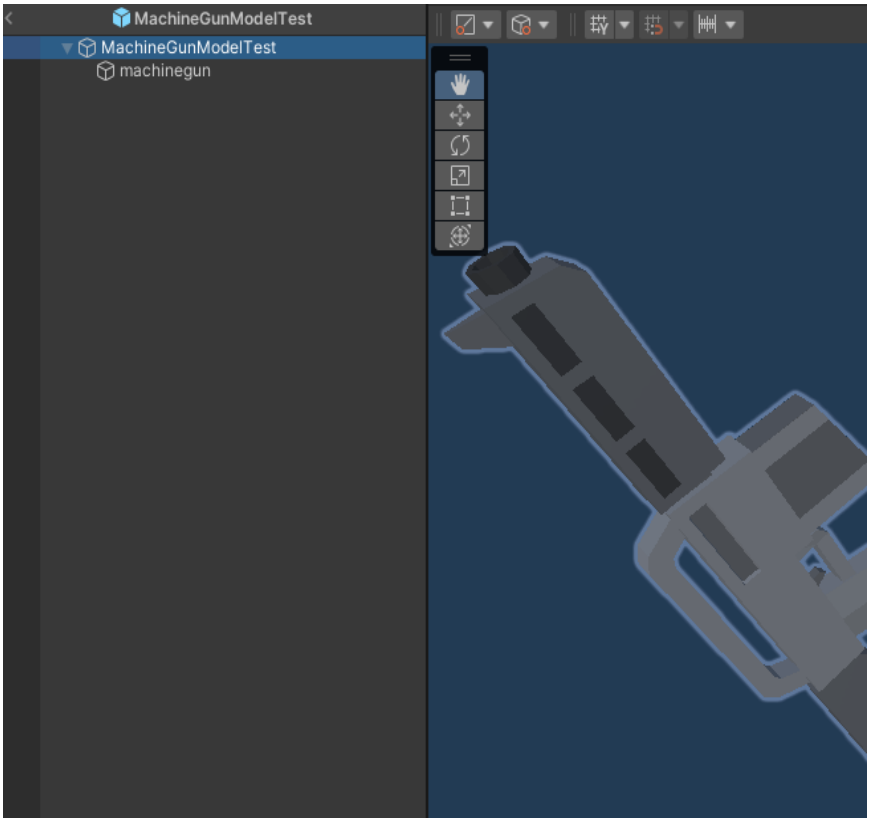


Rename the prefab from “WeaponHolder” to whatever you want.

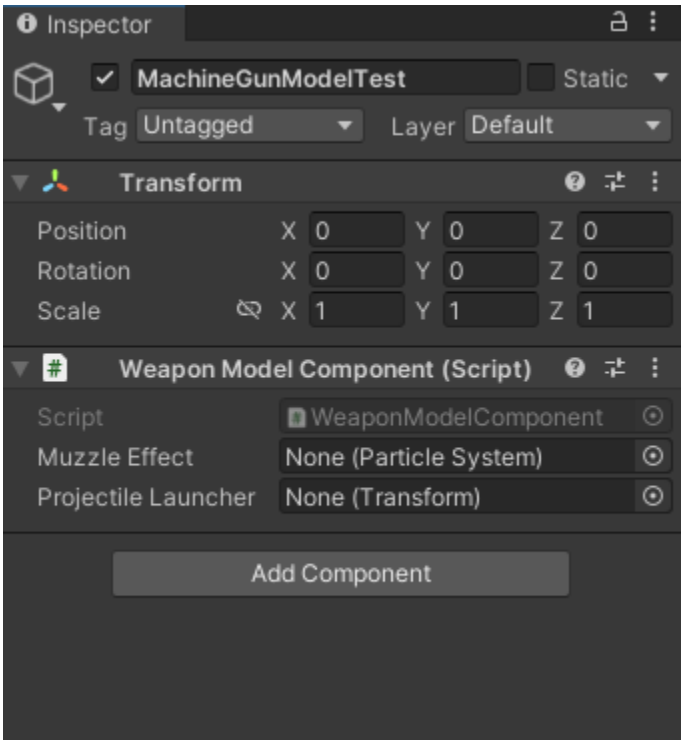
Next right click and unpack the prefab on the character model that is now highlighted blue because you just created a prefab, because you will use this method again for all your new weapon models.



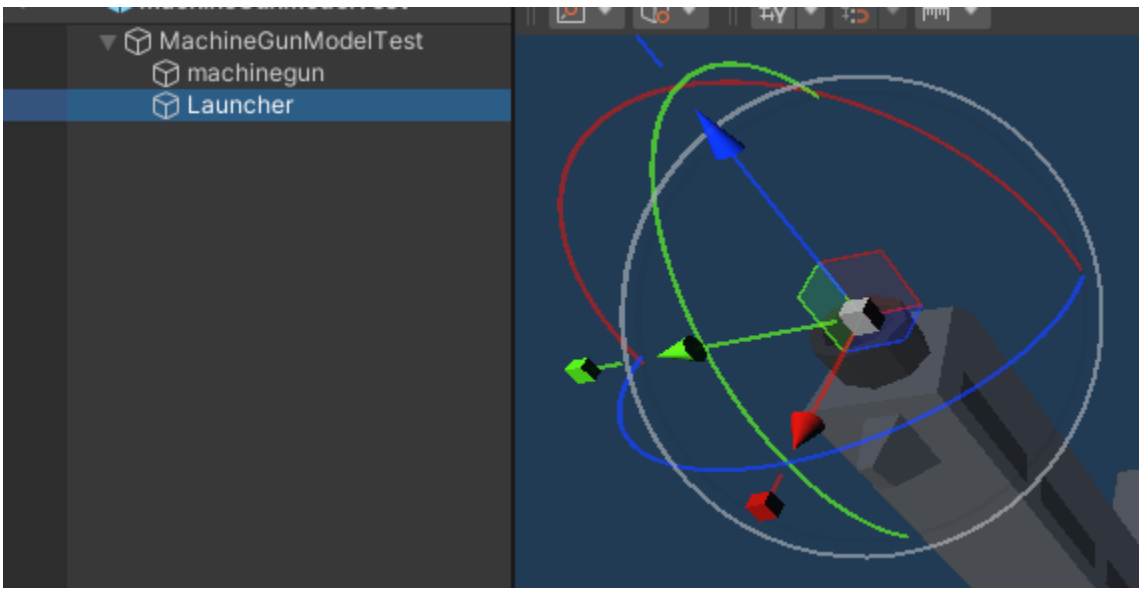
Next delete the weapon model child from your WeaponHolder transform on your character model. Now double click on your on your newly created weapon model prefab in assets folder and double click open up the prefab:



In the inspector of your prefab click add component and add the script: “Weapon Model Component”.

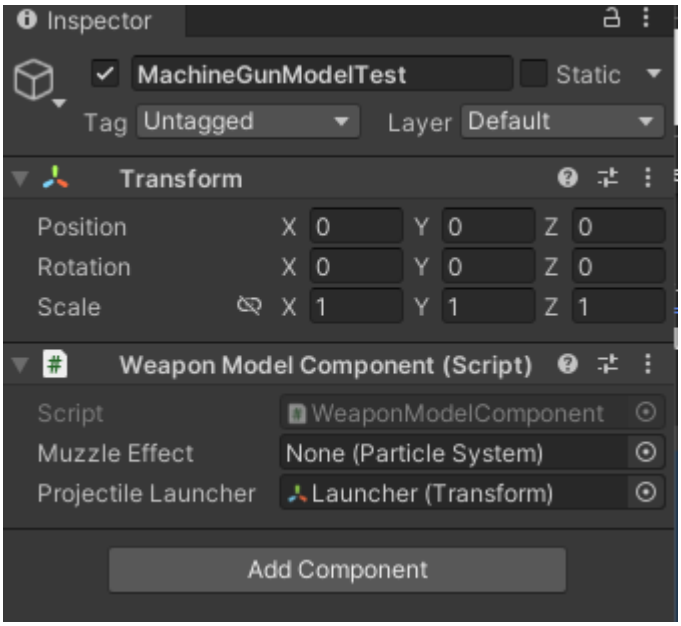


Now add an empty transform as a child object of your prefab and call it “Launcher” or whatever you want and position it at the tip of the barrel of your gun:



Make sure the blue arrow (z-axis, forward) is facing straight out from the barrel like in the example image.

Drag and drop the launcher reference to the “**Projectile Launcher**” field in the inspector.



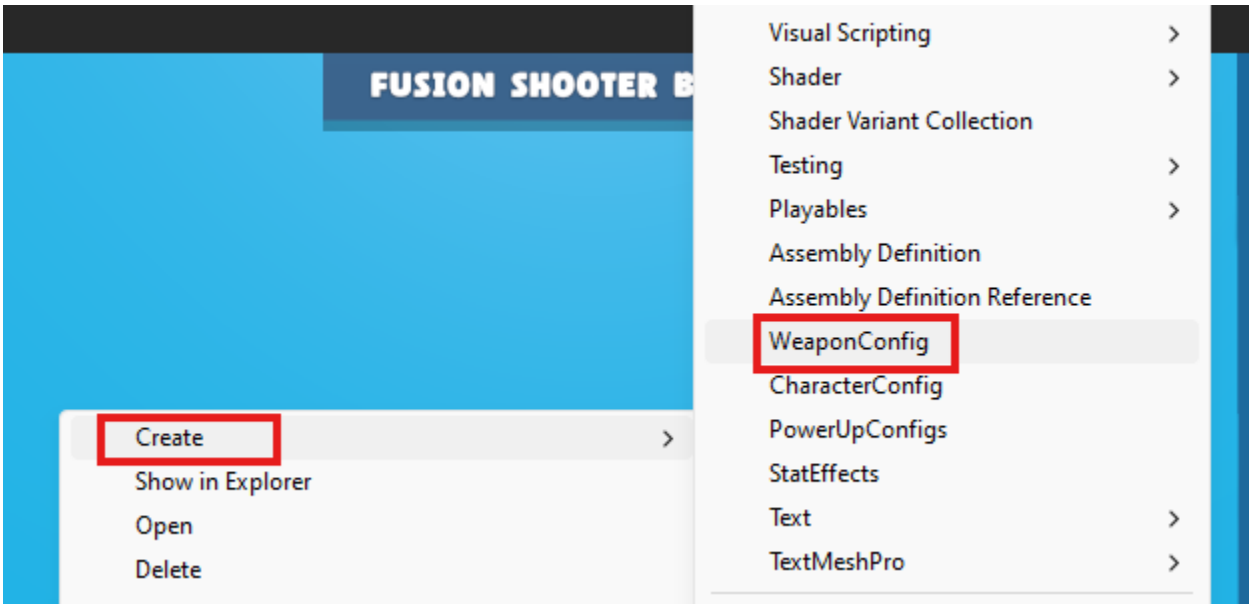
Next just add a muzzle effect in the same position as the launcher or where ever you want it and add it to the “**Muzzle Effect**” field in the inspector.

Now you’re done setting up the model. Repeat that same process for all new weapon models. You can edit the weapon model component script to have new features as well and then reference those changes from the **Player Visuals Controller** script which gets the weapon model component from the players weapon when setting up the weapon in a match.

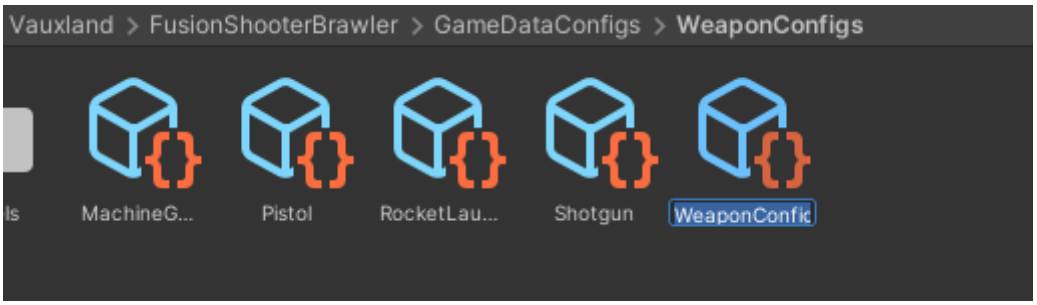
Now lets create the new WeaponConfig scriptable object for our new weapon model we just made. The weapon config stores all of our weapons stats and data configuration used in a match.

Go to Assets > Vauxland > FusionShooterBrawler > GameDataConfigs > WeaponConfigs.

Right click in an empty space in this folder and click create > FusionBrawler > WeaponConfig



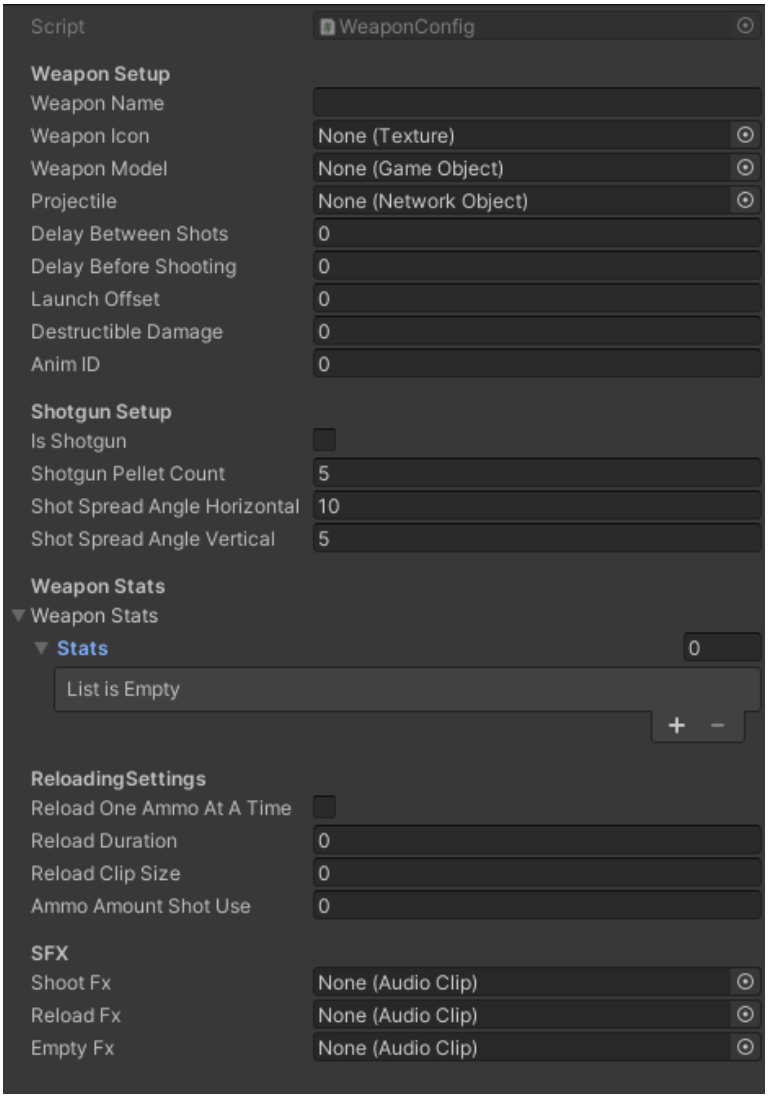
This will create a new Weapon Config asset.



Now call it whatever you'd like and let's look at its inspector:

The **Weapon Name** is the name of the weapon that the Loadout UI will display in the available weapons to choose from.

Weapon Icon is the little icon of the weapon that will appear in the kill feed popups and the currently equipped weapon UI image in the match scene next to the ammo amount text.



Weapon Model the model prefab we just created goes here, drag and drop the weapon model prefab you just made here.

Projectile is the custom projectile this weapon will spawn when shooting we will make that after finishing this weapon set up.

Delay Between Shots is the fire rate basically. How long does it take for the gun to shoot each shot. How fast will this gun pump out projectiles.

Delay Before Shooting is how long it takes before the gun starts shooting after you press the shoot button, adjust this to time better with your animation for you gun. For example the Rocket Launcher takes 1 second before it shoots a round. Because it takes time to position and aim a weapon like that.

Launch Offset is the distance from the projectile launcher the projectile will launch/spawn from, you can use a negative value which will shoot from closer behind the projectile launcher transform or a positive value which shoots further away from the projectile launcher.

Destructible Damage is how much damage the weapons projectile will do to a destructible such as an exploding barrel or destructible block in the match scene. Anything you make as a destructible.

Anim ID is the ID for the animation to play for that weapon when you add the animation to the animator which I will explain after finishing this setup.

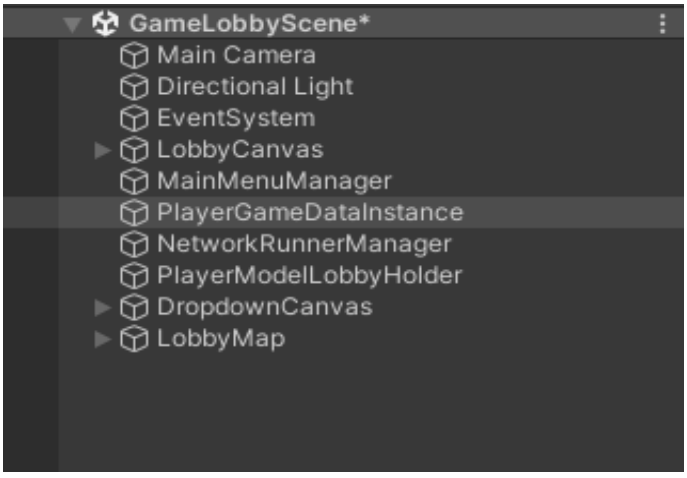
Shotgun Setup is for if you want this weapon to shoot like a shotgun. So a spread shot.

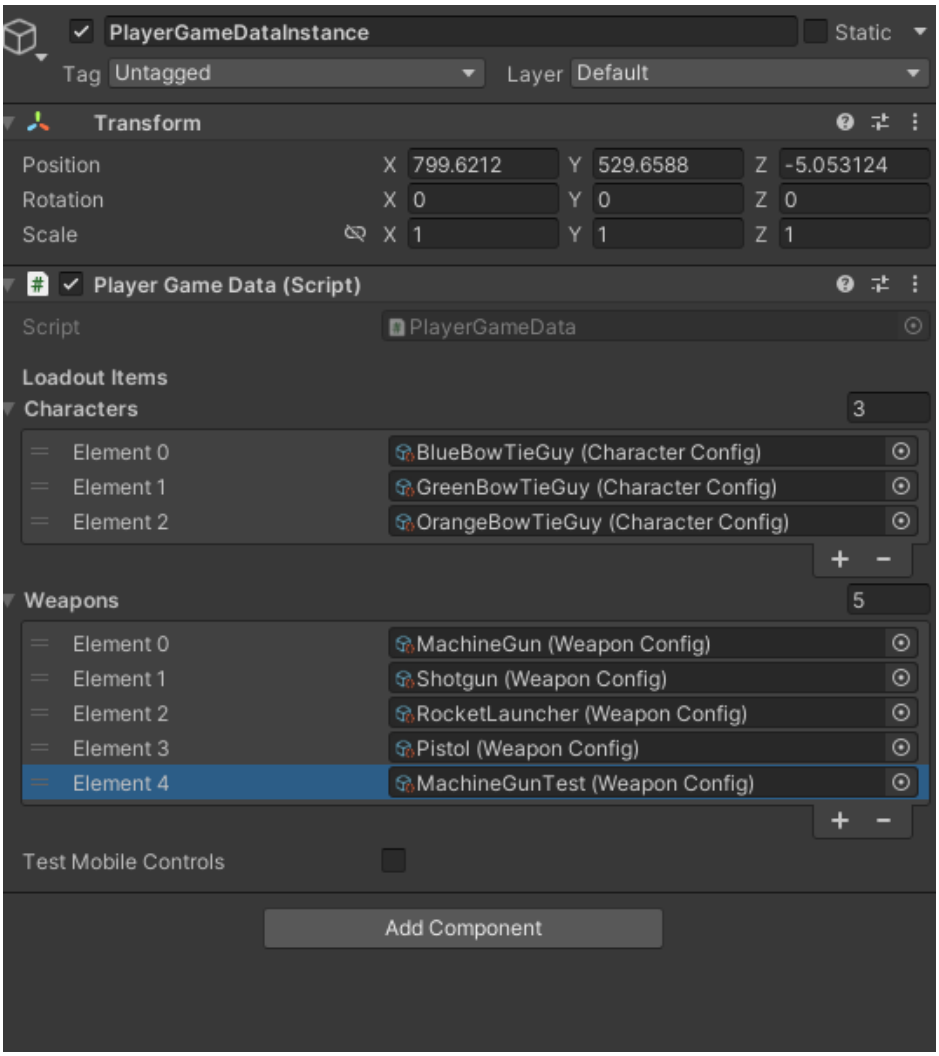
Weapon Stats here is where you will set the stats of the weapon. Any stat you'd like. Attack Damage, HP, Move speed, etc.

Reload Settings all of your settings for reloading the weapon, should one bullet be reloaded at a time? Reload duration is how long it takes to reload the weapon, Reload clip size is the amount of ammo that gets reloaded in a reload. Ammo amount shot use is how much of the ammo is used in one shot.

SFX the sounds you want to play when shooting, reloading or out of ammo;

Once you've set up all of the weapon config's data, go to the **PlayerGameDataInstance's** inspector in the lobby scene and drag and drop the new weapon config data you've just created into the weapon configs dropdown section.





Then that's it aside from adding the animation. Follow these same steps for each new weapon you add.

Now lets add the animation and set up the weapon loadout UI.

Click on your animator component or create a new animator component.

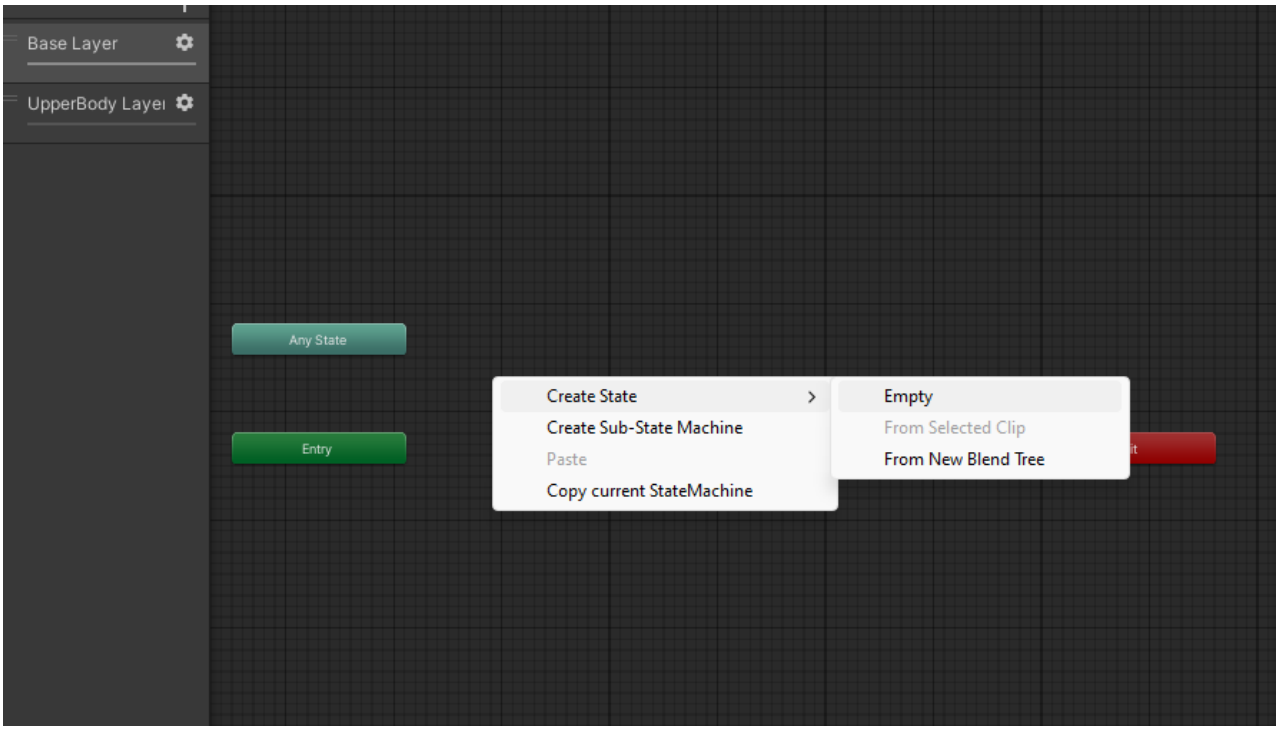
You can look at the current characters animator controller in the demo for reference on how to set up any new animation. The animator controller will have 2 layers, **Base Layer** and **UpperBody Layer**. The base layer handles the animations for the lower half of the characters rig and handles Idle, Running and shooting transition from Idle to Running.

The **UpperBody Layer** handles the top half of the character rig for running and shooting, while the base layer keeps the legs playing the running animation the upper body layer is also playing the shooting animation.

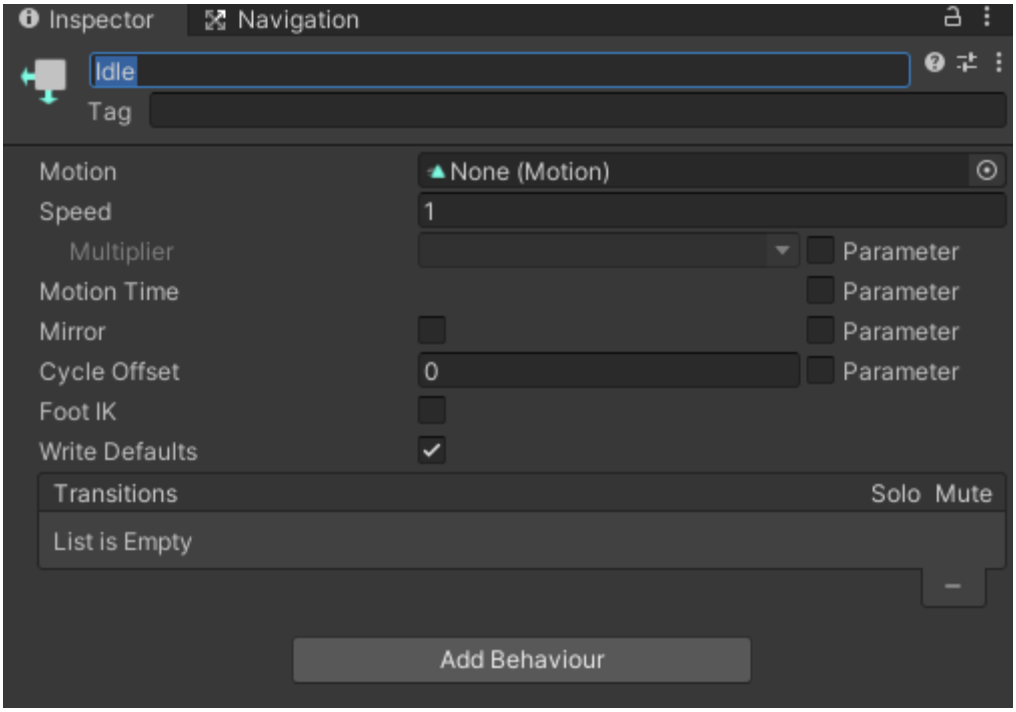
To add a new weapon animation we will need to add it to the base layer and upper body layer.

If you don’t have an idle animation already first add that.

Right click in the base layer animator controller and click Create State > Empty.

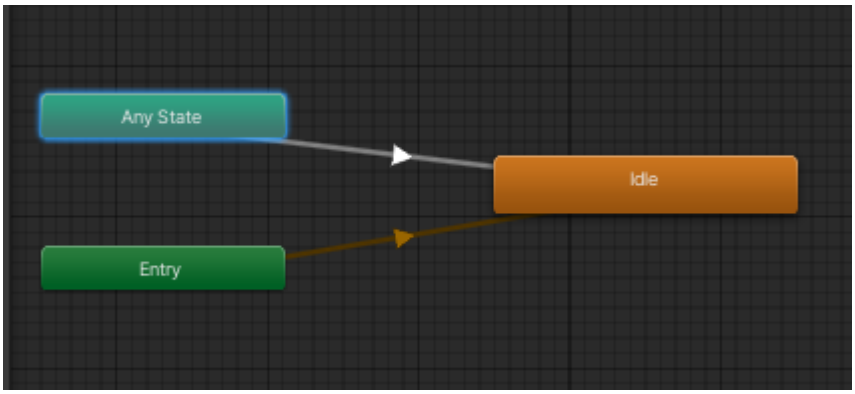


Click on your newly created state and rename it to “Idle”



Add your Idle animation in the Motion Field.

Next right click on the Any State node and click make transition and drag to the Idle state node.



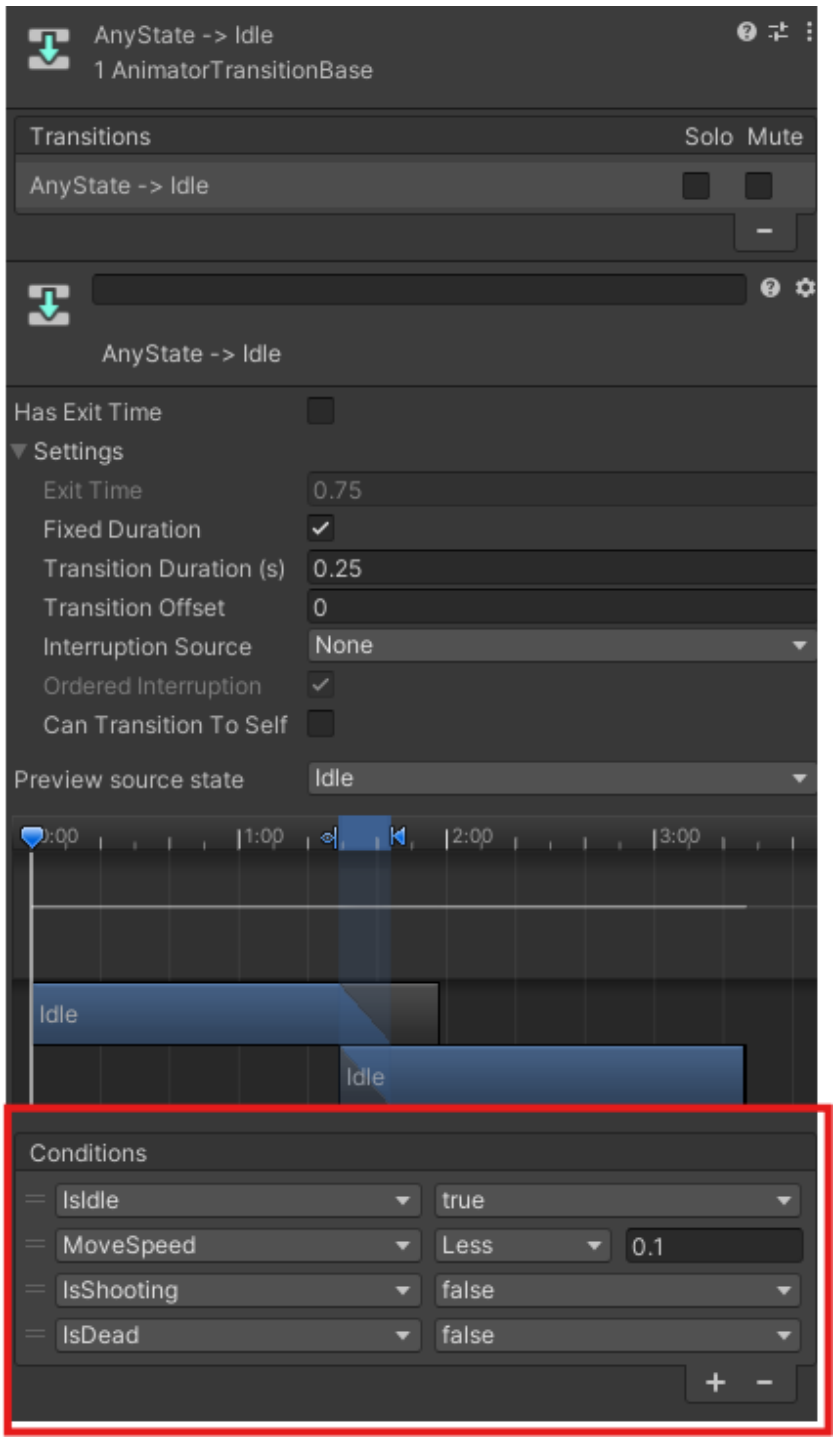
Now Go to Parameters Tab, click the “**+**” button add these parameters:

Move Speed is a float.

IsDead, IsShooting and **IsIdle** are bools.

AnimID is an int.

Next click on the new transition line you made a second ago and in the inspector, uncheck “Can Transition To Self” and add these parameter conditions:



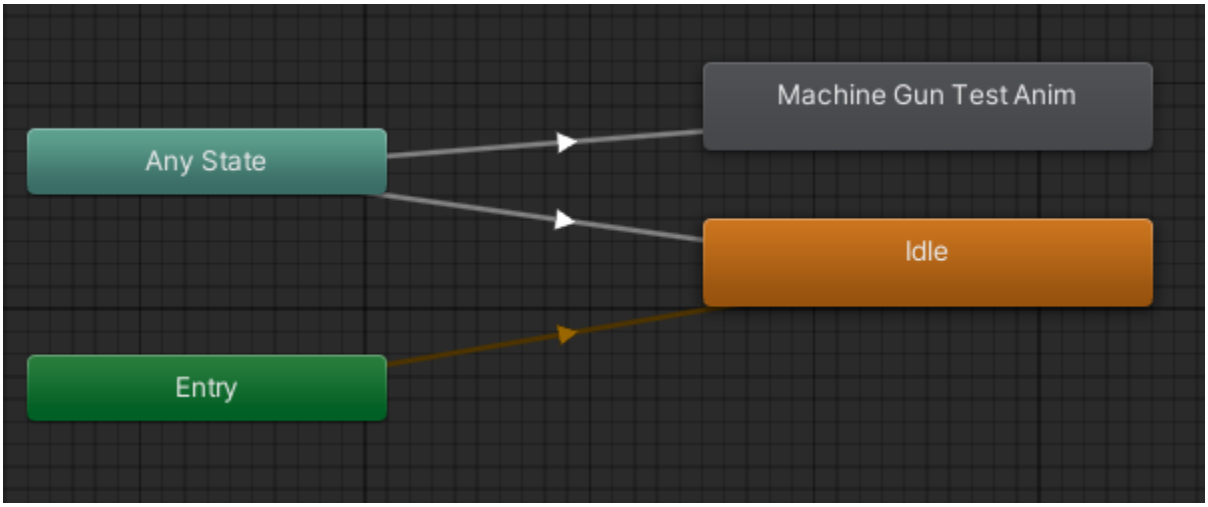
IsIdle to true, **MoveSpeed** to less than 0.1, **IsShooting** to false, and **IsDead** to false.

Now lets add the weapons shooting Animation.

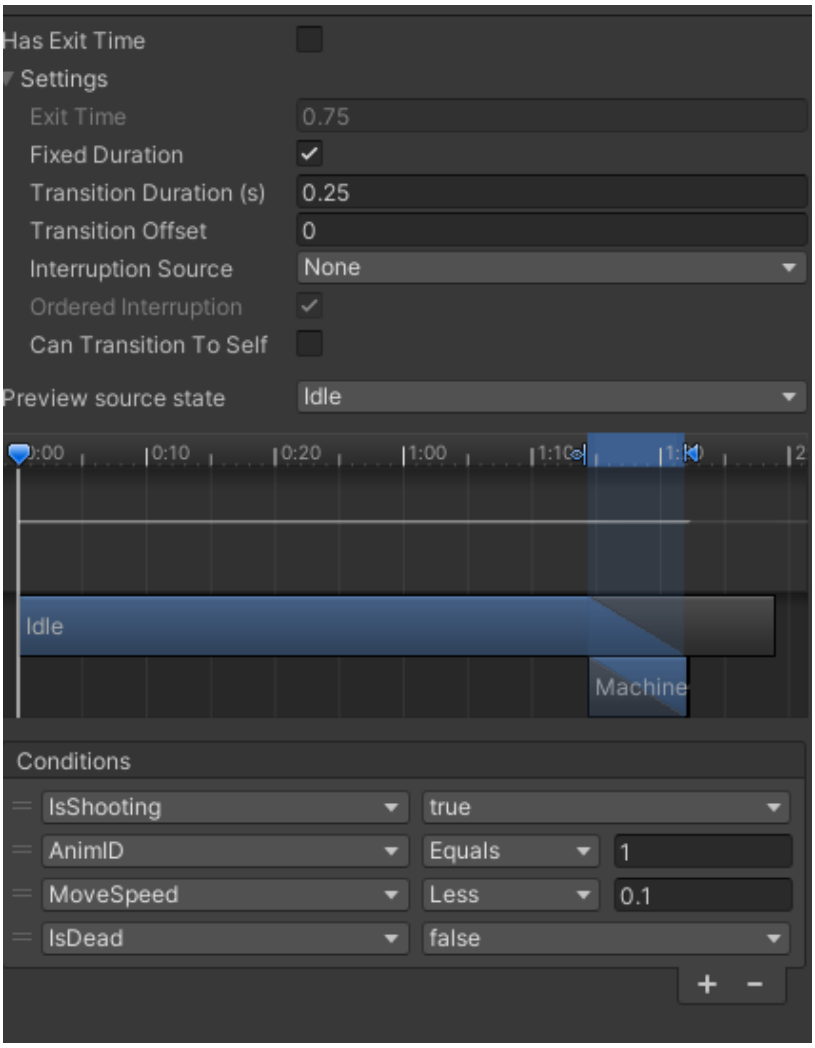
Same as before right click in the base layer, Create State > Empty.

Rename the state to the name of the Weapon Config we created earlier or whatever you'd like and add your shooting animation in the inspector.

Now right click on any state and make a transition from any state to the new weapon anim state.



Now same as before click on the new transition line and add these parameter conditions and uncheck Can Transition To Self:



IsShooting to true, **AnimID** equals the value that you set in the **Weapon Configs** animID value earlier. This must be that number you set in the weapon config or else when you shoot that weapon it wont play this animation.

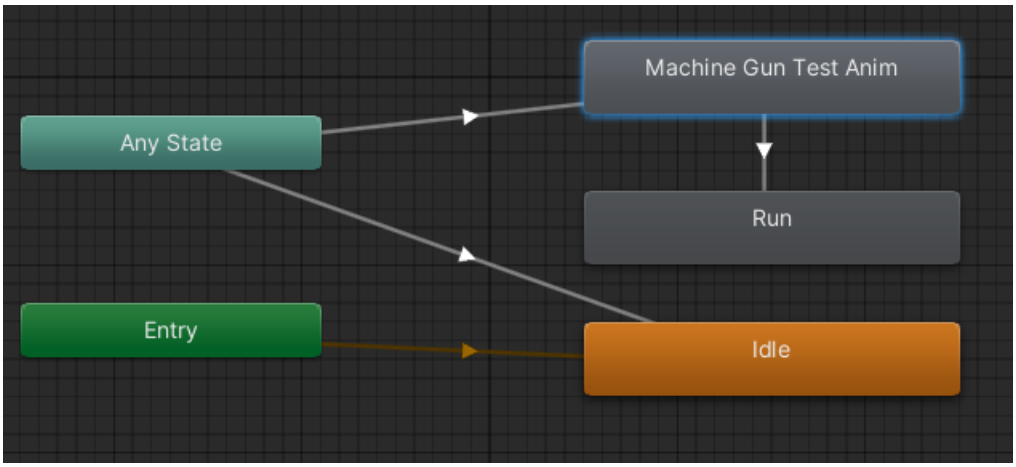
MoveSpeed set to less than 0.1 because we're idle when playing this shooting animation.

IsDead to false. Because we can't shoot if we are dead.

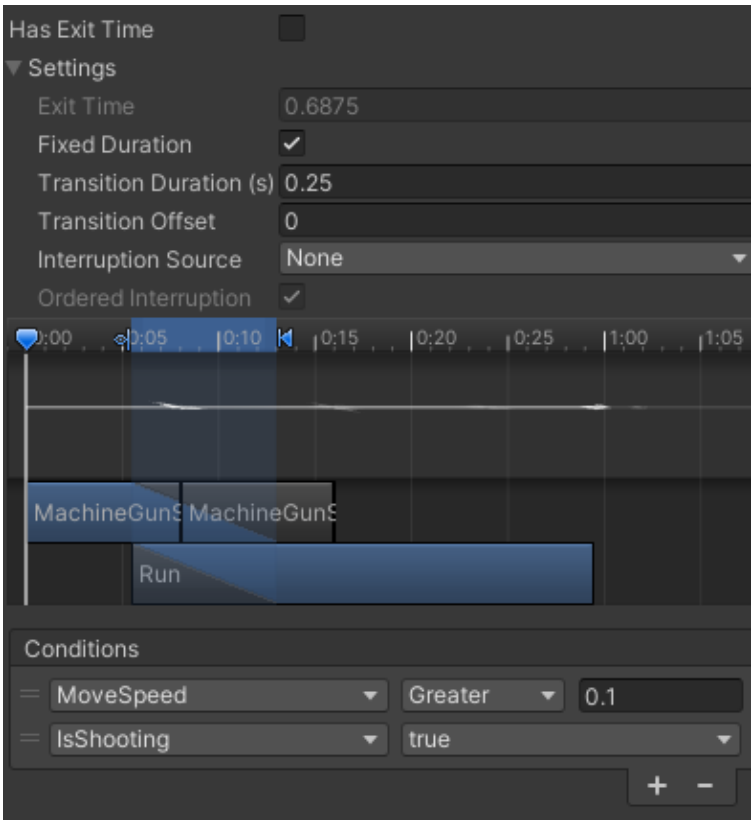
Now we need to add a running animation because we will need to transition this new weapon anim to the running anim.

Again right click in base layer and create a new empty state. Rename it to Run. Add your running animation.

Now right click on the weapon animation state you made and make a transition to the run state.



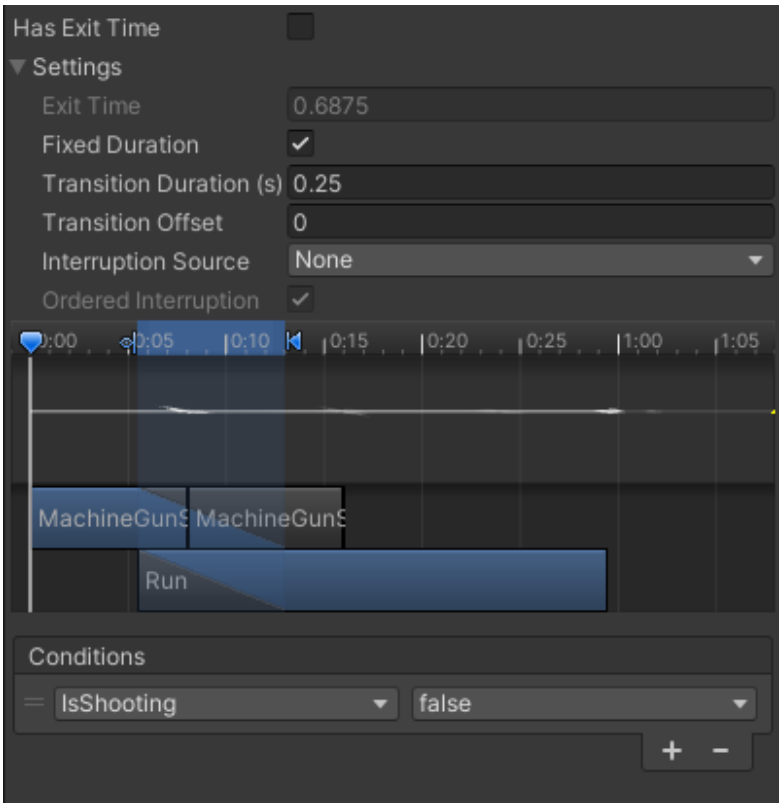
Now click on the transition line and uncheck “Has Exit Time” and add these parameters:



MoveSpeed set is Greater than 0.1, and **IsShooting** to true.

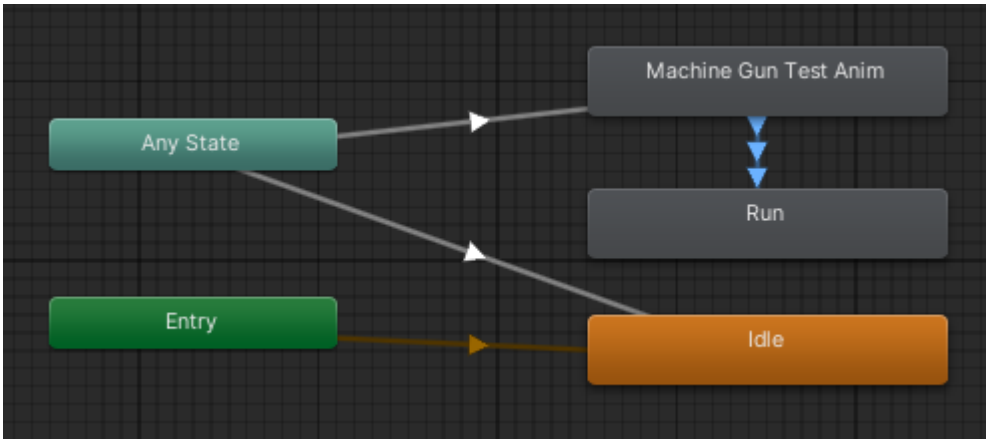
Now Create another transition from the weapon anim state to running.

Click on that second transition line and same as before uncheck Has Exit Time and add these parameters:



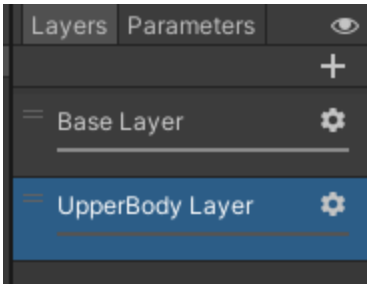
IsShooting set to false.

Your animator should look like this currently:



Now we're done with setting the weapon anim on the base layer. Follow the demos provided animator see the other transitions from Run to Idle and Idle to Run and the Dead anim state.

For now click on the UpperBody Layer or if you haven't added it yet click the "**+**" button and add a layer and name it UpperBody or whatever you want.

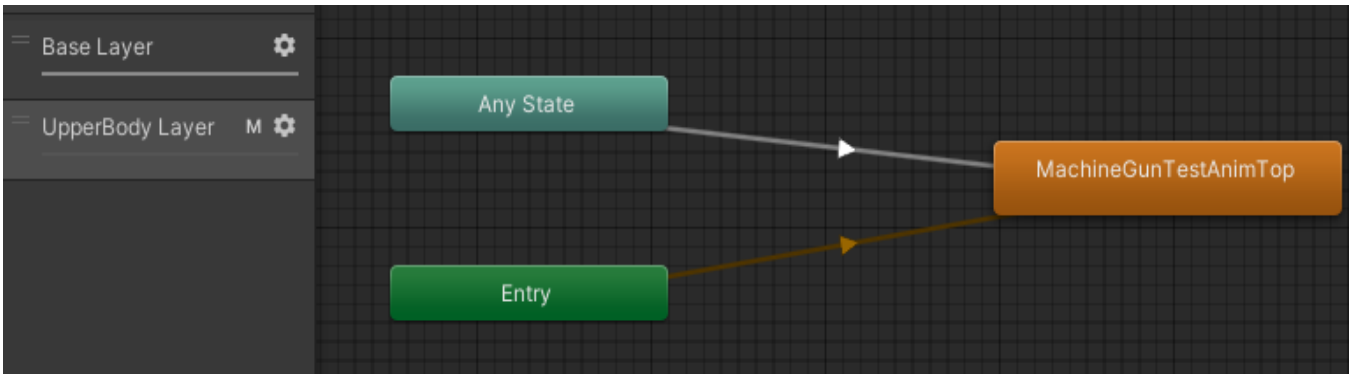


Now click on the gear Icon and in the "Mask" Field add the mask avatar "Low_Poly_Dude_UpperbodyMaskAnim" if you're using the provided character rig or create a new avatar mask for your character and set it in the mask field. You can follow the provided low poly dude's avatar mask as an example to set up yours located in Vauxland > FusionShooterBrawler > GameDataConfigs > CharacterConfigs > Models folder.

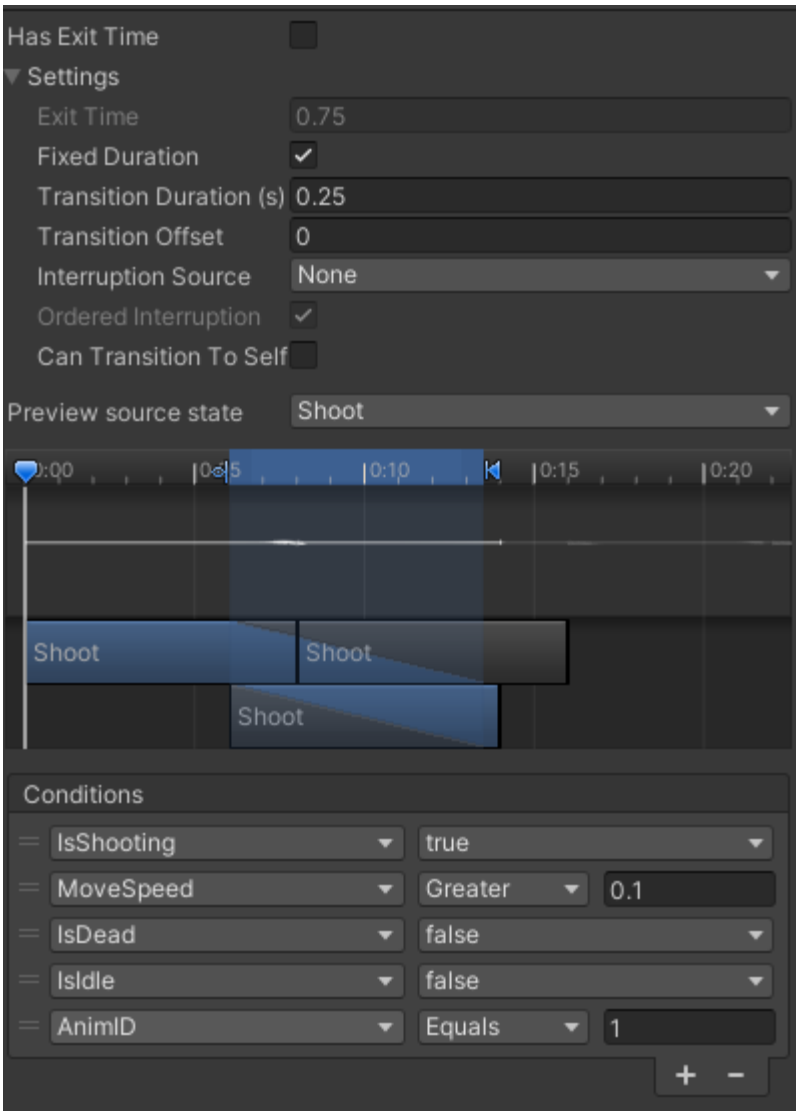
Now right click in the upper body layer and create a new empty state.

Name it the same as your base layer weapon anim if you'd like, add the same animation you added to the base layer weapon anim to this states motion field.

Now create a transition from **Any State** to the new weapon state.



Add these parameters to the conditions of the transition line:

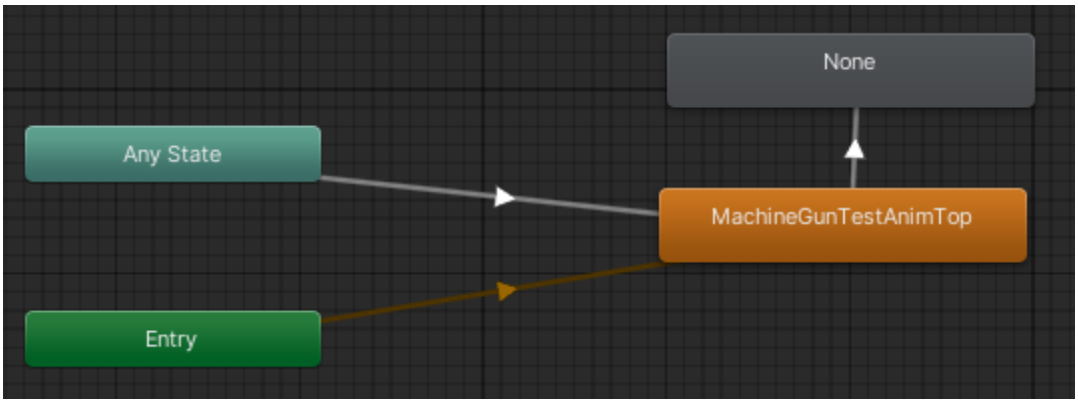


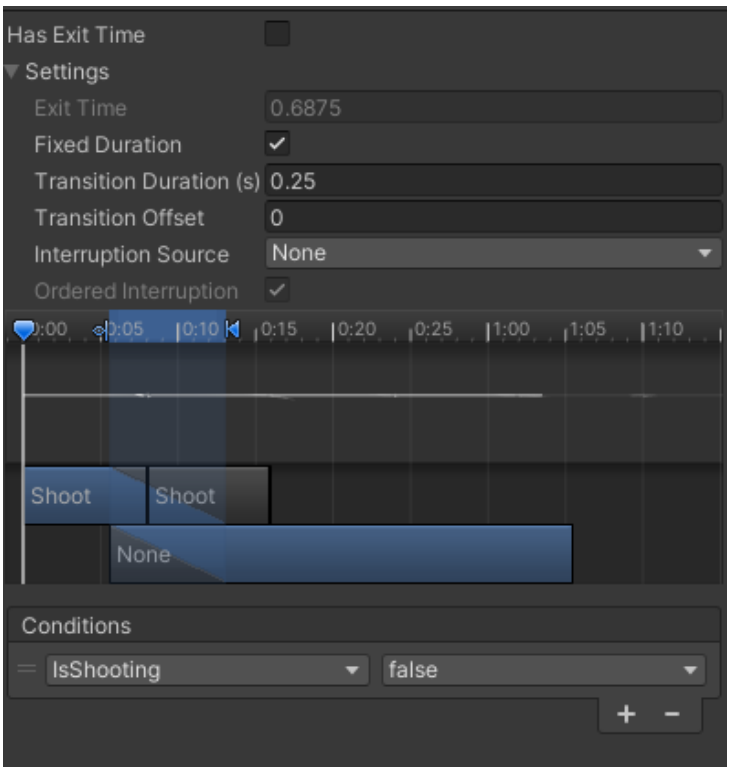
Uncheck Has Exit Time and Can Transition To Self.

IsShooting set to true, **MoveSpeed** set to greater than 0.1, **IsDead** set to false, **IsIdle** set to false because we can't play this shooting animation if we are idle this is only for running and shooting. **AnimID** just like we set in the base layer, needs to be the value you set in the weapon configs data.

Now create another empty state in the upper body layer and name it "none" and keep it empty.

Now create a transition from your weapon anim state to the none state and add these parameters conditions:

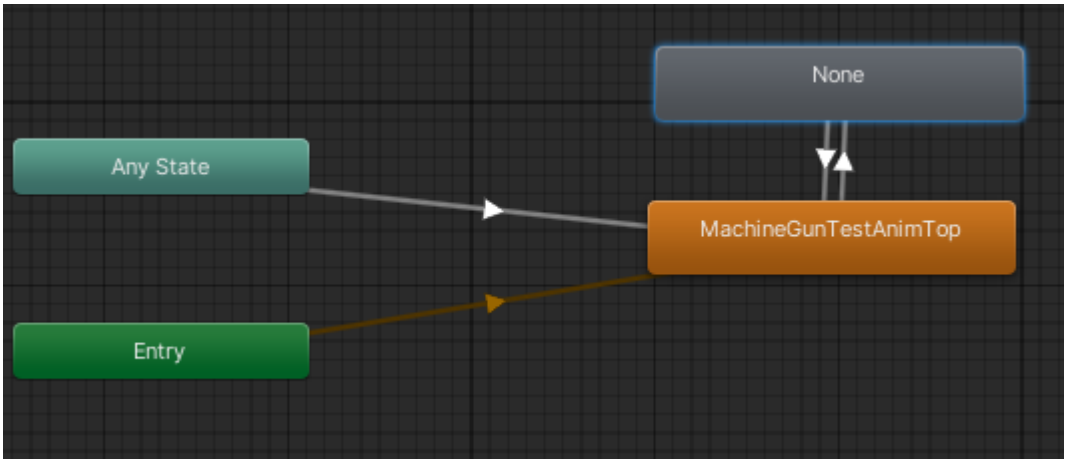


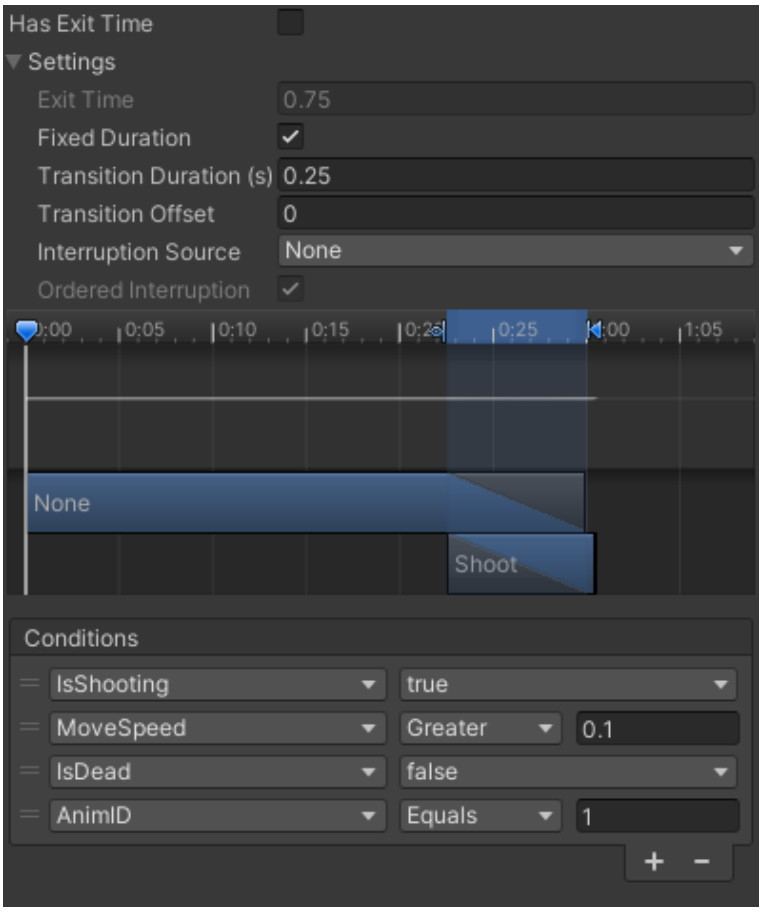


Uncheck Has Exit Time.

IsShooting set to false.

Now make a transition from None state to your weapon anim state and add these parameter conditions:





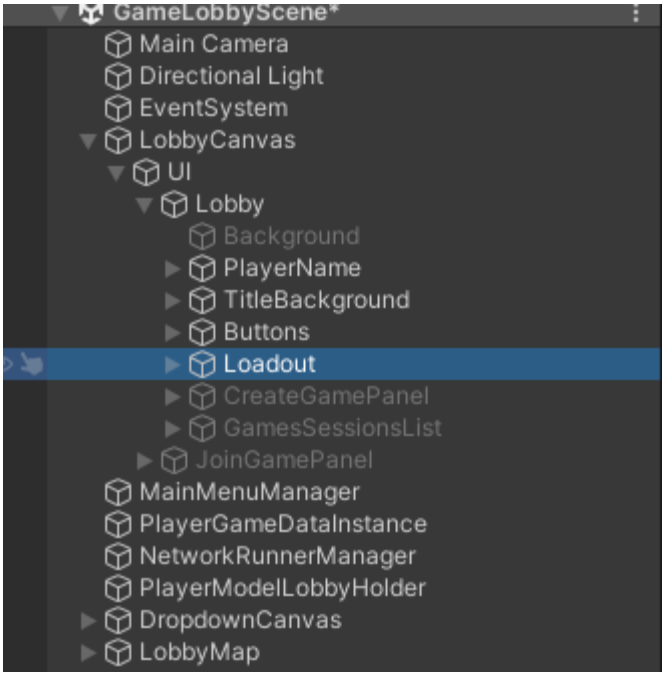
Uncheck Has Exit Time.

IsShooting set to true. **MoveSpeed** set to greater than 0.1, **IsDead** set to false. **AnimID** same as your weapon config datas AnimID value of course.

And that's it now you're done setting up your new weapon's animation. Follow these same steps for all new weapon animations.

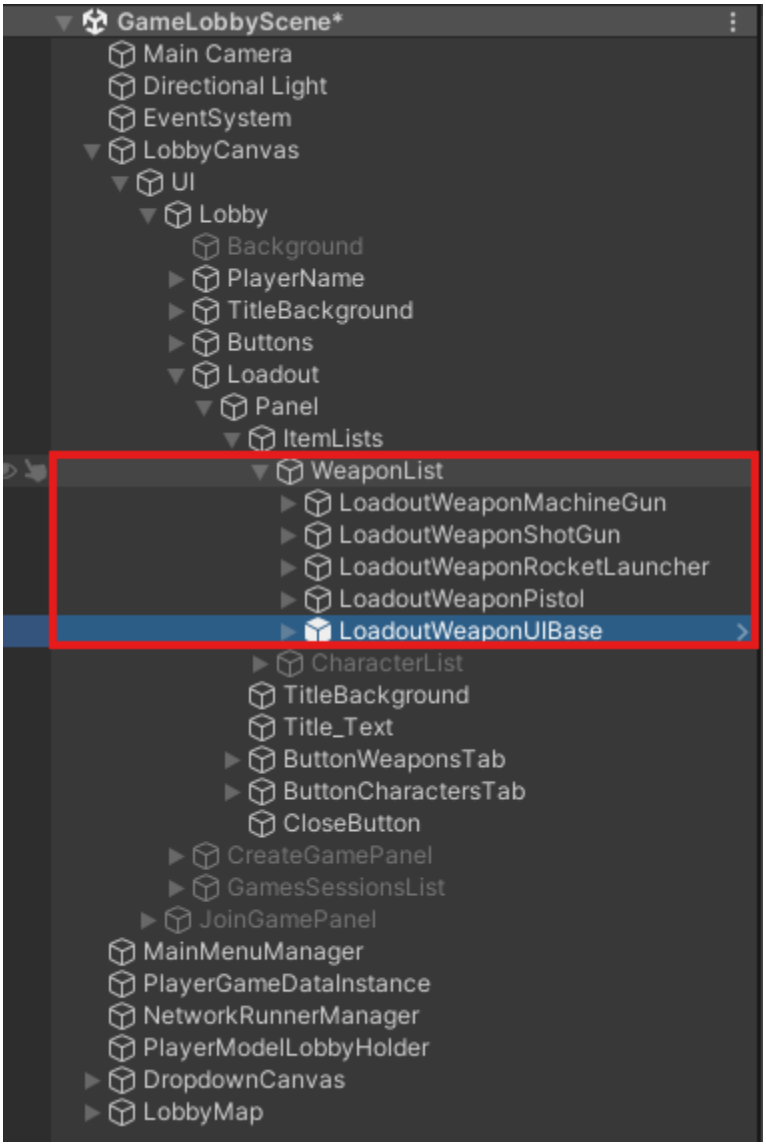
Now lets add the weapons loadout selection to the Loadout UI in the lobby scene.

Go to the lobby scene and set active the **Loadout** panel in the **LobbyCanvas.**



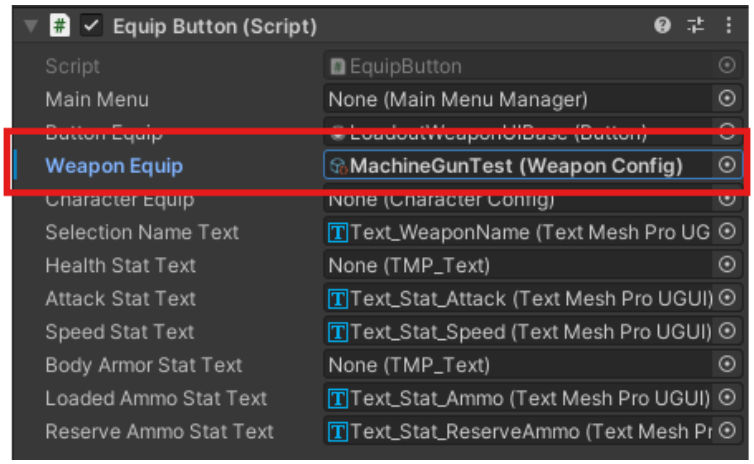
Now go to the weapon loadout UI prefab in **Vauxland > FusionShooterBrawler > Prefabs > UIPrefabs > LobbyUIPrefabs**

And drag and drop the **LoadoutWeaponUIBase** prefab into the **WeaponList** UI:



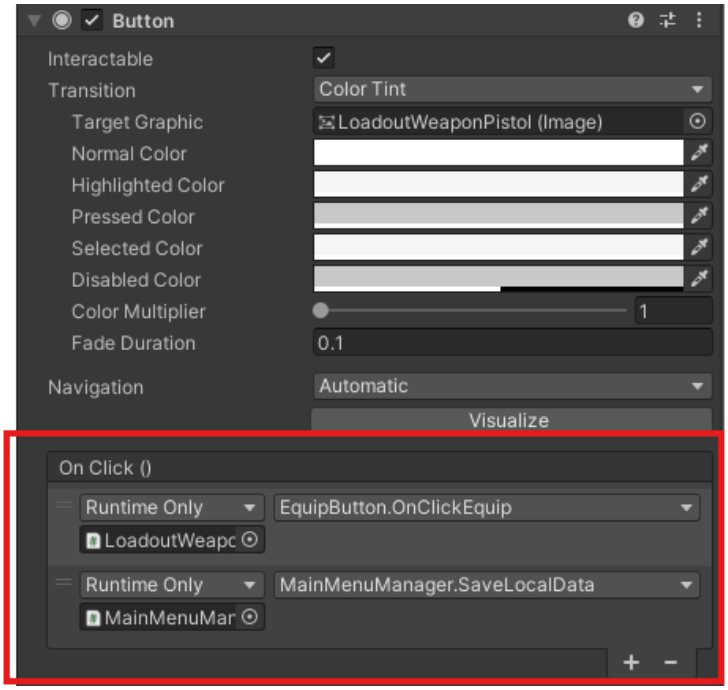
Now set up your loadout UI how you would like add your weapon Icon etc. Change the stat Icons if you want.

Now go to the inspector of this loadout weapon UI base you just set up and scroll down to the **Equip Button** script and in the **Weapon Equip** field add your newly created weapon config we made earlier to this field. This will equip this weapon when you select it.



Also add the Main Menu Manager in the scene to the **Main Menu** field. That's it. You're done completely adding your weapon. Rinse and repeat all steps for any new weapon.

NOTE: You can set up the loadout UI however you like, this is why I didn't add automatically populating list using the loadout weapon UI base prefab for each new weapon config in the player data. This way you can set up your Loadout UI however you'd like as long as you attach the EquipButton script and a button and set the button On click fields to call the **OnClickEquip()** method from the Equip Button script. Also call the **MainMenuManager.SaveLocalData()** method as an on click event as well on the button.



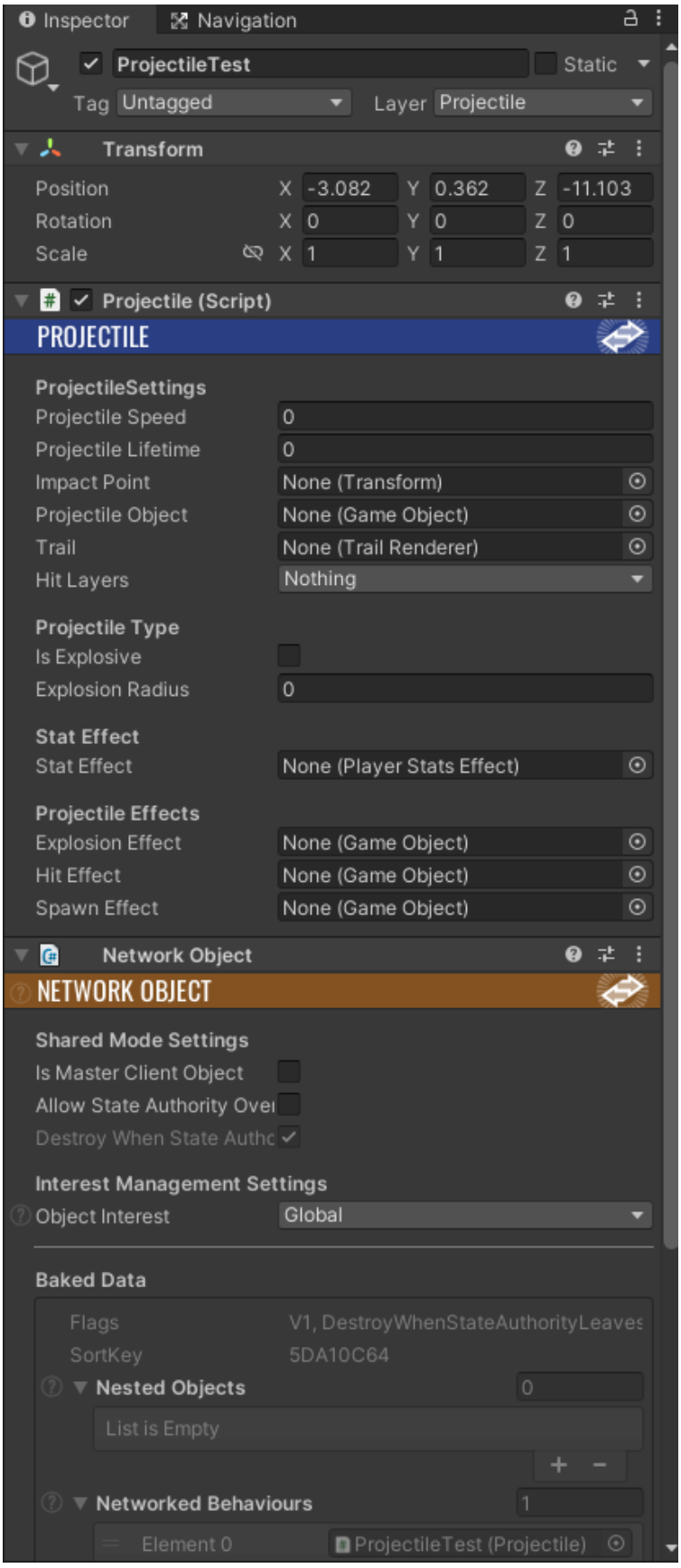
Making a New Projectile

Now lets make a new projectile for your new weapon to shoot.

You can duplicate one of the existing projectiles in **Vauxland > FusionShooterBrawler > Prefabs > Projectiles** and change the visuals of the bullet prefab and effects or make your own from scratch.

To make your own, start by creating an empty object in whatever scene you're in and rename it to whatever you want to call your projectile. Now drag and drop your bullet visual (particle effect or object whatever you want to represent the look of your bullet) as a child of the empty object you just made.

Now go to the inspector of the parent empty projectile object you just made and add a sphere collider and the component **Projectile script** which will then prompt you to add a network object component to your object, click "Add Network Object". Now change the layer to Projectile as well: →→→


Inspector
Navigation
ProjectileTest
Static
Tag Untagged
Layer Projectile
Transform
Position X -3.082 Y 0.362 Z -11.103
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Projectile (Script)
PROJECTILE
ProjectileSettings
Projectile Speed 0
Projectile Lifetime 0
Impact Point None (Transform)
Projectile Object None (Game Object)
Trail None (Trail Renderer)
Hit Layers Nothing
Projectile Type
Is Explosive
Explosion Radius 0
Stat Effect
Stat Effect None (Player Stats Effect)
Projectile Effects
Explosion Effect None (Game Object)
Hit Effect None (Game Object)
Spawn Effect None (Game Object)
Network Object
NETWORK OBJECT
Shared Mode Settings
Is Master Client Object
Allow State Authority Ove
Destroy When State Authc
Interest Management Settings
Object Interest Global
Baked Data
Flags V1, DestroyWhenStateAuthorityLeaves
SortKey 5DA10C64
Nested Objects 0
List is Empty
Networked Behaviours 1
Element 0 ProjectileTest (Projectile)

Let's go over **Projectile Settings**.

Projectile Speed: the speed of the projectile.

Projectile Lifetime: how long you want the projectile to stay active without it hitting anything before it despawns.

ImpactPoint: the part of the projectile that will register the hit first so the tip of it. This is useful for longer projectiles, but the impact point of the projectiles in the demo are just the projectile transform itself.

Projectile Object: the visual of the projectile so the mesh object or effect object that you added a second ago to the empty projectile. This will be set inactive when the projectile hits something.

Trail: the trail renderer attached to the projectile. You do not have to use this.

Hit Layers: all the layers you want to projectile to be affected by. In the demo its Player, Obstacle and Destructable.

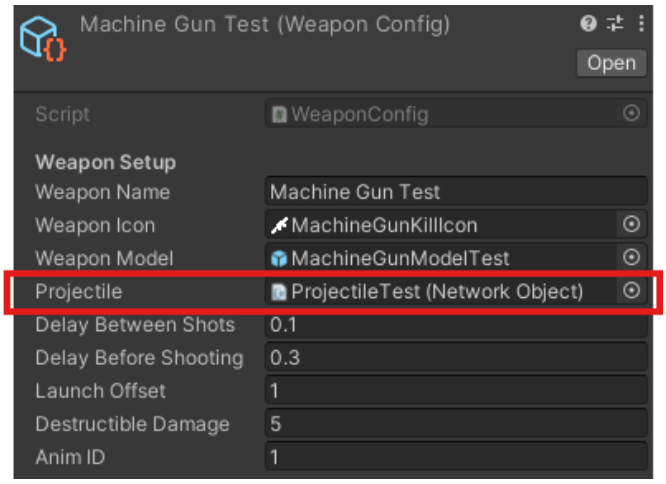
Projectile Type: set if a projectile is an explosive and if so the explosion radius.

Stat Effect: give a projectile a stat effect to apply to other players on hit such as slow down, slowly lose health over time etc.

Projectile Effects: these are the particle effects that will play when a projectile hits something or explodes or if you want a spawn effect to play when the bullet first spawns.

Now back to creating the projectile, set your settings accordingly and add your particle effects for all the effects you want as child objects to the projectile and then assign them to their fields in the projectile inspector and the projectile visual object to the Projectile Object field. Then drag and drop the projectile you just made into the projectiles folder in Vauxland > FusionShooterBrawler > Prefabs > Projectiles.

Now go back to your weapon config data prefab we made earlier and assign this projectile into the projectile field.



Congrats you've completely set up a weapon.

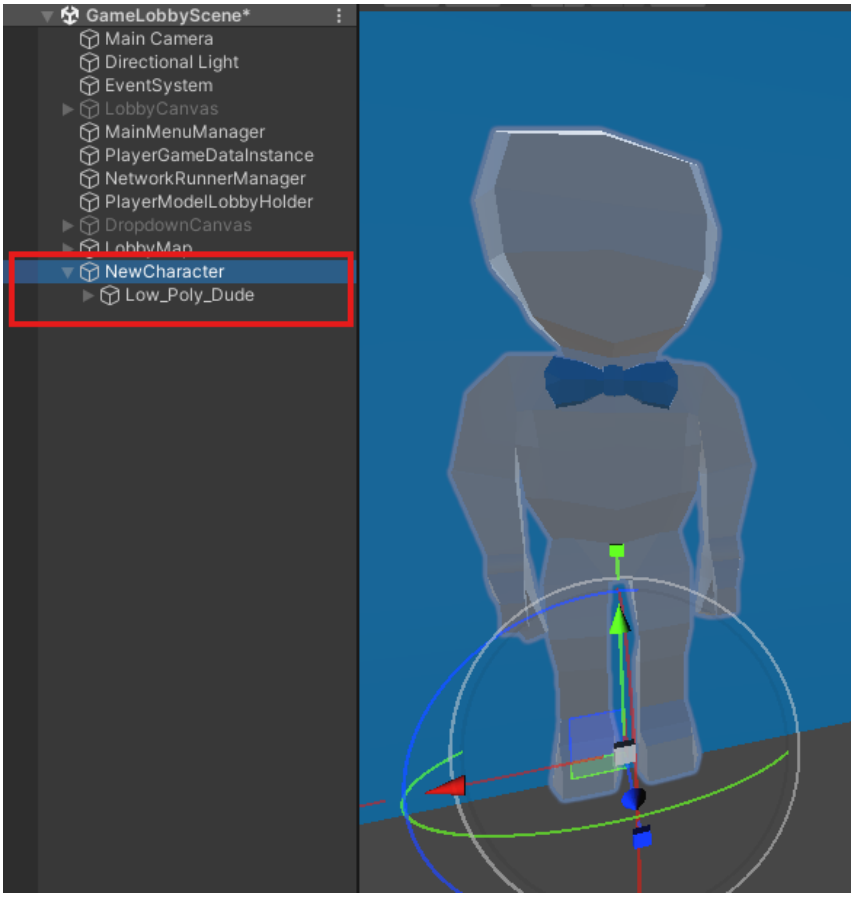
Key Notes: When you set the stats in your weapon config data for your weapon whatever amount you set as the **Reserve Ammo** stat will be your reserve ammo to load from in the match and whatever amount you set for **Loaded Ammo** will be the initial amount of loaded ammo you start with.

Making a New Character

Let’s create a new character.

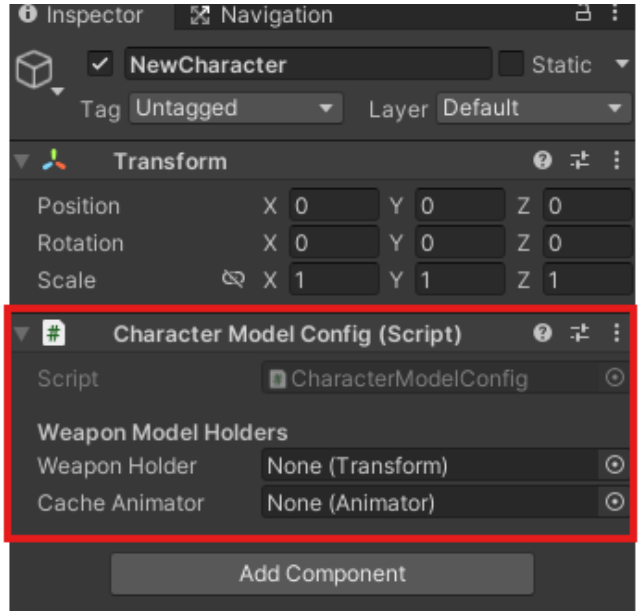
Start by creating an empty object in your Scene and setting the transform position to 0,0,0 name it whatever you want.

Next drag and drop your rigged character you want, to be a child of that empty object you just made.

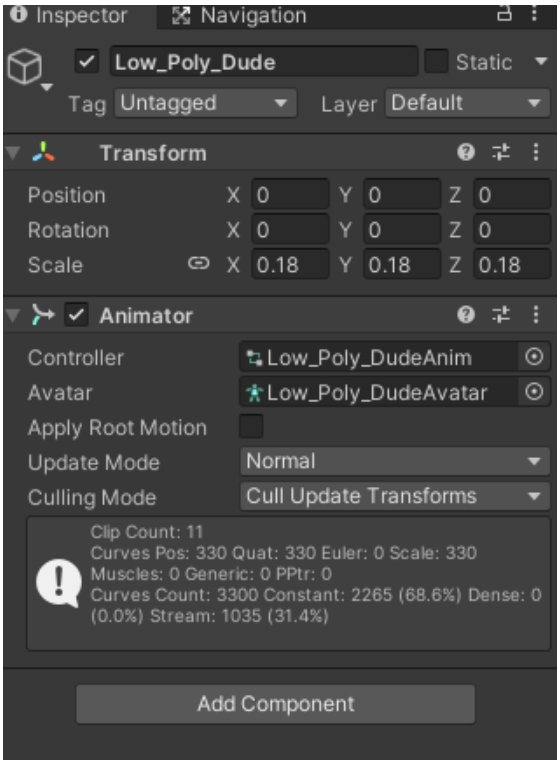


Now in the inspector of your new character add the component

‘**Character Model Config’** script.



Now add the animator we made earlier while making the weapon or if using the character from the demo just add the same animator from the demo to your child rigged character model.



then drag and drop your character model into the **Cache Animator** field.

This will get its animator.

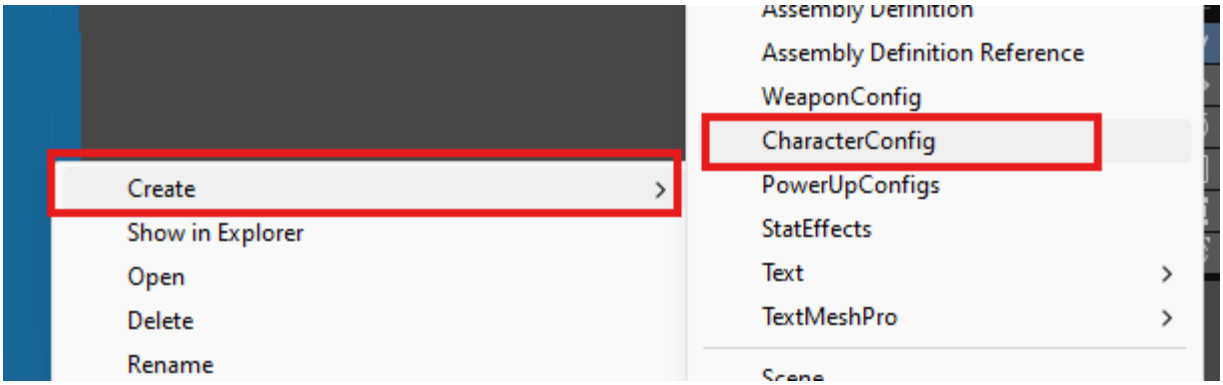
Make the weapon holder just like we did earlier with a character rig when first setting up the weapon model, but on this character model and drag and drop it into the Weapon Holder field.

Now your character model is set up drag and drop it into your assets folder to create a prefab of it and delete this one in the scene.

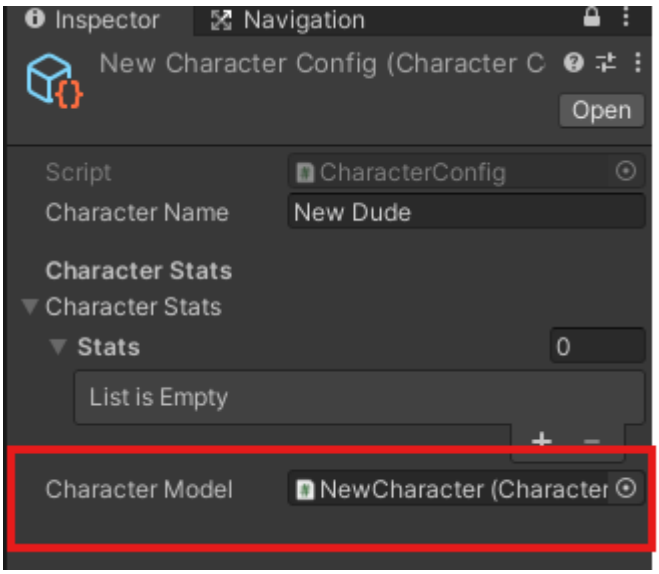
Now lets create the **Character Config** data.

Go to Vauxland > FusionShooterBrawler > GameDataConfigs > CharacterConfigs

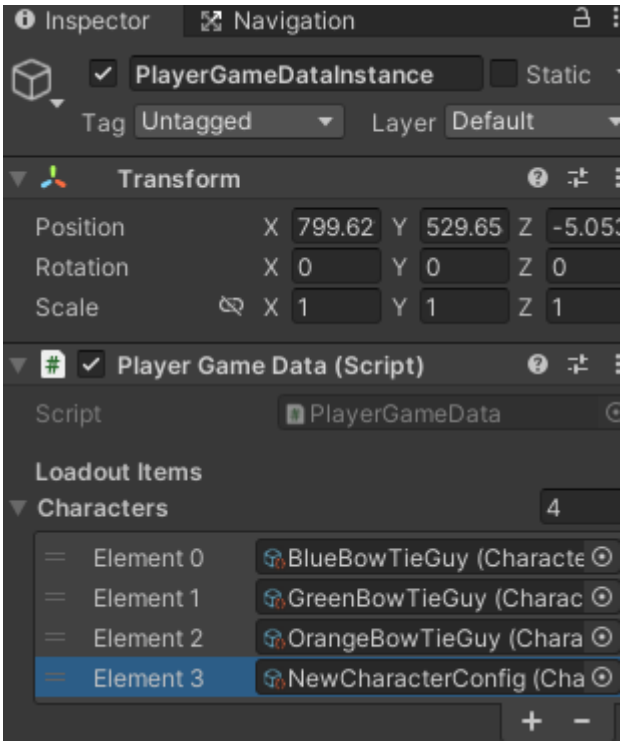
Right click in this folder and Create > FusionBrawler > CharacterConfig.



Now go to the inspector of your newly created character config and drag and drop your new model prefab we just made before this into the **Character Model** field.



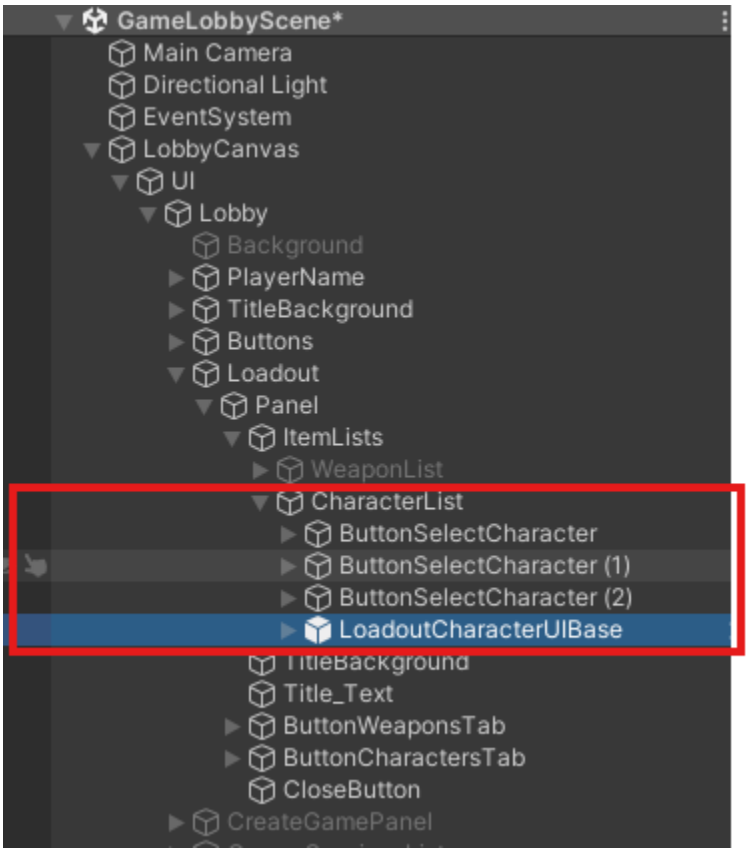
Now set whatever stats you'd like then go to the **PlayerGameDataInstance** object in the lobby scene and drag and drop your new character config data into the **Characters** field of the inspector.



Now lets make the loadout UI for the character.

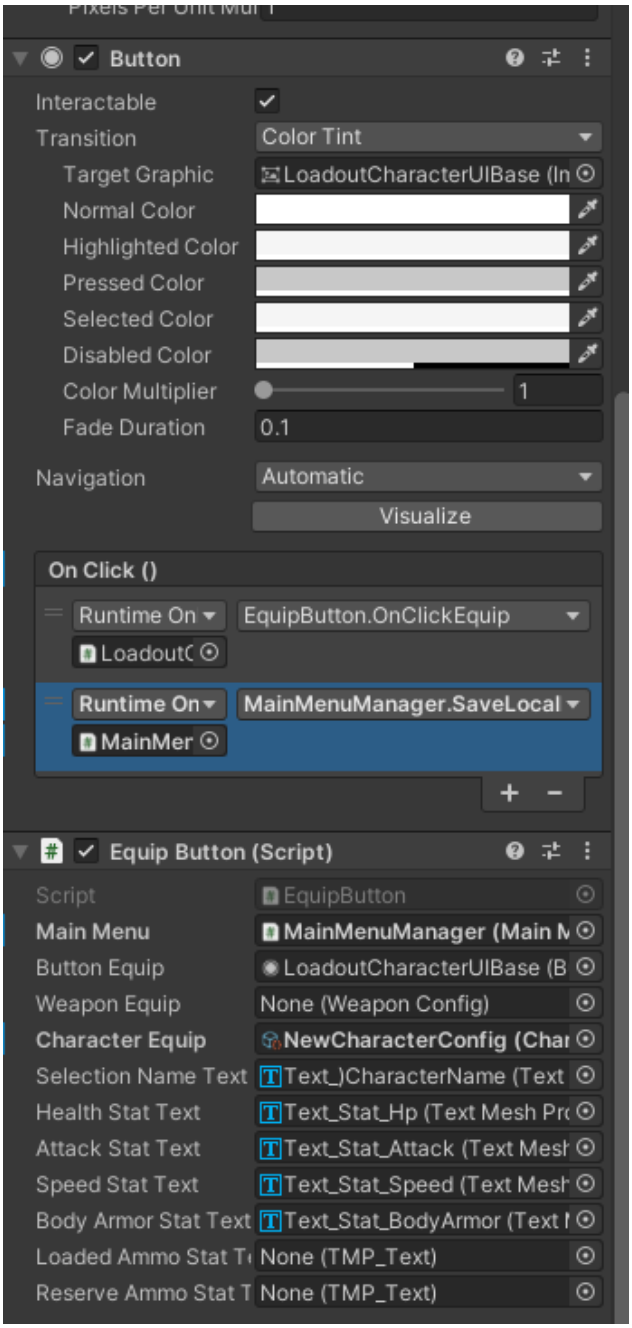
Now same as with making a weapon, go to the lobby scene and set active the loadout UI panel and then go to the character loadout UI prefab in **Vauxland > FusionShooterBrawler > Prefabs > UIPrefabs > LobbyUIPrefabs**

And drag and drop the **LoadoutCharacterUIBase** prefab into the **CharacterList** UI:



Now set up the Ui how you'd like, the character Icon, the stats icons etc.

Next in the inspector of the **LoadoutCharacterUIBase** you just added to your scene scroll down to the **Equip Button** script and in the **Character Equip** field add your newly created character config we made earlier to this field. This will choose this character when you select it. Next add the main menu manager in the scene to the **Main Menu** field. Then add an On click() event to the button and select the main menu manager > SaveLocalData(). Just like we did with the weapon loadout UI.



That’s it for characters, follow same steps for all new characters.

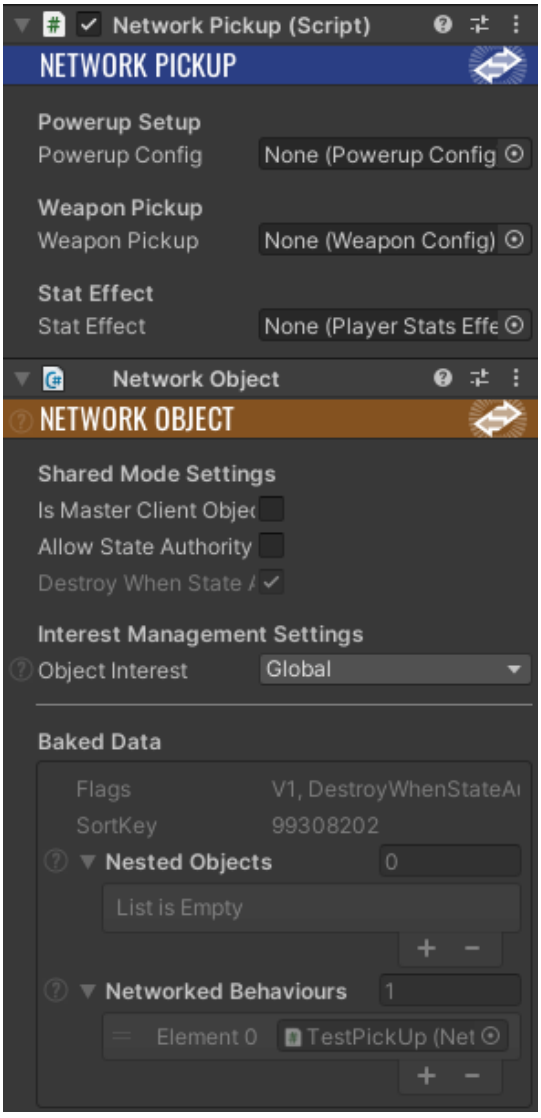
Making a New Pickup

Let’s make a new pickup, start by creating an empty object in your scene again, set transform to 0,0,0.

Drag and drop any model or effect you want to represent the visual of the pickup as a child of the empty object you just made.

Now in the inspector of the empty object you just made add the component script: **Network Pickup**.

It will prompt you to add a network object, click add network object.



To make it a weapon pickup, drag and drop one of your weapon configs you've made into weapon config field.

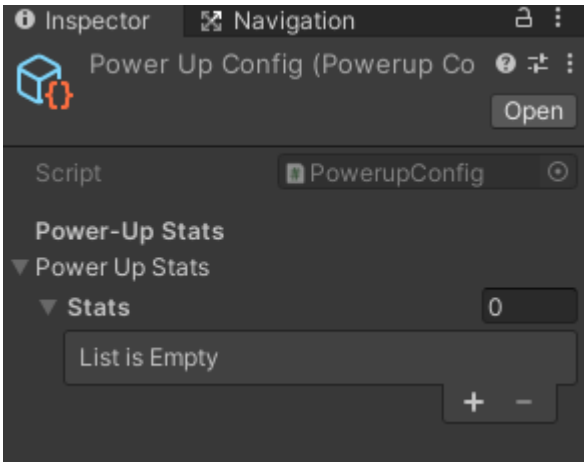
To make a powerup pick up lets create a powerup config.

Go to Vauxland > FusionShooterBrawler > GameDataConfigs > PowerupConfigs

Right click in this folder and create > FusionBrawler > PowerupConfig.

Now Add whatever stats you want this powerup config to give a player on pick up. These are permanent until the player dies and then their stats are reset.

Powerup configs are useful for replenishing stats like ammo, health and body armor pickups.

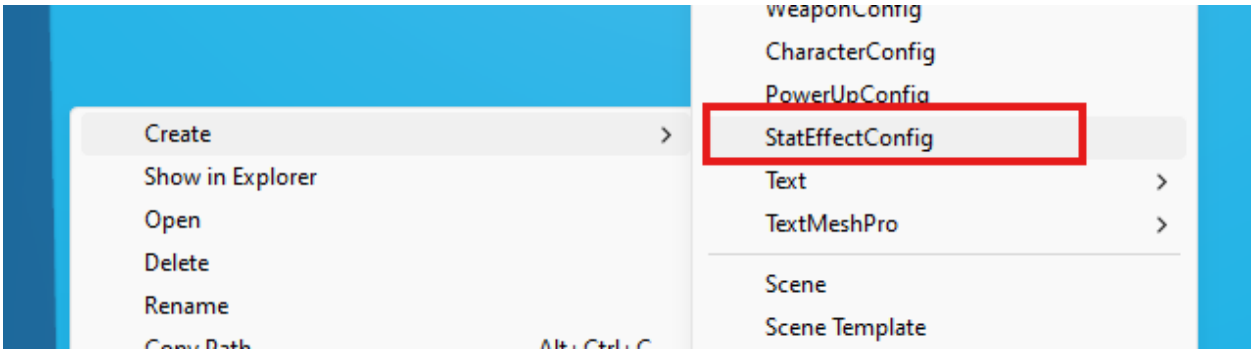


Drag and drop your powerup into the **Powerup Config** field for power up pick up.

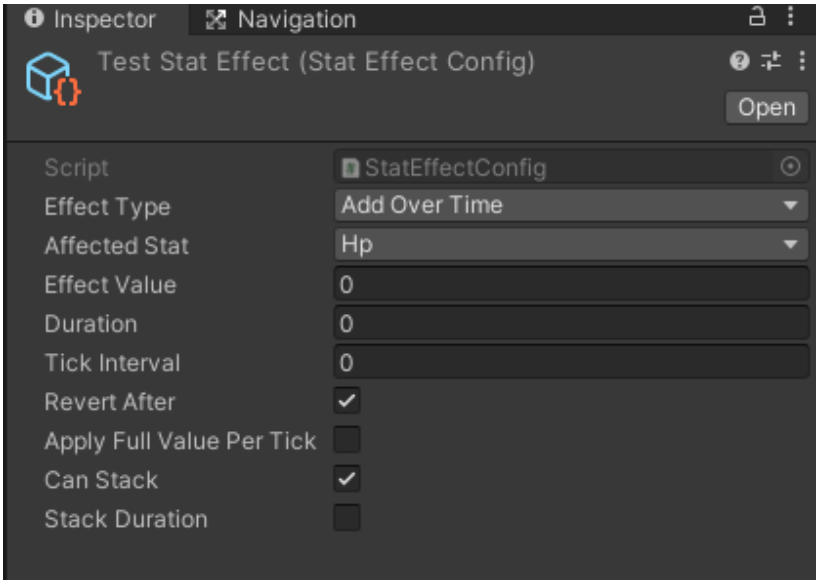
For a **Stat Effect** pickup we'll need to make a **Stat Effect Config**

Go to Vauxland > FusionShooterBrawler > GameDataConfigs > StatEffectConfigs > PickupStatEffects.

Right click in this folder and create > StatEffectConfig.



Now lets look at the inspector of the Stat Effect Config:



The **Effect Type** consist of 3 types: **Add Over Time** which will increase a stat by the effect value amount over the duration set. The **Tick Interval** is how often the value you is applied, so for example if you have an add over time of Hp effect value of 60, for a duration of 6 seconds and a tick interval of 2, it would apply 20 hp to the player every 2 seconds. If you checked the **Apply Full Value Per Tick** box it would apply the full 60 every 2 seconds. The **Can Stack** check is for if the stat effect can stack, so lets say a player picks up 2 Hp pickups, if this box is checked then the player will receive 40 Hp every 2 seconds or if the apply full value is also ticked then 120 Hp every 2 seconds. If you have **Stack Duration** checked instead then it will just stack the duration of the effect. You can't check both, its either can stack the full effect or just the duration of the effect.

The same applies for the second **Effect Type: Reduce Over Time** except it reduces the stat by the effect value now.

The third **Effect Type** is **Temporary Boost** which is like a power up but will only last for the duration unless you uncheck revert after then its permanent until death.

Stat effect configs can be applied to pickups and projectiles.

Now once you've set up your stat effect config drag and drop it into the **Stat Effect** field of the network pickup if you're making a stat effect pickup.

Once you're done setting up your network pick you will add it to the **Spawn Manager** in the **Match Scene** which is covered in the next page.

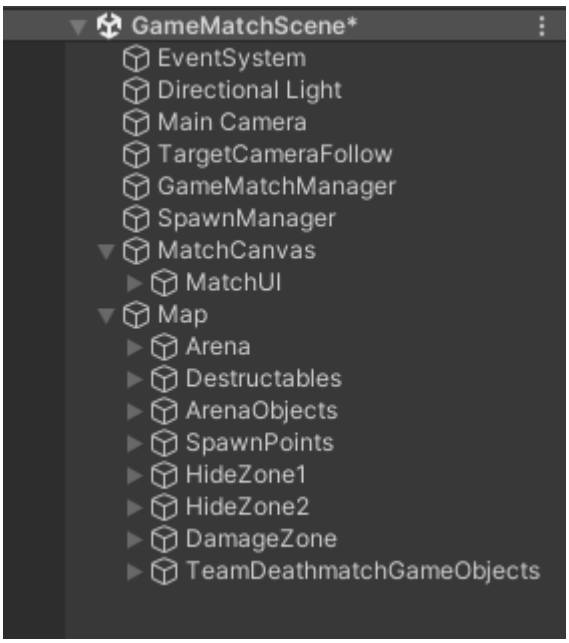
That's it for pickups, follow same steps for all network pickups.

Match Scene

Let’s take a look at the match scene.

Go to Vauxland > FusionShooterBrawler > Scenes > BrawlScene

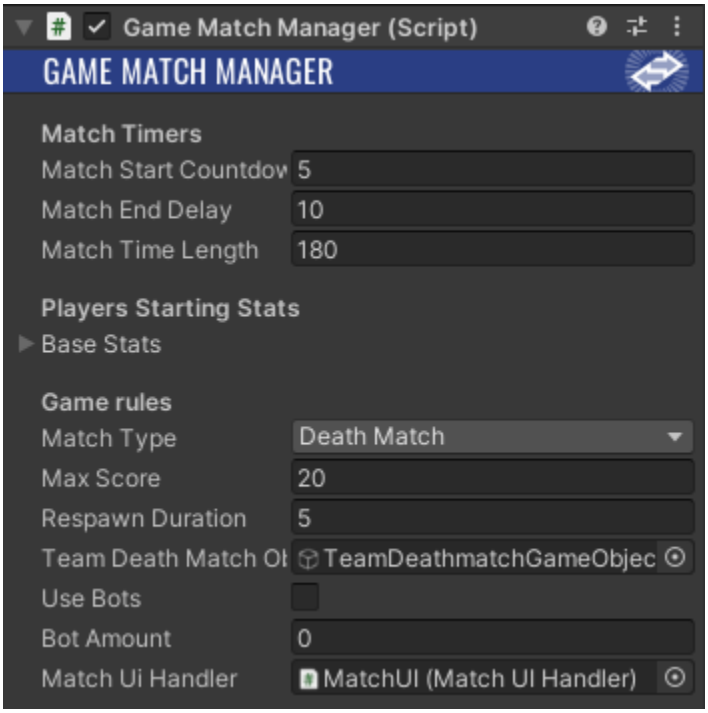
to open the scene.



A match scene will require 3 main components: a **Game Match Manager**, a **Spawn Manager** and a **Match UI Handler.**

The camera will need a **CameraFollow** component attached to it.

Lets start by looking at the **GameMatchManager** in the hierarchy.



The Game Match Manager controls the flow of the match.

Match Start Countdown is countdown time that shows before players spawn an the match starts.

Match End Delay is the amount of time before the player returns to the lobby scene after the match has ended.

Match Time Length is the match time limit in seconds. However long you want your matches to be.

Base Stats the starting stats for all players, everything else they bring to the match from the selected character or weapon will be added on top of this.

Match Type choose between deathmatch or team deathmatch for this scene. You can set the match type from the main menu lobby as well when creating a game.

Max Score the max score allowed in the match. Once reached the match is over.

Respawn Duration is the amount of time it takes for a player to respawn after they die.

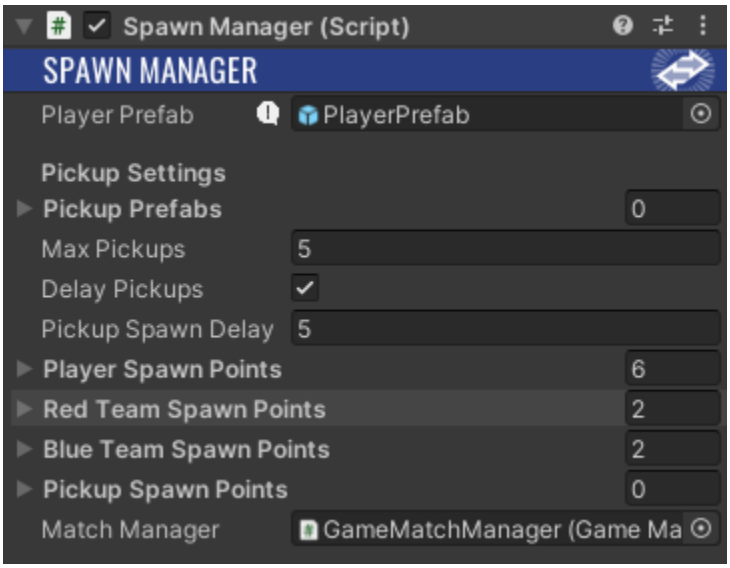
Team Death Match Objects are the map objects specifically for a team deathmatch match. These will be set active for team deathmatch and turned off for deathmatch.

Use Bots if you want to allow bots for this match. Can also be set in create game panel from lobby scene.

Bot Amount the amount of bots you want if you use bots.

Match UI Handler the match UI handler component in the scene that handles our match UI.

Now lets look at the **Spawn Manager** inspector:



Player Prefab this is the players prefab that contains all of the players scripts. Its located in FusionShooterBrawler > Prefabs > PlayerPrefabs.

Pickup Prefabs here you would set all of the network pickup prefabs that we went over how to make earlier.

Max Pickups the max amount you want spawned at any given time.

Delay Pickups do you want pick ups to respawn immediately when picked up or delay their respawn with the **Pickup Spawn Delay** amount.

Player Spawn Points here you set all of your spawn points you have in the match for players. It will choose randomly from them.

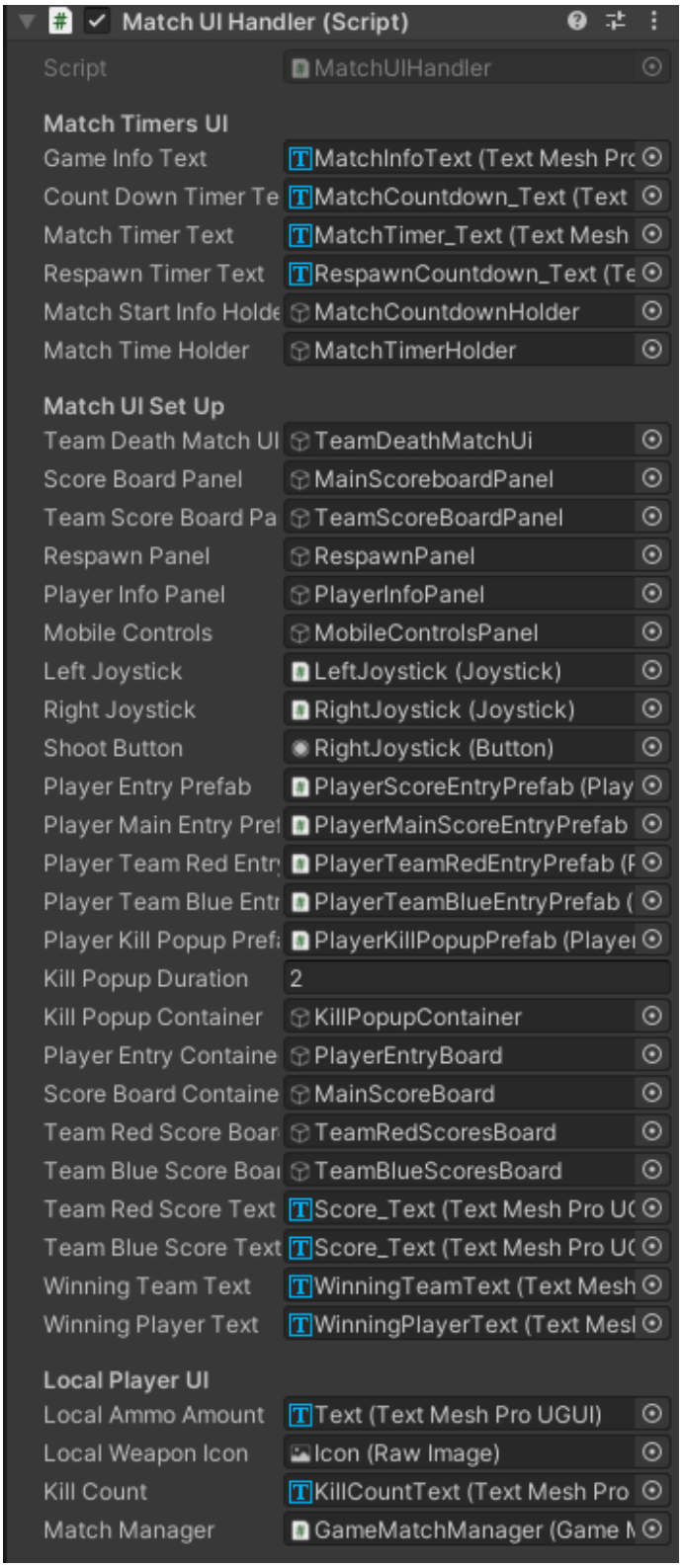
Red Team Spawn Points set all of the red team specific spawn points here.

Blue Team Spawn Points set all of the blue team specific spawn points here.

Pickup Spawn Points all of the specific pickup spawn points go here.

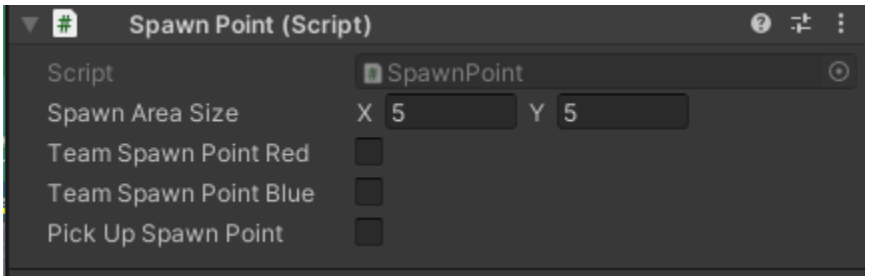
Match Manager our game match manager in the scene.

Now let's look at the inspector of the **Match UI Handler**:



This controls all of the matches UI. Click on each one and see where it goes in the **Match Canvas** UI object.

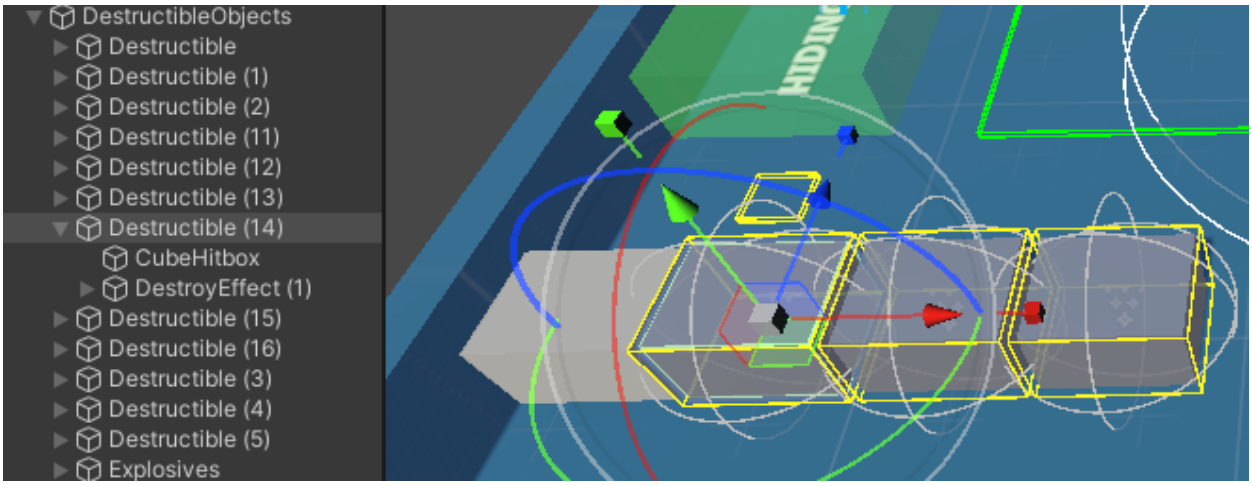
To make a new **Spawn Point** create an empty object in the scene an add the component **Spawn Point** script to it in the inspector.



Adjust the area as you like and check one of the boxes for which spawn point it is if you want, you do not have to do this, this just changes the color of its gizmo to keep track of them easier.

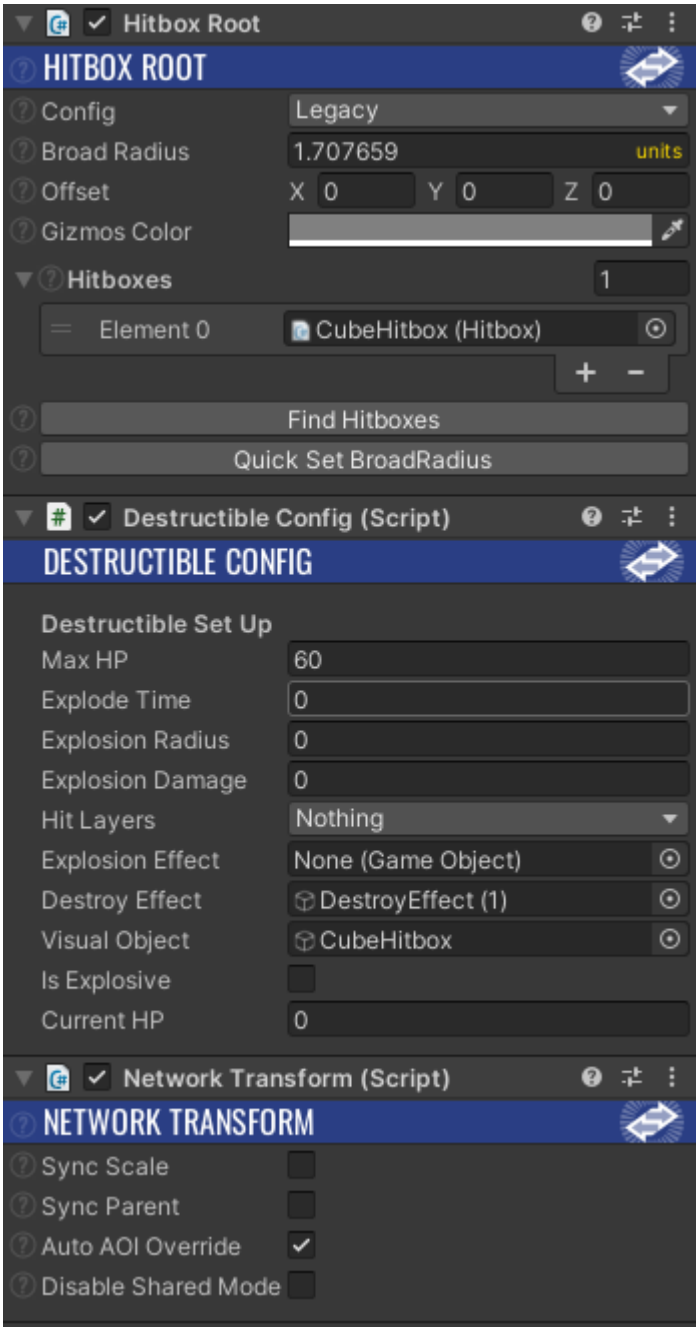
Now lets go over **Destructible** objects.

This is a destructible in the scene:



A destructible consist of a child object representing the visual of the destructible and containing the **Hitbox** component and a particle effect to play when the destructible is destroyed.

The parent object of the destructible contains the **Destructible Config** component and the **Hitbox Root** component and a **Network Transform** component.



The Hitbox Root tracks the child Hitbox component and registers the hits it receives over the network. This must be on the object to work. If you want multiple hitboxes for the destructible add them to child objects and then click **Find Hitboxes**.

In the Destructible Config component here we set up the destructible.

Max Hp the max hp of the destructible.

Explode Time if you make this destructible an explosive you would set the time it takes to explode if its hit with a projectile initially. If the hp is not reduced to 0 after the initial hit before the explode timer runs out then it will automatically explode. Reducing the hp of the destructible causes it to explode faster.

Explosion Radius the range the explosion has to damage things

Explosion Damage how much damage the explosion does to players and other destructible objects.

Hit Layers the layers you want the explosive destructible to affect.

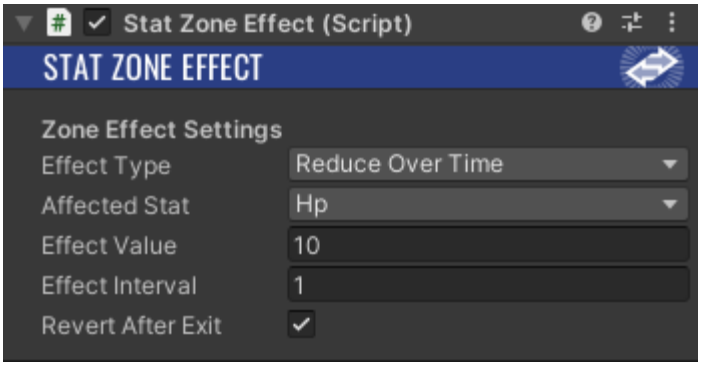
Explosion Effect if an explosive the effect to play

Destroy Effect if just a destructible the effect to play.

Visual Object the destructible objects model that will be set inactive when destroyed.

Is Explosive set whether you want it to be an explosive.

To create a new **Stat Effect** zone create an empty game object in the scene and attach the component **Stat Zone Effect** to it.



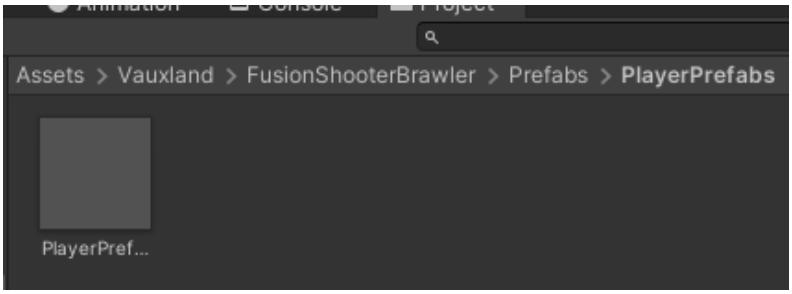
Like stat effect config, choose the effect type to either add the stat or reduce the stat or boost the stat while the player is in the zone.

Set the bounds of the zone by setting the size of the collider on the object you're making the stat effect zone. Set the collider to trigger.

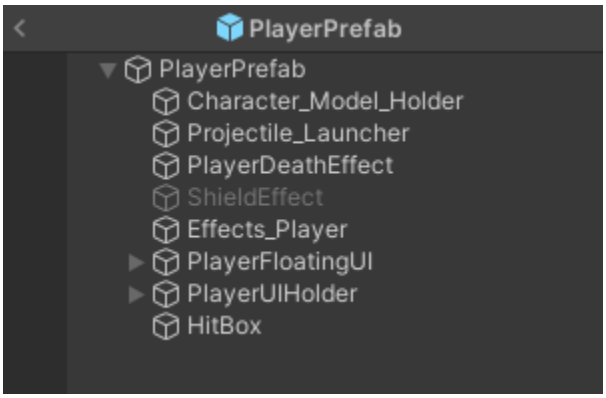
PLAYER PREFAB

The player prefab is the core player object that is spawned when the match scene loads and contains all of the player scripts and UI components to control the player.

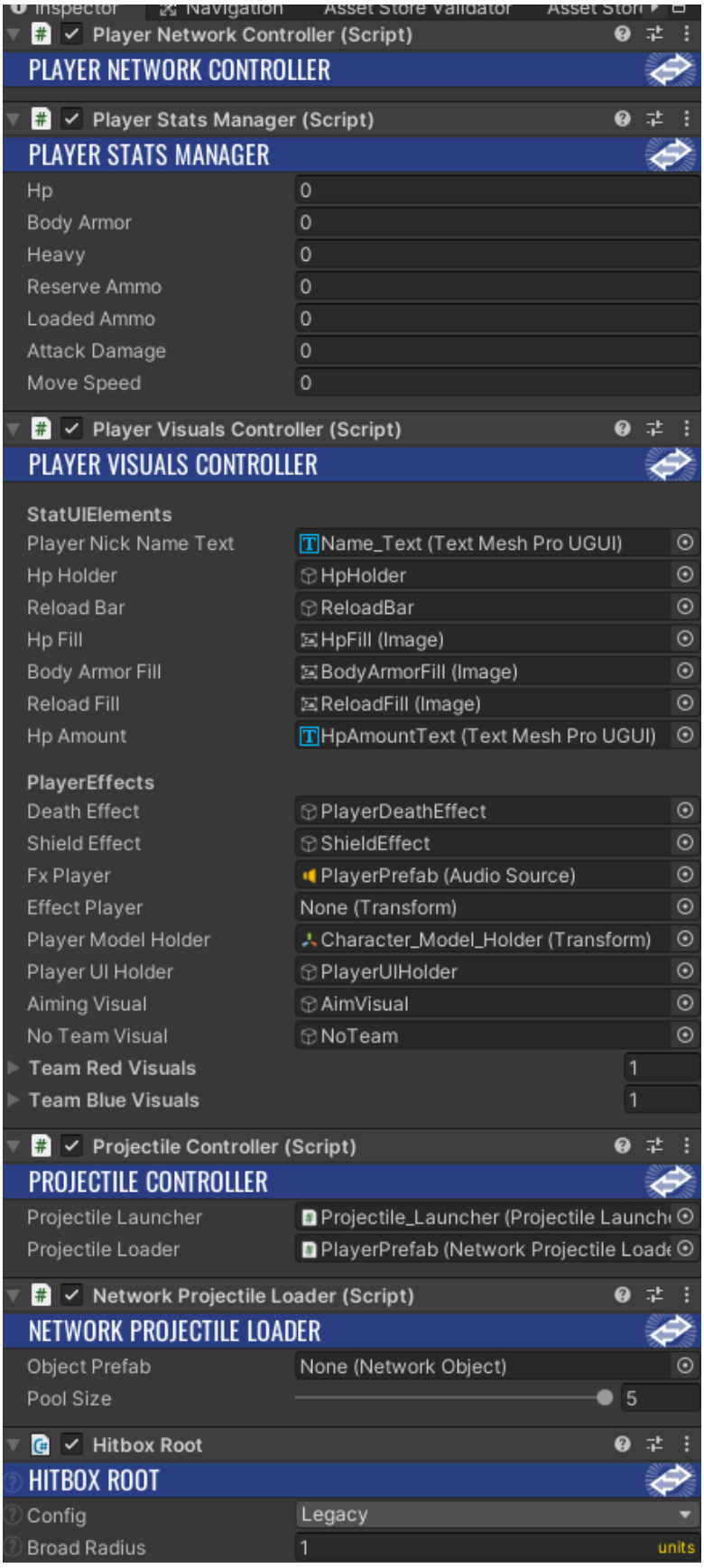
You can find the player prefab in Vauxland > FusionShooterBrawler > Prefabs > Player prefab.

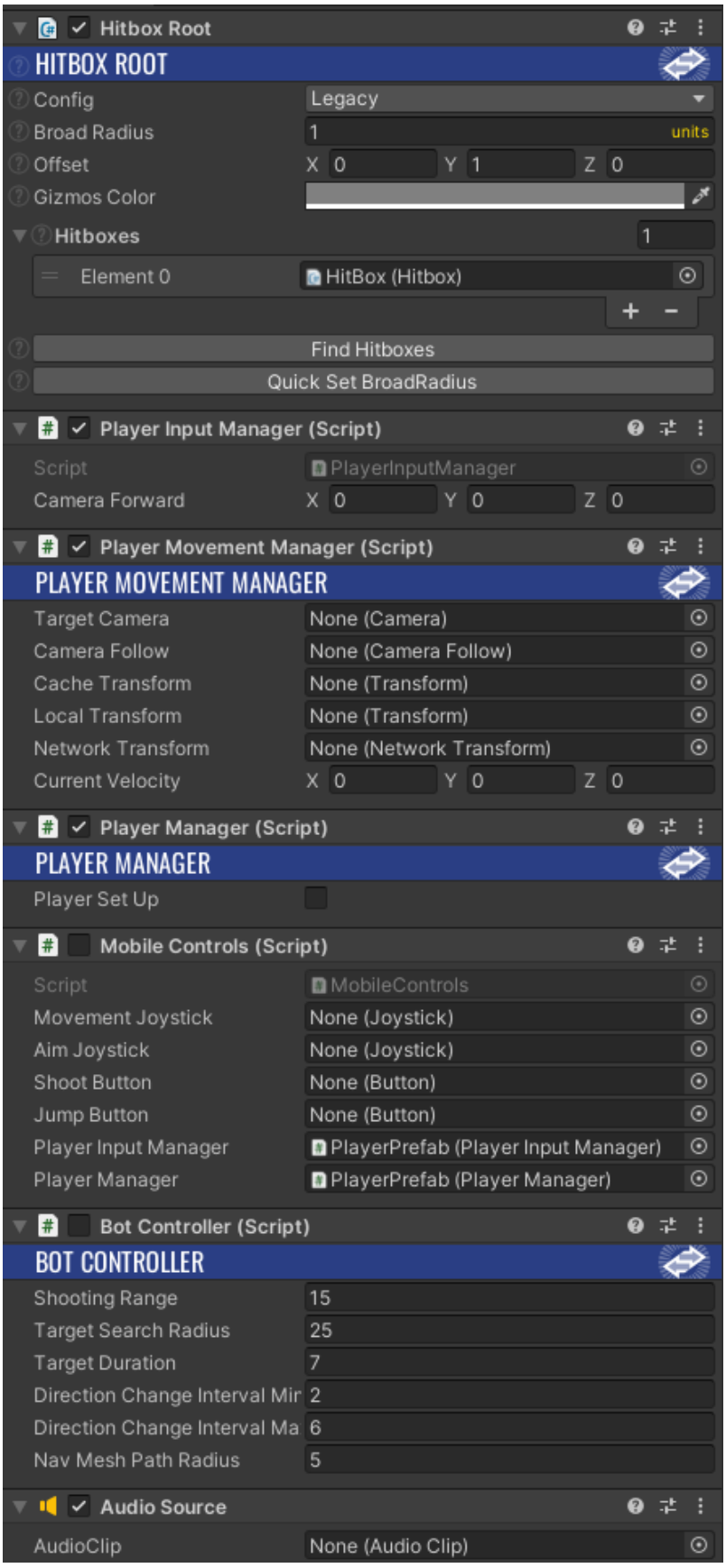


Double click this and open it up if you want to edit the UI components of the player object, change things out, add scripts etc.



Lets take a look at the scripts on the player object:


Player Network Controller (Script)
PLAYER NETWORK CONTROLLER
Player Stats Manager (Script)
PLAYER STATS MANAGER
Hp 0
Body Armor 0
Heavy 0
Reserve Ammo 0
Loaded Ammo 0
Attack Damage 0
Move Speed 0
Player Visuals Controller (Script)
PLAYER VISUALS CONTROLLER
StatUIElements
Player Nick Name Text Name_Text (Text Mesh Pro UGUI)
Hp Holder HpHolder
Reload Bar ReloadBar
Hp Fill HpFill (Image)
Body Armor Fill BodyArmorFill (Image)
Reload Fill ReloadFill (Image)
Hp Amount HpAmountText (Text Mesh Pro UGUI)
PlayerEffects
Death Effect PlayerDeathEffect
Shield Effect ShieldEffect
Fx Player PlayerPrefab (Audio Source)
Effect Player None (Transform)
Player Model Holder Character_Model_Holder (Transform)
Player UI Holder PlayerUIHolder
Aiming Visual AimVisual
No Team Visual NoTeam
Team Red Visuals 1
Team Blue Visuals 1
Projectile Controller (Script)
PROJECTILE CONTROLLER
Projectile Launcher Projectile_Launcher (Projectile Launch
Projectile Loader PlayerPrefab (Network Projectile Loade
Network Projectile Loader (Script)
NETWORK PROJECTILE LOADER
Object Prefab None (Network Object)
Pool Size 5
Hitbox Root
HITBOX ROOT
Config Legacy
Broad Radius 1 units


Hitbox Root
HITBOX ROOT
Config
Legacy
Broad Radius
1
units
Offset
X 0
Y 1
Z 0
Gizmos Color
Hitboxes
1
Element 0
HitBox (Hitbox)
Find Hitboxes
Quick Set BroadRadius
Player Input Manager (Script)
Script
PlayerInputManager
Camera Forward
X 0
Y 0
Z 0
Player Movement Manager (Script)
PLAYER MOVEMENT MANAGER
Target Camera
None (Camera)
Camera Follow
None (Camera Follow)
Cache Transform
None (Transform)
Local Transform
None (Transform)
Network Transform
None (Network Transform)
Current Velocity
X 0
Y 0
Z 0
Player Manager (Script)
PLAYER MANAGER
Player Set Up
Mobile Controls (Script)
Script
MobileControls
Movement Joystick
None (Joystick)
Aim Joystick
None (Joystick)
Shoot Button
None (Button)
Jump Button
None (Button)
Player Input Manager
PlayerPrefab (Player Input Manager)
Player Manager
PlayerPrefab (Player Manager)
Bot Controller (Script)
BOT CONTROLLER
Shooting Range
15
Target Search Radius
25
Target Duration
7
Direction Change Interval Mir
2
Direction Change Interval Ma
6
Nav Mesh Path Radius
5
Audio Source
AudioClip
None (Audio Clip)

The **Player Network Controller** script handles setting the players nickname in the match, players team, players kills and deaths, if the player is hiding in a hiding zone or not and if the player is a bot or not.

The **Player Stats Manager** script handles all of the players stats and everything that alters them. It also controls setting all of the players loadout, characters, cosmetics, reloading, and respawning. Any new stat you would like to add you would follow how the other stats are added in here and then also add the name of it to the **PlayerStats** script.

The **Player Visuals Controller** script handles all of the player objects UI changes and effects.

The **Projectile Controller** script handles shooting the projectiles from the players weapon

The **Network Projectile Loader** script handles pre spawning projectiles for clients to compensate lag on clients for smooth network shooting

The **Hitbox Root** is a fusion script that handles registering hits received on the player

The **Player Input Manager** script handles setting our input received from the controls and sends them to the fusion simulation and tell the other players this is where our player is moving and is doing

The **Player Movement Manager** script handles moving and rotating the player transform locally and the bots.

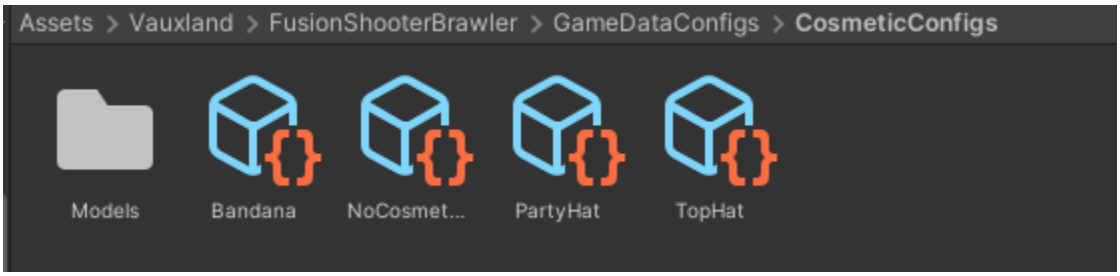
The **Player Manager** script sets all of our other players scripts on the player object in the OnSpawn method which is a fusion method called when a networked object is spawned in the simulation. This script sets up our player and is used by the other scripts to call from or to other scripts on our player.

The **Mobile Controls** script handles using and setting the mobile controls and is set active if the application is on mobile.

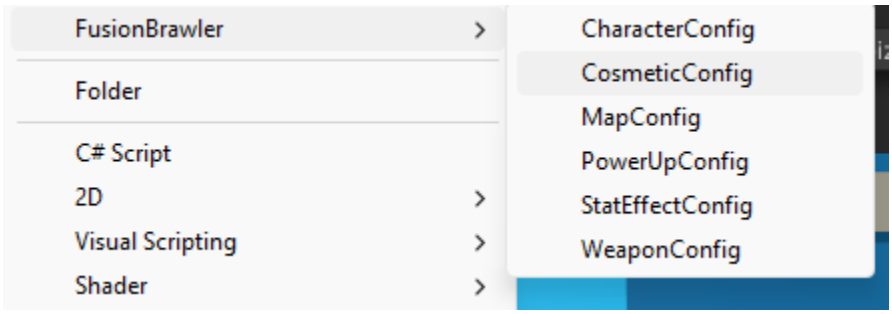
The **Bot Controller** scripts handles controlling our bots and is set active if the player object is spawned in as a bot. You would edit this script to edit the bot's behavior.

MAKING A NEW COSMETIC

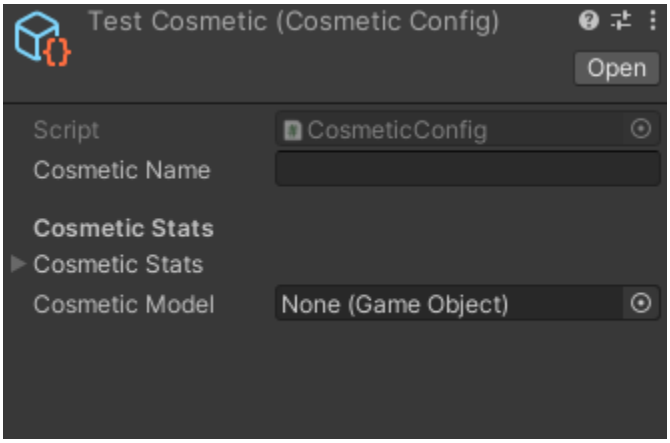
To make a new cosmetic go to Vauxland > FusionShooterBrawler > GameDataConfigs > CosmeticConfigs.



Right click in the folder and go to Create > FusionBrawler > CosmeticConfig.



After you've created a new cosmetic config click on it to look at its inspector:

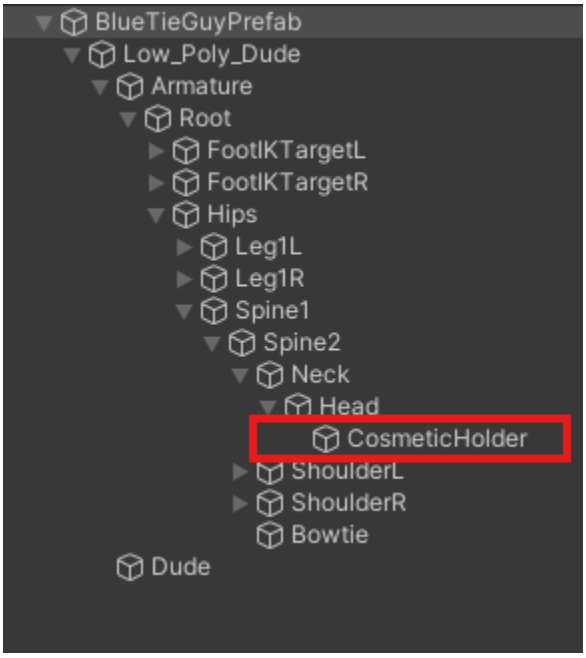


Whatever you put as the cosmetic name will be the name displayed in the loadout UI.

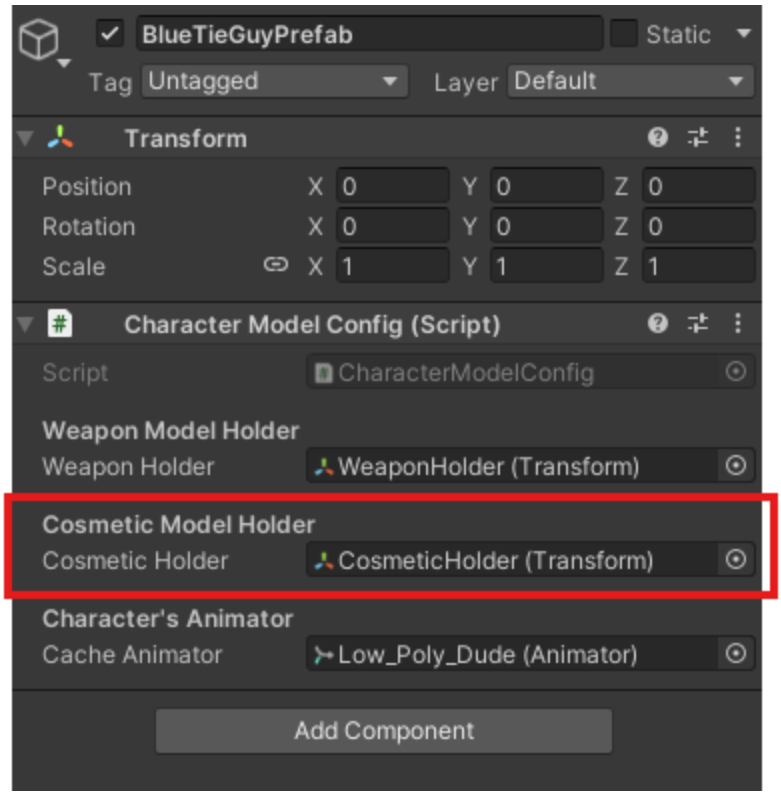
You can add stats to cosmetics but the demo doesn't use stats on the cosmetics.

Now lets make the **Cosmetic Model** prefab to assign to the cosmetic model field.

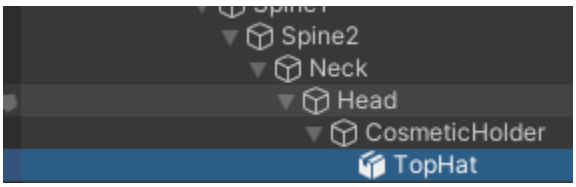
Double click on one of your character model prefabs of your current **Character Configs** to open up its prefab scene. Drop down its armature like we did when creating the weapon model prefabs. Now add a empty game object onto the head root or anywhere you would like and call it cosmetic holder and leave its position at 0,0,0. (you can expand the cosmetic system to have multiple cosmetics just copy the current code set up to make a new cosmetic slot).



Now drag and drop this object into the **Cosmetic Model Holder** field in the character model config script that's on this character model.



Now to create a Cosmetic Model take whatever model take whatever model you want to use and drag and drop it on the cosmetic model holder object as a child:



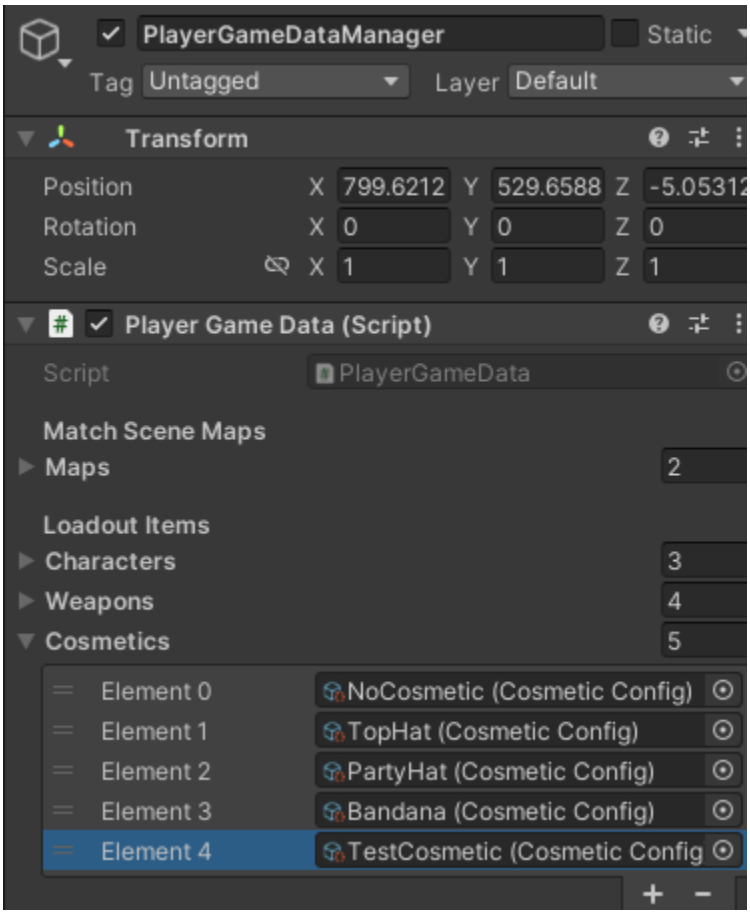
Now click on the Cosmetic Holder object and drag and drop it into your assets folder to make it a prefab. Rename it to whatever you want to call your cosmetic model.

Now right click on the cosmetic model holder object on your character and click ‘unpack prefab’ because we don’t want this as a prefab. Also delete the cosmetic model that’s a child of the cometic holder. You want your cosmetic holder empty on your character object. The cosmetic model prefab we just created will be instantiated on it in the game.

Repeat these steps for all cosmetics models.

Now drag and drop your cosmetic model prefab you just made in to the cosmetic model field of the cosmetic config we made earlier.

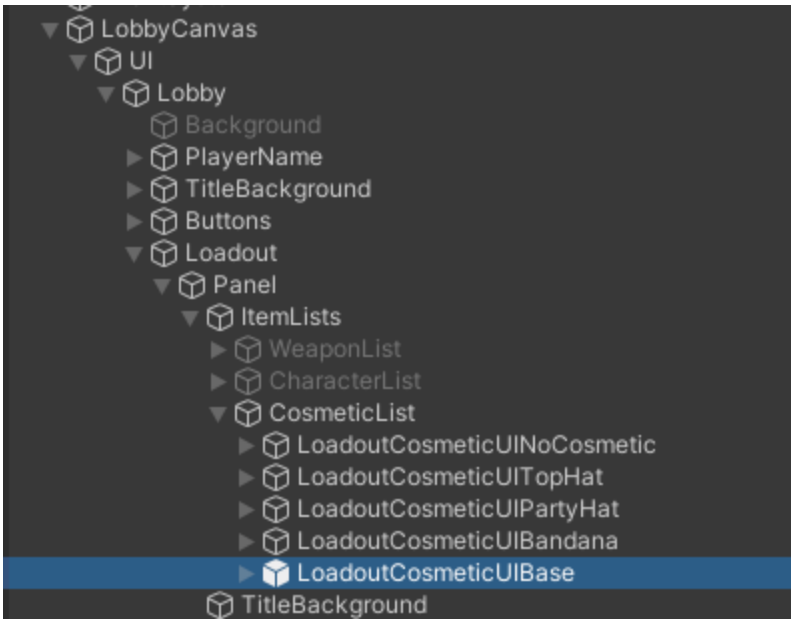
Now go to the **Player Game Data Manager** inspector in the lobby scene and drag and drop the cosmetic config into the cosmetics list.



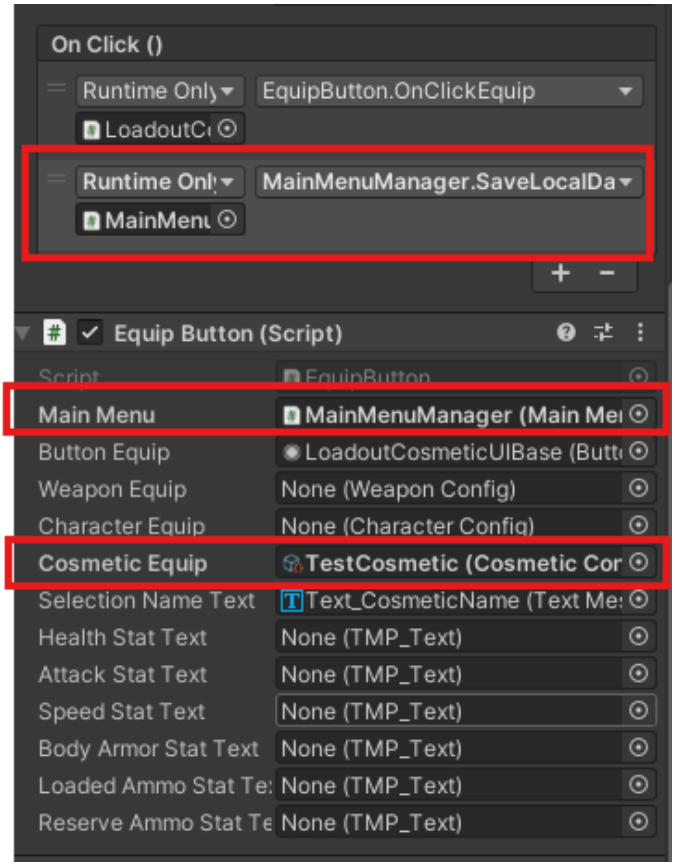
Now lets create the loadout UI for the cosmetic.

Go to the lobby scene and dropdown the **Cosmetics List** in the loadout panel of the lobby canvas.

Drag and drop a **Cosmetic UI Base** prefab into the list from FusionShooterBrawler > Prefabs > UIPrefabs > LobbyUIPrefabs.



In the inspector of the new cosmetic UI base you just added, assign the main menu manager, your cosmetic config data and the add the button event to call the SavePlayerLocalData() method from the main menu manager. Then unpack and customize the Cosmetic UI Base prefab however you'd like.

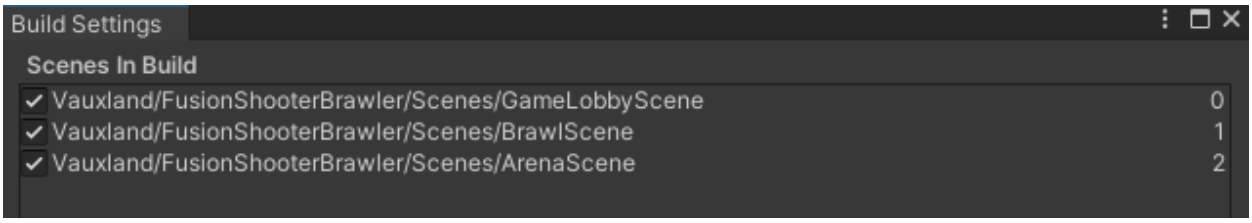


Making a New Map Config

Lets make a new map config

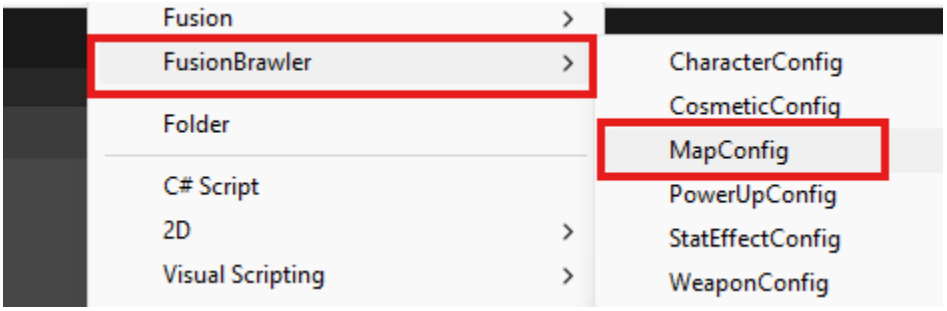
To create a new scene you can refer to the **Match Scene** section of this documentation it explains to you what you will need for a match scene. Or you can look at a current scene in the demo and see what scripts and UI components you will need and how the scene is set up.

After you've made your new scene add it to the build settings in File > Build Settings:

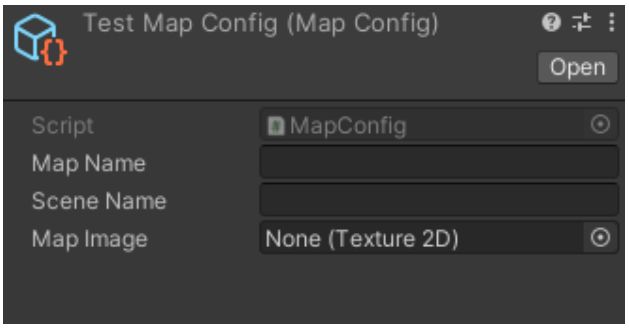


Then open the **MapConfigs** folder in FusionShooterBrawler > Scenes.

Right click in this folder and go to Create > FusionBrawler > MapConfig.



Click on the new map config you just created and fill out the fields in the inspector:

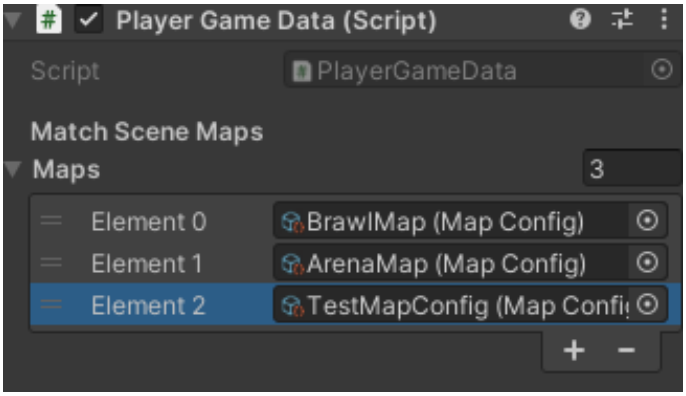


Map Name is the name that will show in the create game panel when choosing a map

Scene Name is the actual name of the scene you made, this will need to be the same name as what you named your scene. This is how it finds the scene to load.

Map Image is the image icon you want to represent your map when navigating through the available maps.

Once you're done go to the lobby scene and click on the **Player Game Data Manager** and add the new map config to the map configs list:



Now your map will show up in the create game panel to choose from.