FPS Kit 3.0 Tutorial

“Replace Soldier Model”

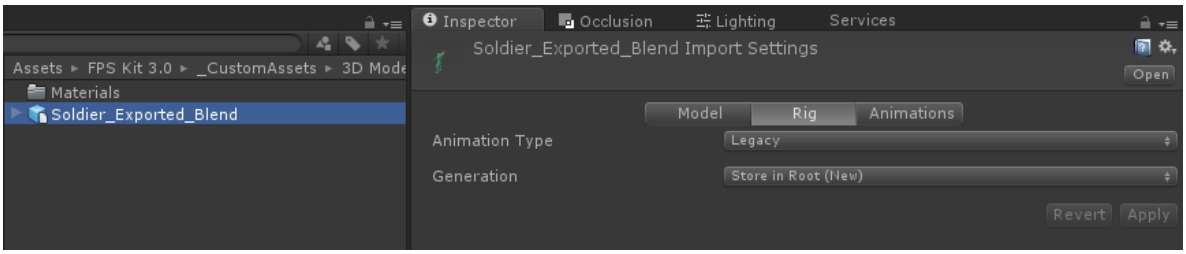
This tutorial will show you how to replace soldier model in FPS Kit 3.0

Before begin make sure your model is fully rigged and have all necessary animations such as walking front/left/right, crouching front/left/right, reloading, jumping, and idle/firing animation for 3 types of weapons (Machine Gun, Pistol and Knife).

Also make sure that model have 2 different textures available to make each team look different.

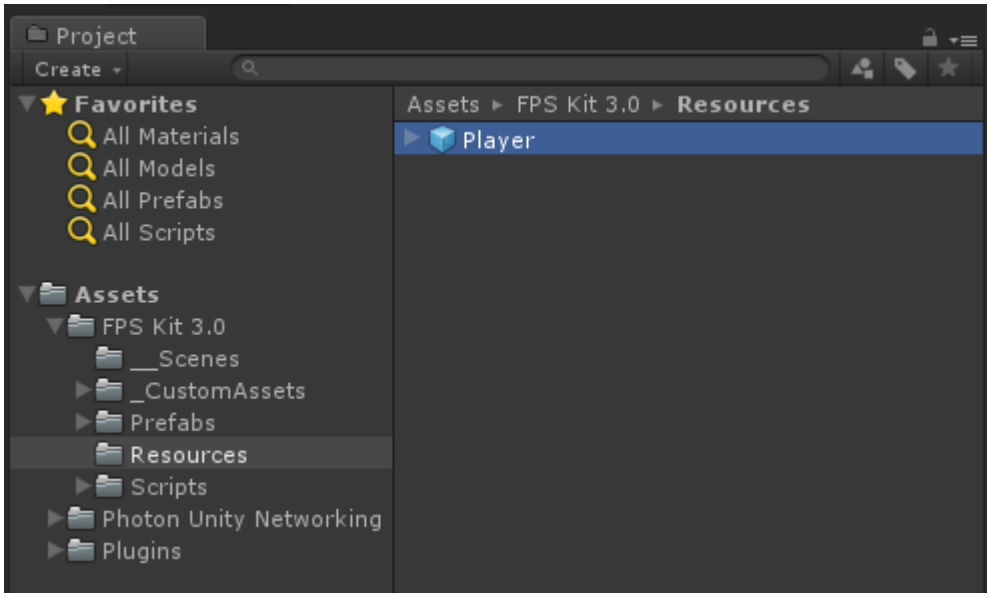
And lastly, since we will be using Legacy animation, make sure that soldier model have proper import settings.

Select model in project view, then in Inspector view select Rig tab and set these settings:

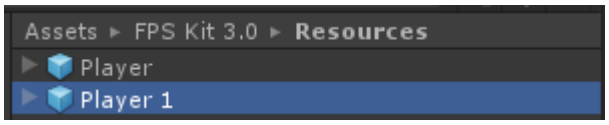


Let's start:

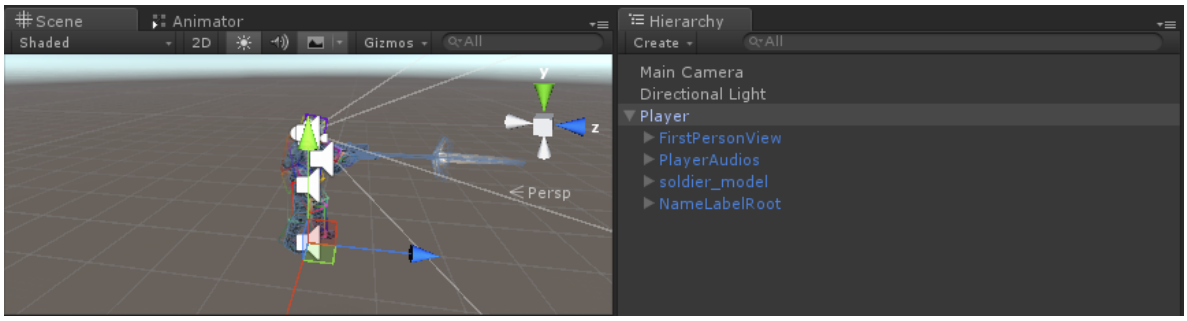
1. First make sure to create backup copy of Player prefab. Go to Assets - > FPS Kit 3.0 -> Resources and select Player prefab



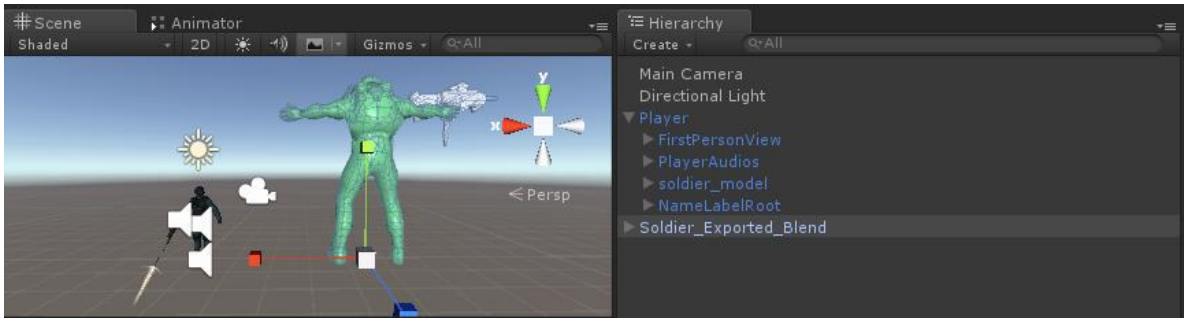
Press Ctrl + D to duplicate it. “Player 1” will be backup copy, so we going to leave it and go back to Player prefab



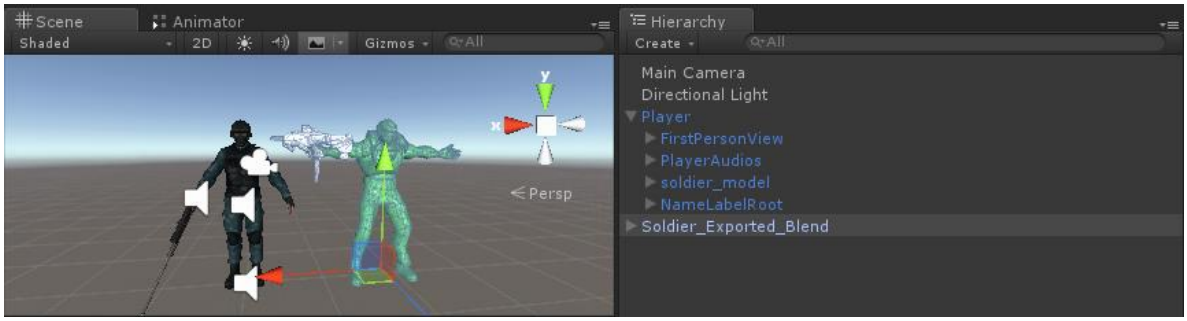
2. Create new scene, then Drag and Drop player prefab



3. Drag and drop your soldier model to Scene view and move it to Player prefab

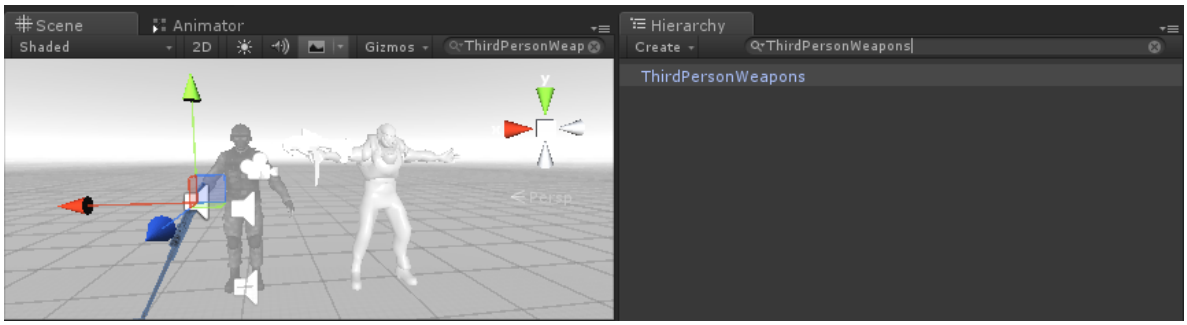


4. In most cases you will notice that new model have different rotation and scale, so we will have to rescale and rotate it to match current model

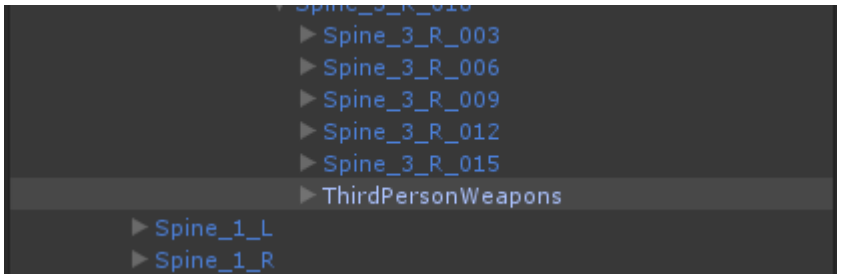


5. Now, before merging new model with Player prefab we need to find an object called "ThirdPersonWeapons”

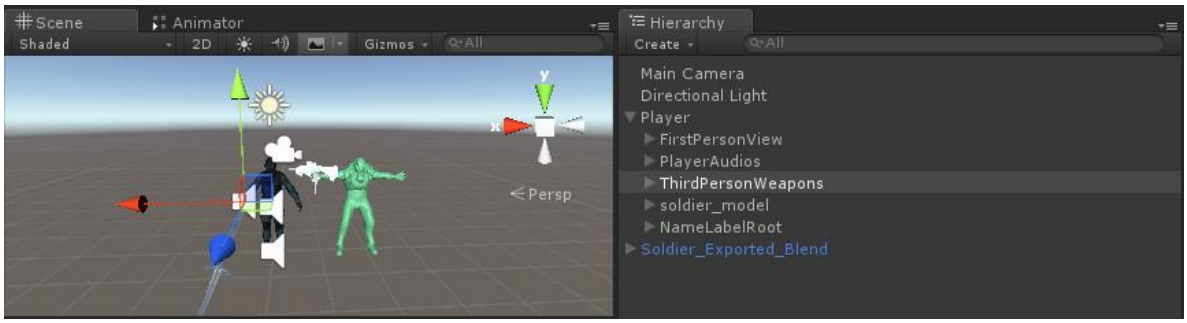
Simplest way to find it is by using Hierarchy search:



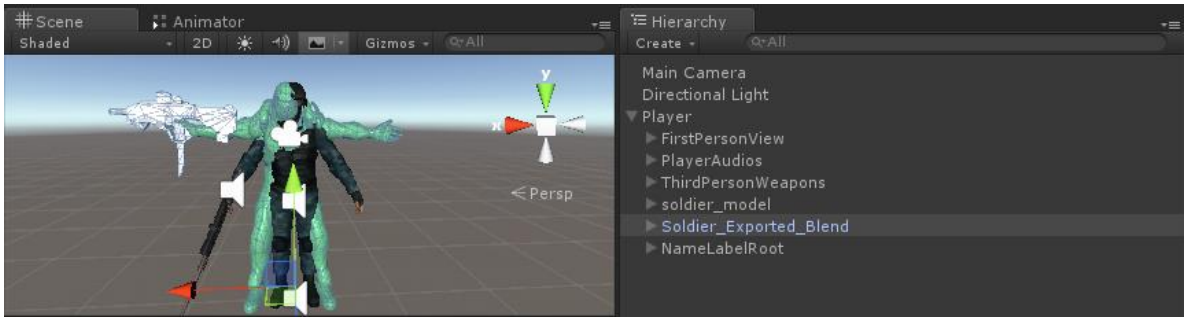
Click on object, then clear search line to go back to normal view



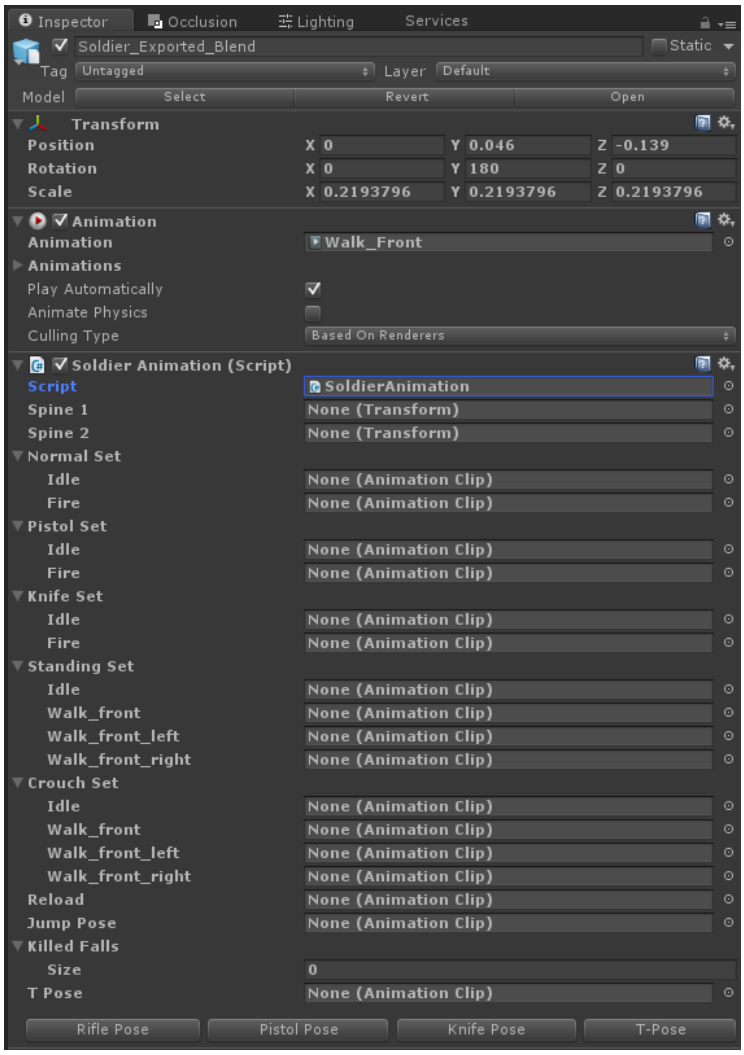
Now we need to un-parent it. Drag the object then Drop outside of old soldier model like this:



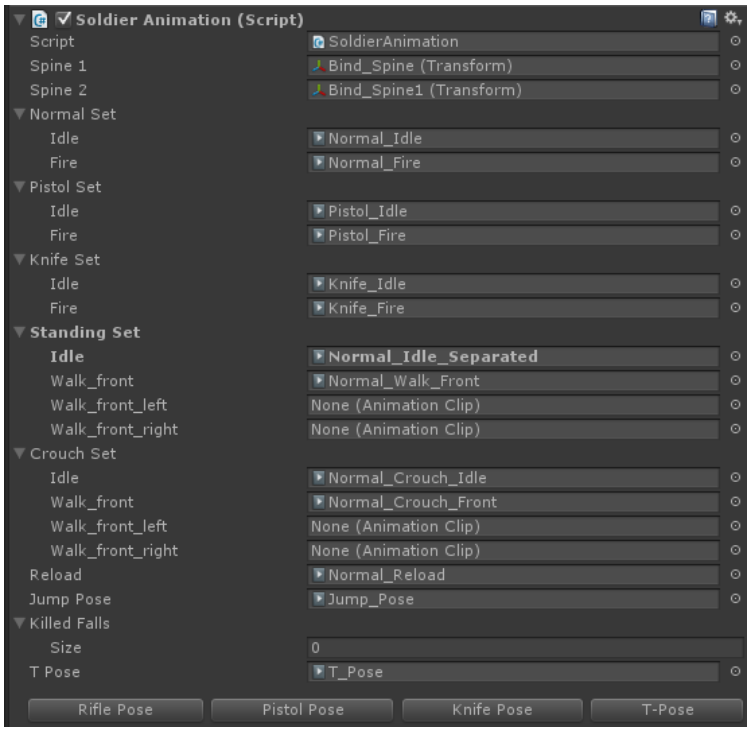
6. Now Drag and Drop new soldier model inside Player prefab then move it exactly where the old soldier model is:



7. Add *SoldierAnimation.cs* component to new soldier model.
   NOTE: It need to be attached to root of the model, near Animation component



8. Now begin assigning all necessary variables, you can use old soldier model as example.

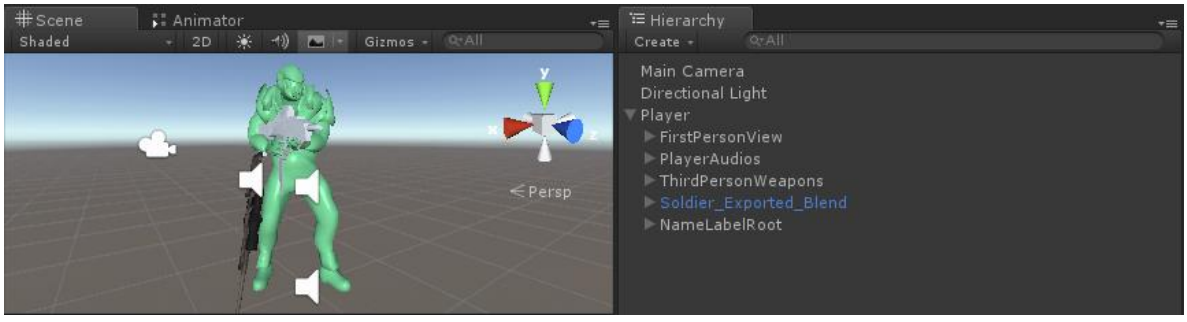


After all variables are assigned, press “Rifle Pose” that way it will be easier to align third person weapons.

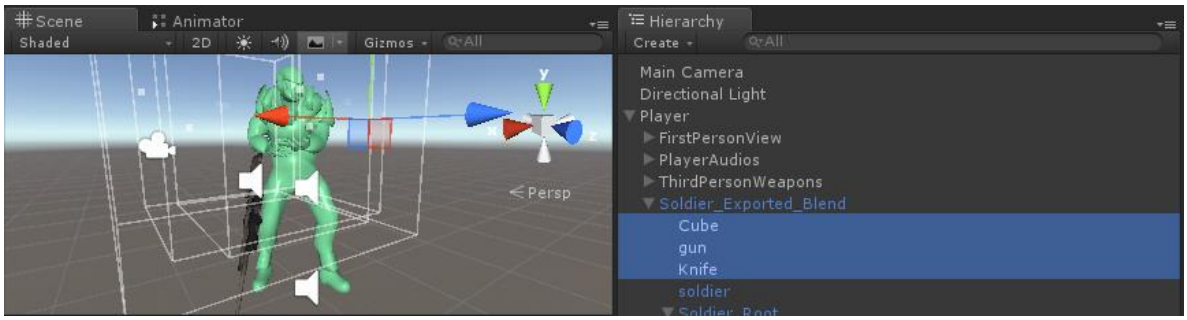
NOTE: If your model does not have all necessary animations, you can leave them unassigned, however for best result it’s preferable to have all the required animations.

Also make sure you do not assign same animation in 2 or more different variables, otherwise it will not work properly.

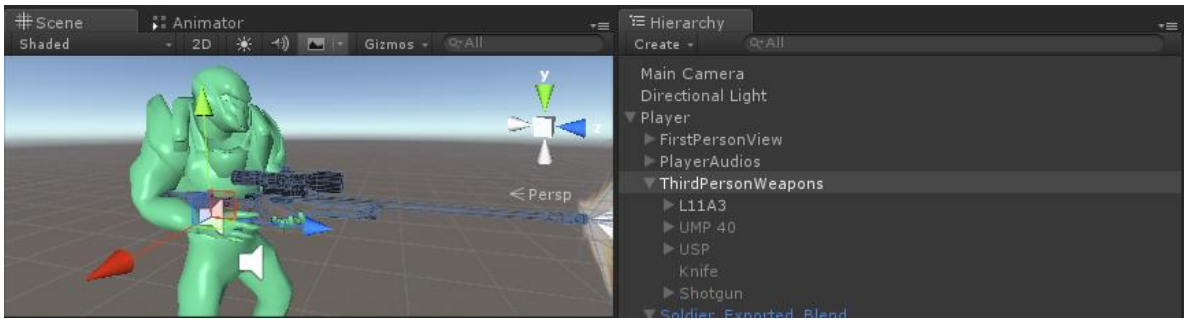
9. Now we can remove old soldier model



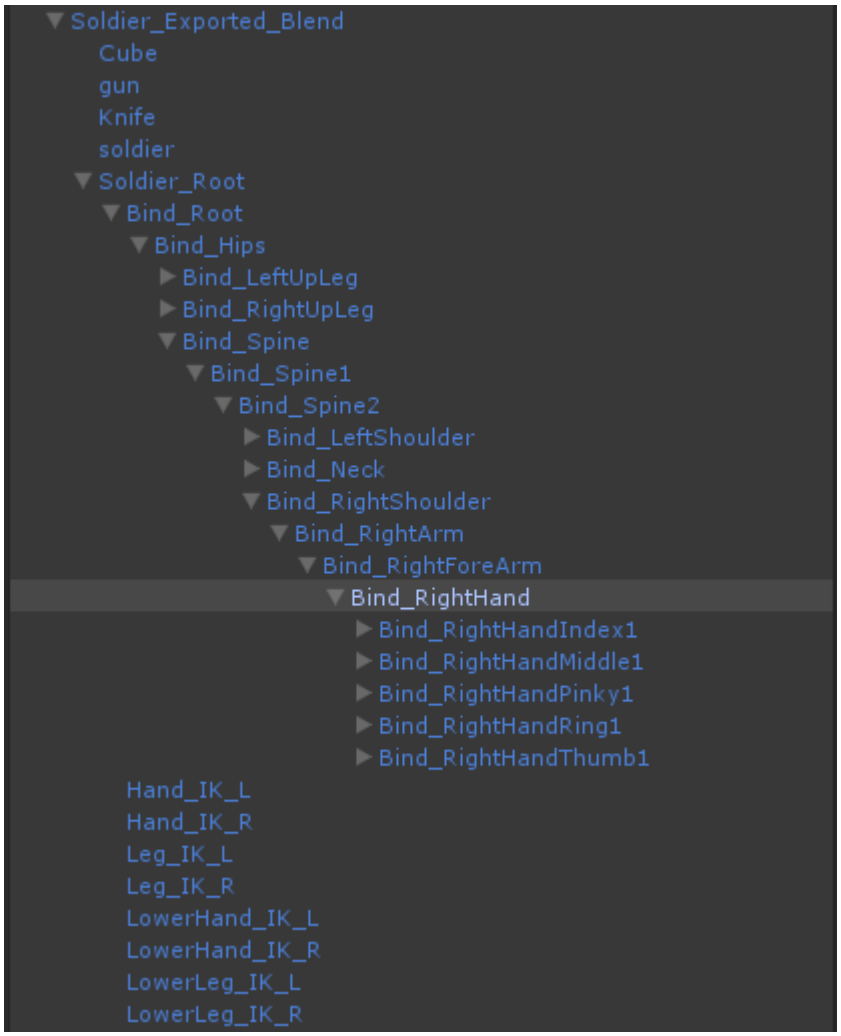
NOTE: In case your model have placeholder weapons, select them and disable Renderer component for each so they become invisible



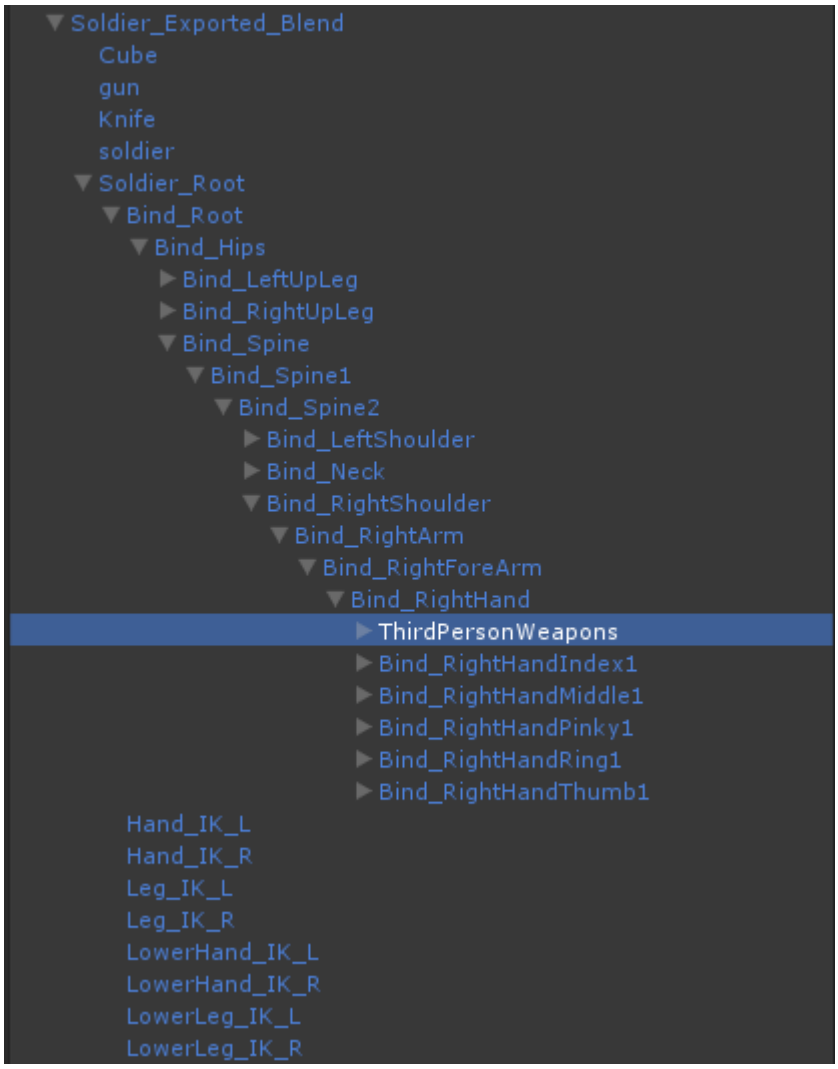
10. Select “ThirdPersonWeapons” object and align it with soldier hands



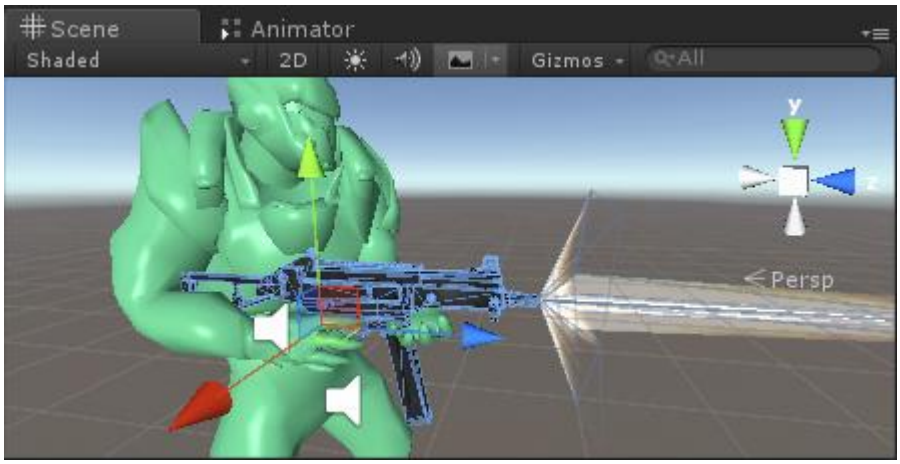
11. Search through soldier model “bones” and select the one that represent right wrist area



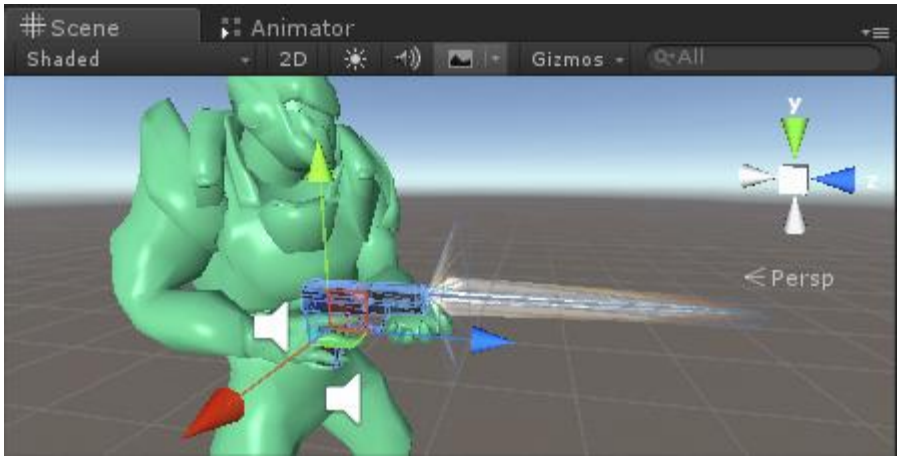
12. Make “ThirdPersonWeapons” child of that Transform



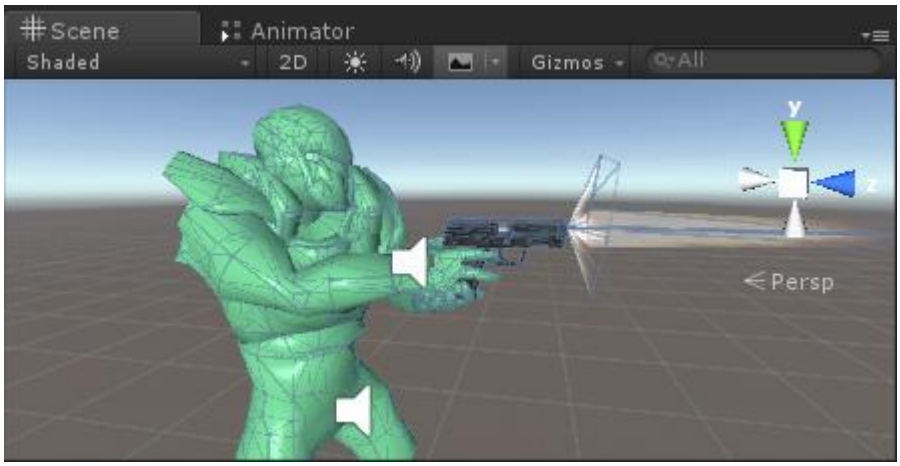
13. Now align the rest of the weapons. After expanding “ThirdPersonWeapons” object you will see that it have many child object, each one represent a weapon. Select active object and deactivate it, then select next one and activate it



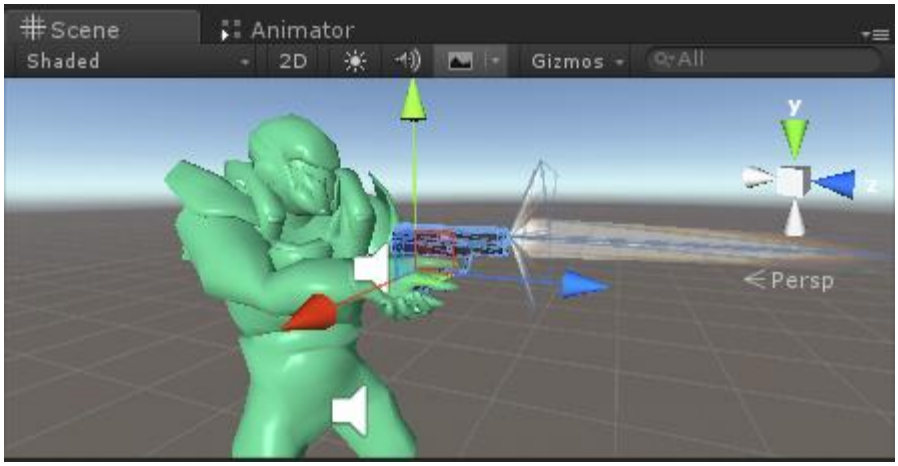
NOTE: For some weapons you will need to change standing pose to align it properly



So select Soldier model root where *SoldierAnimation.cs* was attached and press "Pistol Pose" button

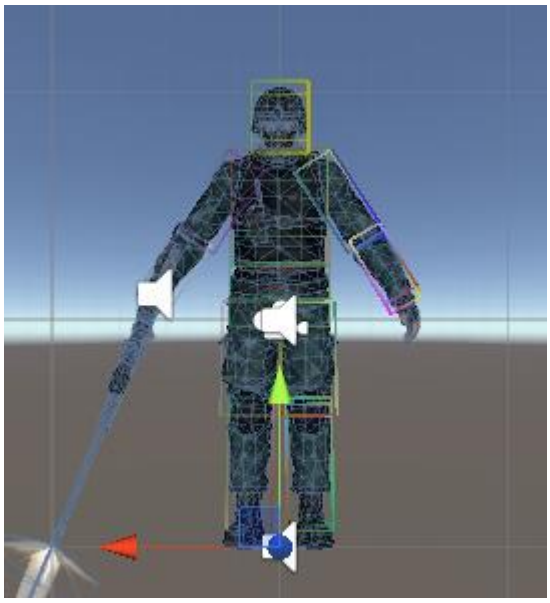


Need to align a bit

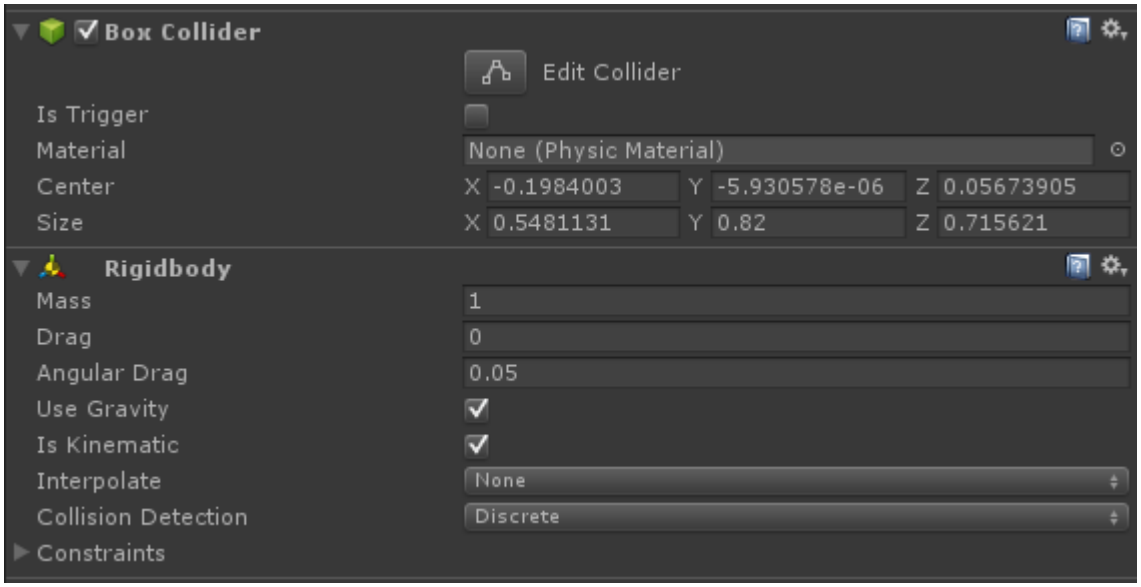


That's better!

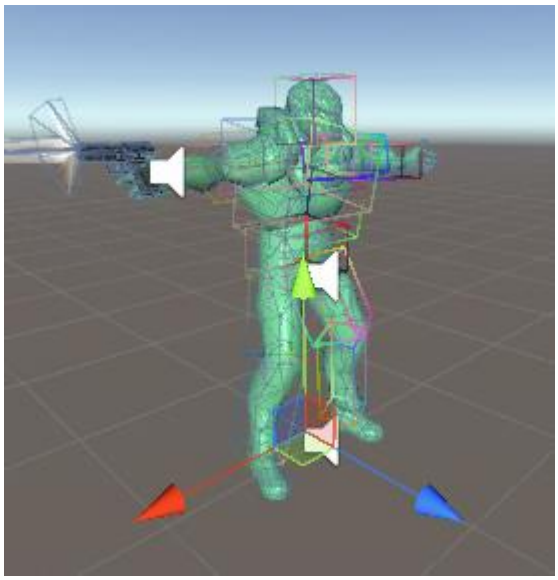
14. After all weapons are aligned we can move to Hit Boxes.
    Hit Boxes are the Box Colliders assigned to major "bones". They are used to receive bullet damage in game.
    Here is example from old soldier model:



NOTE: It's important to also assign Rigidbody component near each Box Collider and mark it as Kinematic

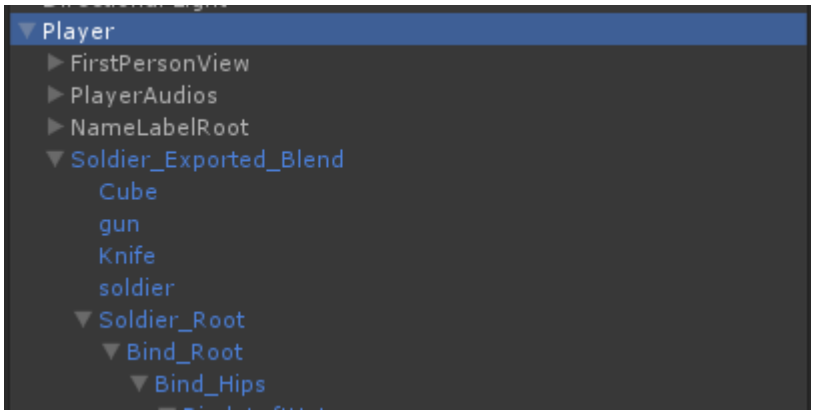


15. So let's begin creating Hit Boxes for our new soldier model

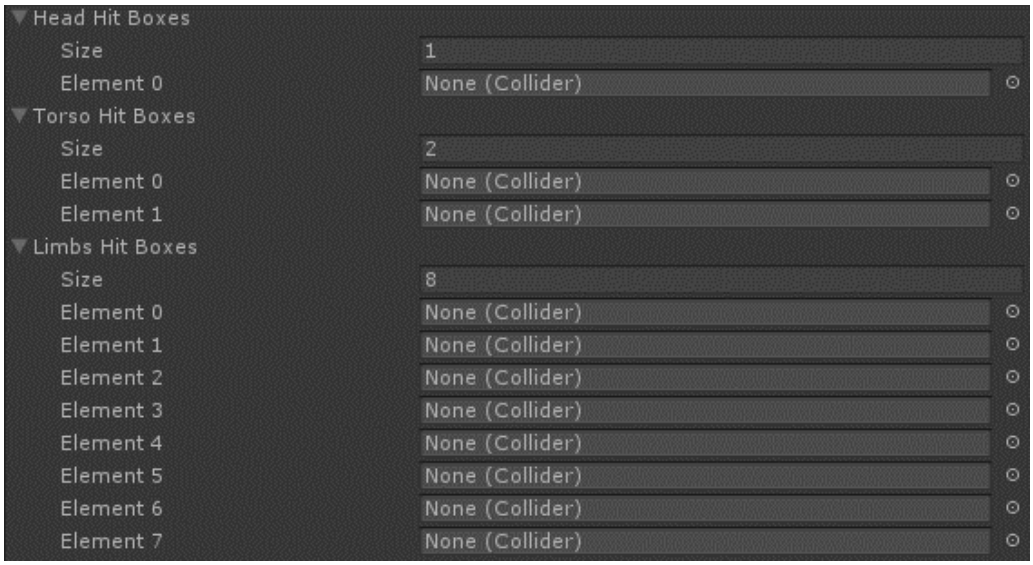


Done!

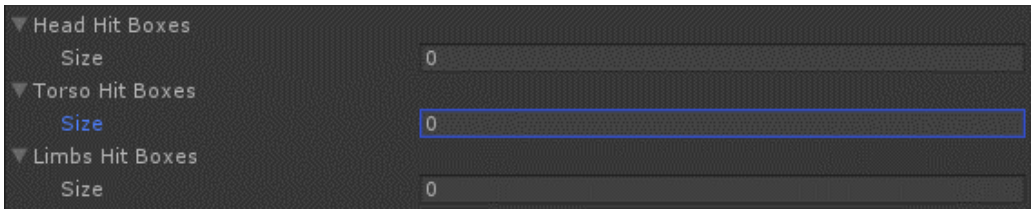
16. Assign new hit boxes to respective variables.
    Select Player root object



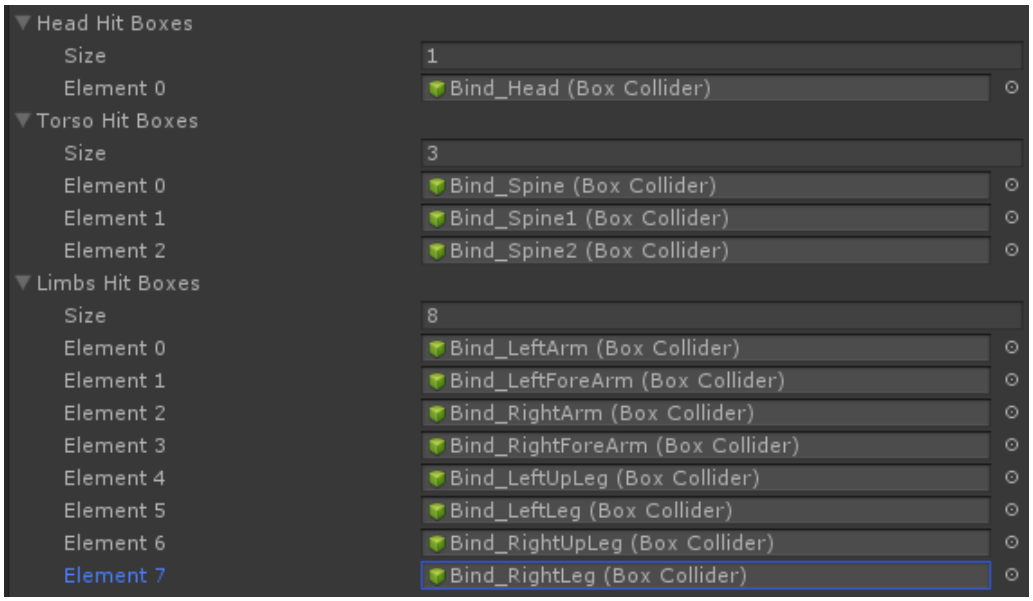
In Inspector view under *PlayerNetwork.cs* component you will see 3 arrays “Head Hit Boxes”, “Torso Hit Boxes” and “Limbs Hit Boxes”



Set length of each Array to 0



Then begin assigning new Hit Boxes (Drag and Drop each object to respective section)

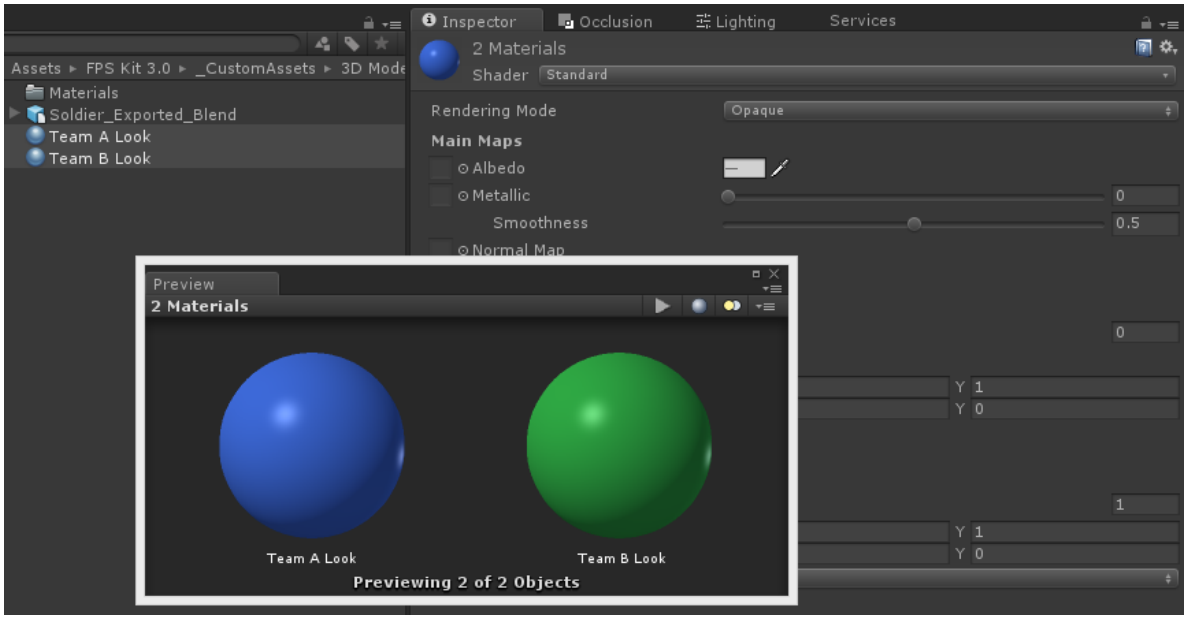


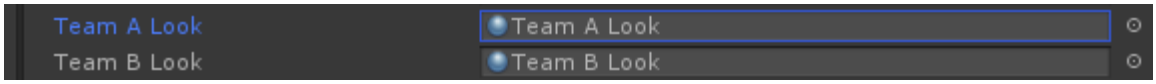
17. Under *PlayerNetwork.cs* we also need to assign couple more components

Soldier Animation – is a component that we previously attached to new soldier model, Drag and Drop on variable

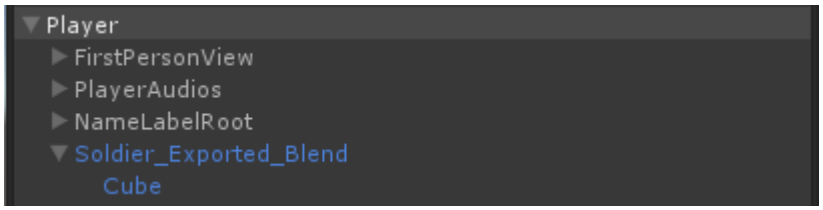
Soldier Body – is a SkinnedMeshRenderer of our soldier

Team A Look, Team B Look – for this you need to create 2 materials and assign a different variation of your soldier texture. If your model only have one texture you can assign it to both materials and just change color on each material





18. Now it's time to save our soldier back to prefab. Select Player object in scene view



In Inspector view click Apply to save it back to Player Prefab´

Done! Now new player prefab is ready for testing.