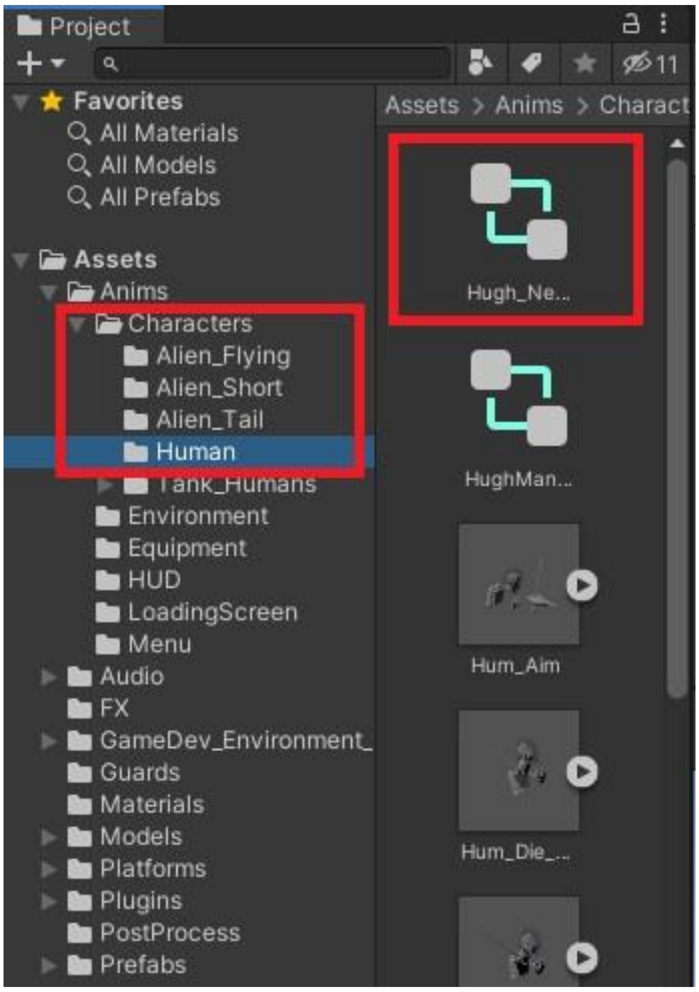
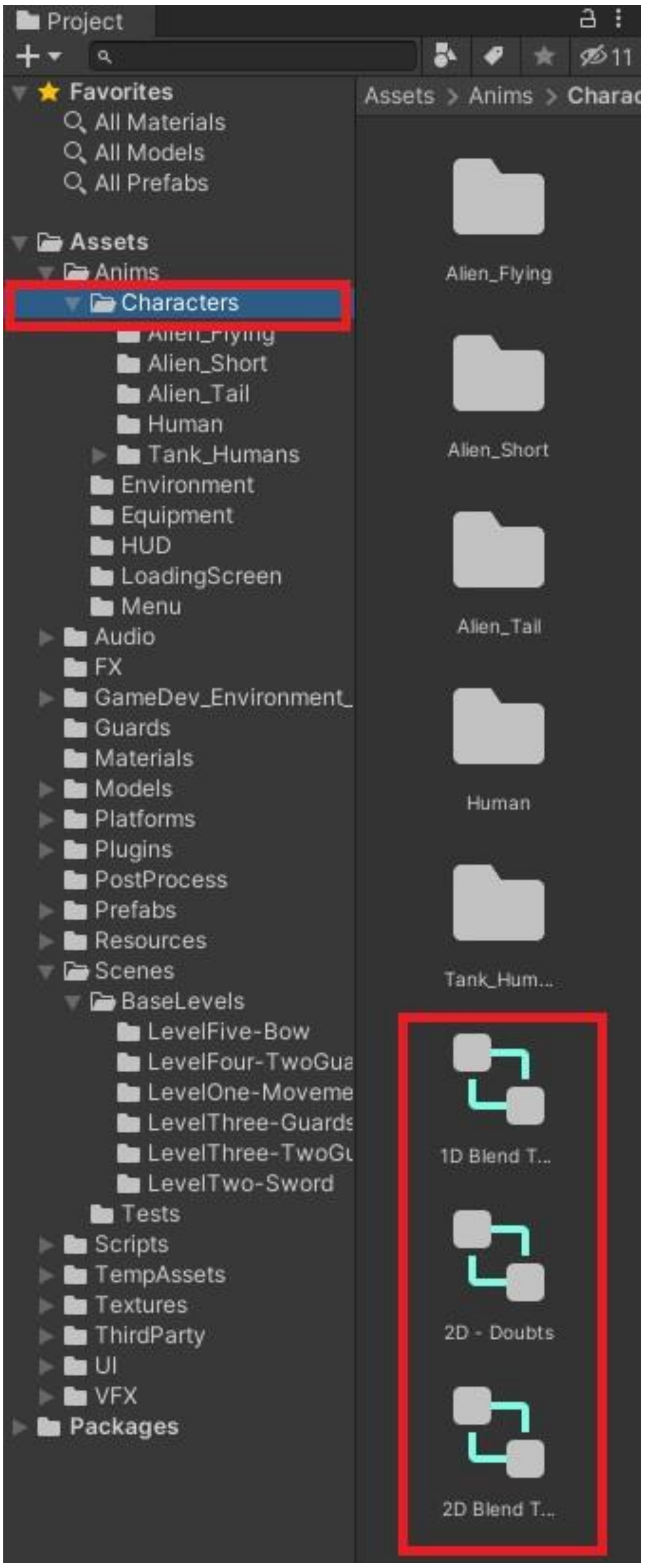
Sprint 03

Assignment 03: Character Blend Trees

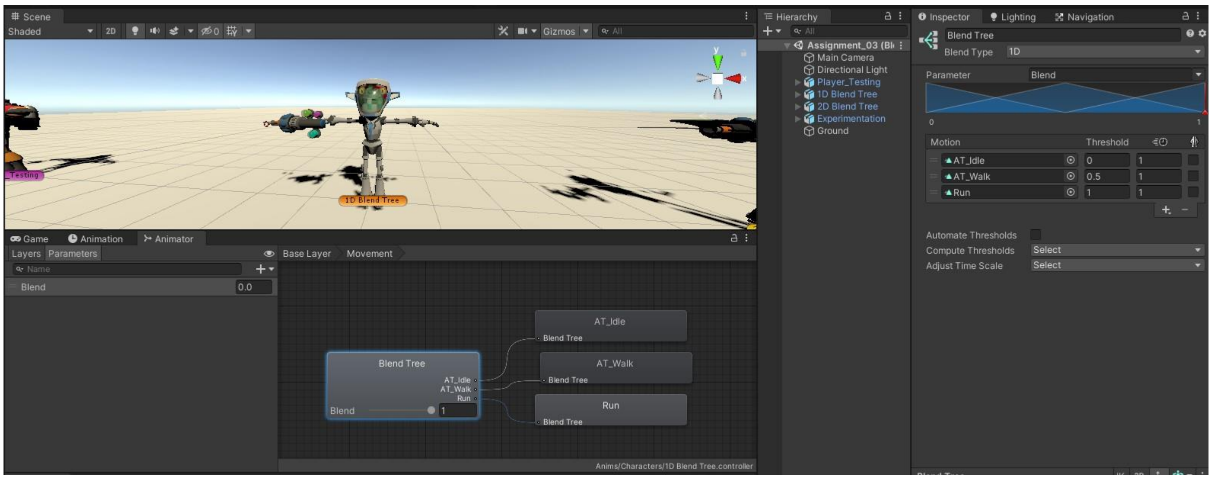
Step 01: Setup

- Location of files I created.




Project
Favorites
All Materials
All Models
All Prefabs
Assets
Anims
Characters
Alien_Flying
Alien_Short
Alien_Tail
Human
Tank_Humans
Environment
Equipment
HUD
LoadingScreen
Menu
Audio
FX
GameDev_Environment_
Guards
Materials
Models
Platforms
Plugins
PostProcess
Prefabs
Resources
Scenes
BaseLevels
LevelFive-Bow
LevelFour-TwoGua
LevelOne-Moveme
LevelThree-Guards
LevelThree-TwoGu
LevelTwo-Sword
Tests
Scripts
TempAssets
Textures
ThirdParty
UI
VFX
Packages
Assets > Anims > Charac
Alien_Flying
Alien_Short
Alien_Tail
Human
Tank_Hum...
1D Blend T...
2D - Doubts
2D Blend T...

- I first created Empty Scene and in there, load "*TailGuard*" Model.
- Then I created a 1D Blend Tree named it as along.
- Then added a "*Forward*" float parameter.
- Then added 3 Clips, "*Idle, Walk & Run*", with respective 0, 0.5 & 1 Threshold values.

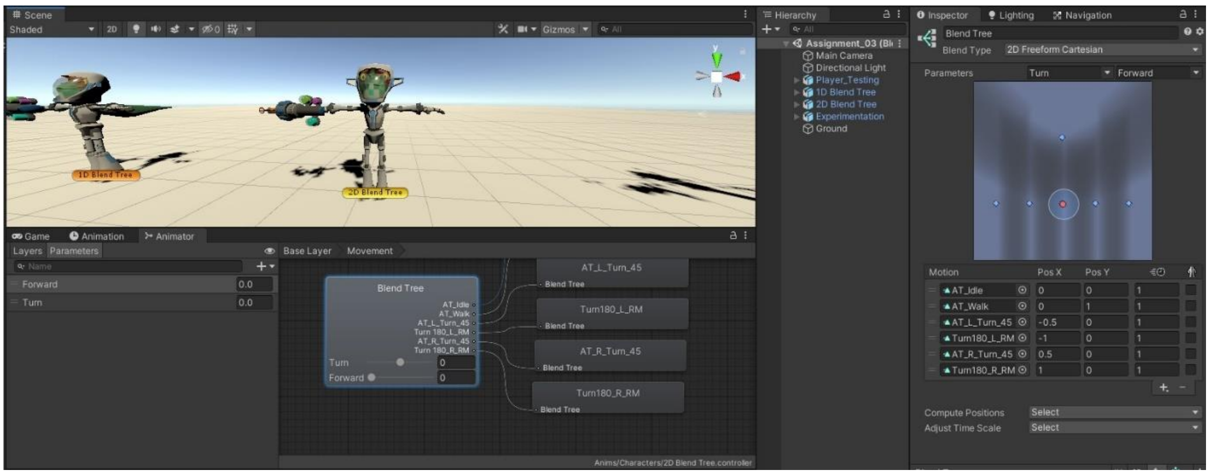


Blend Tree
Blend Type 1D
Parameter Blend
Blend 0.0

Motion	Threshold		
AT_Idle	0	1	
AT_Walk	0.5	1	
Run	1	1	

Automate Thresholds
Compute Thresholds Select
Adjust Time Scale Select

- Then added 3 Clips, "*Idle, Walk & Run*", with respective 0, 0.5 & 1 Threshold values.
- Next for the 2D Cartesian Blend Tree, I created another float parameter "*Turn*".
- And did the setup like this:



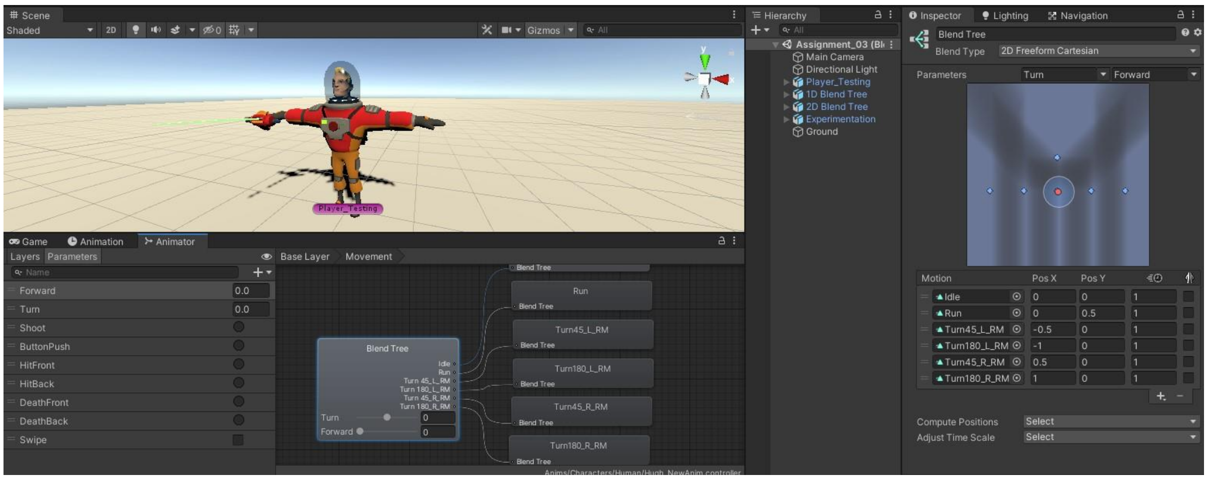
Blend Tree
Blend Type 2D Freeform Cartesian
Parameters Turn Forward
Forward 0.0
Turn 0.0

Motion	Pos X	Pos Y		
AT_Idle	0	0	1	
AT_Walk	0	1	1	
AT_L_Turn_45	-0.5	0	1	
Turn180_L_RM	-1	0	1	
AT_R_Turn_45	0.5	0	1	
Turn180_R_RM	1	0	1	

Compute Positions Select
Adjust Time Scale Select

- Here, we also put in some values to the "*Pos X*" as we want to move the Character with respective to that Axis, as "*Pos Y*" is our Forward Axis, i.e. "*SpeedZ*".
- Next, I did the same thing for our Player, "*HughhMan*":

- I duplicated the "*HughMann Animator*" and in the 2nd one added a Blend Tree instead of the Idle > Run states.
- The setup for the Player's Animator is same as previous one.

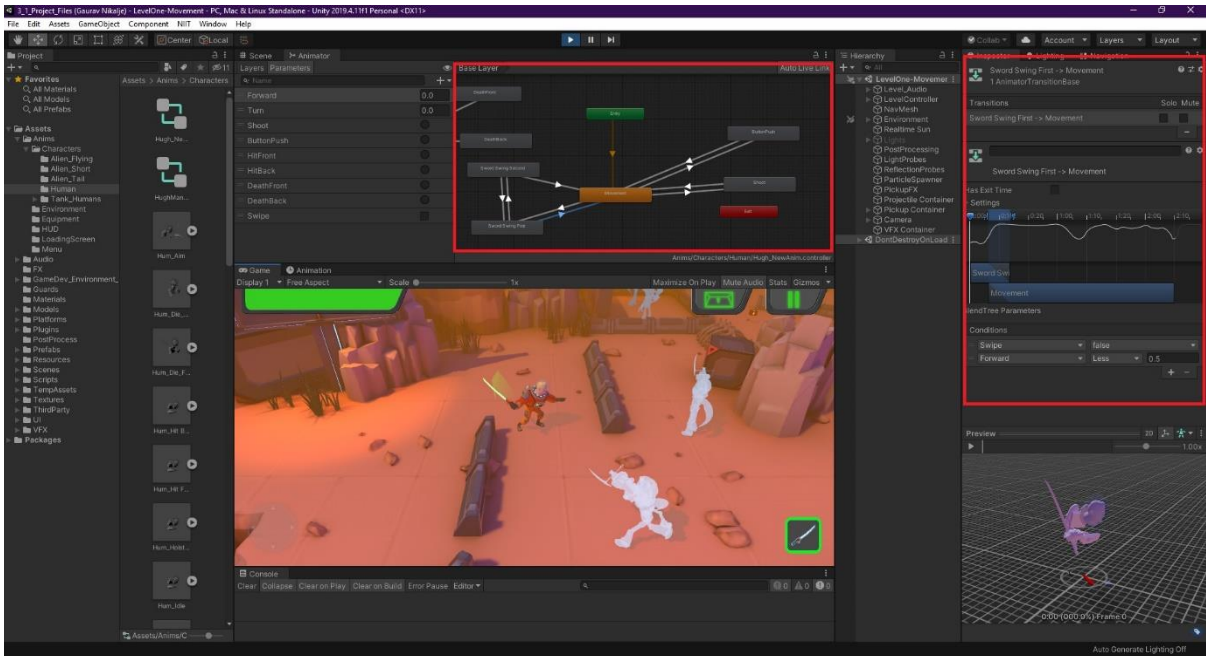


Blend Tree
Blend Type 2D Freeform Cartesian
Parameters Turn Forward

Motion	Pos X	Pos Y	
Idle	0	0	1
Run	0	0.5	1
Turn45_L_RM	-0.5	0	1
Turn180_L_RM	-1	0	1
Turn45_R_RM	0.5	0	1
Turn180_R_RM	1	0	1

Compute Positions Select
Adjust Time Scale Select

- Then, for the "*SwordSwipe, Shoot & ButtonPress*" states I added the respective transitions, with "*ExitTime*" set to uncheck/false, as we want instant transition to those states.
- When Entering from Movement to any other States, I added the respective Boolean or Trigger conditions set to True.
- When Returning from Movement to any state, I added the Forward value Less than "*0.5*" and the Boolean or Trigger set to False.



Sword Swing First -> Movement

Conditions

Swipe	false	
Forward	Less	0.5

Step 02: What I leant

- **Describe a situation where it might be useful to use a 1D blend tree.**

  - 1D Blend Tree operates on Vector 1 Platform.
  - It is used in Games where there is simple Player Movement in 1 Direction.
  - The movement or Rotation of the Player is achieved through scripts.
  - 1D Blend Trees are usually used in 2D or 3D Platformer Games where the Player movement is focused on a single axis direction, i.e., Forward and/or Backward or Left & Right.
  - Games like Mario, Sonic: The Hedgehog, Temple Run, Subway Surfers, etc.
  - Mostly will be useful in 2D Platformer games.

- **Explain why root motion animations are useful, and a situation where we'd want to avoid them.**

- Root Motion is an animation where the Object is moved in a 3D World Space without any contribution of external components like Scripts or Physics Forces.
- Root Motion when turned On in Unity, forces the Character or Object to move in World Space, and when turned Off, makes it static, and performs the animation at one place.
- Root Motion are useful where we want proper animation in Games, and where the Animations play an important role.
- An Example of this would be a Game where the Character moves or takes a 45 Degree or 90 Degrees Turn to turn Left or Right.

- **What's the difference between a 2D Freeform Directional blend tree and a 2D Freeform Cartesian Blend Tree?**

  - 2D Freeform Directional uses both the parameters as Directional Axes, while 2D Freeform Cartesian uses 1st for Direction (X Pos) and 2nd as Rotation (Y Pos).
  - Both support multiple animations on same axis.
  - In, 2D Freeform Directional, blending is calculated with respect to 2 different directions, while in 2D

Freeform Cartesian, blending is calculated with respect to direction as well as rotation.

- Example for 2D Freeform Directional: Walk, Walk & Strafe Left, and son on…
- Example for 2D Freeform Cartesian: Walk, Walk & Turn Right (45, 90 or so degrees) and so on…
- Best use of 2D Freeform Directional will be in FPS Games like CS: GO and Valorant, where the Player strafes Left or Right instead of rotation, as these games focus on rapid player movements.
- Best use of 2D Freeform Cartesian will be in Open-World or RPG Games like GTAV, Assassin's Creed or Sleeping Dogs where the Player Turns or Rotates towards Left or Right with certain Angle, as these games focus on more dynamic or realistic player movements.
- In 2D Freeform Cartesian, Root Motion can play an important role, as for giving precise transformations while turning, while in 2D Freeform Directional Root Motion doesn't make much difference and solely depends on the use case decided by the developers.

-----------------------THE END---------------------