Top-down ARPG Camera & Character Controller (Version 1.1)

Summary

Thank you for purchasing this package. (Version 1.1)

The demo scenes provided are simple demonstration on how to use the scripts. It can help you if you are struggling with some features and not sure how to fix them. If you want the transparent objects or if the player moves to a border (or any collider) in your scene there is a setup guide included in the document on how to create the obstacles.

From version 1.1 there's 8 prefabs you can use:

- "ARPG Player" Normal top-down camera and uses mouse to move character (Like Diablo).
- "ARPG Follow Cam Player" Follow camera and uses mouse to move character (Like Diablo).
- "1.Axis Mouse ARPG Follow Cam Player" Follow camera and uses keys and mouse rotation to move.
- "1.Axis Mouse ARPG Player" Normal top-down camera and uses keys and mouse rotation to move.
- "2.Axis top down Mouse ARPG Follow Cam Player" Follow camera and uses keys and mouse position to move.
- "2.Axis top down Mouse ARPG Player" Normal top-down camera and uses keys and mouse position to move.
- "3.Axis ARPG Player" Normal top-down camera and uses keys to move.
- "4.Axis Turn ARPG Follow Cam Player" Follow camera and uses keys to move/rotate.

The Demo's prefabs are also included if you want to use both camera scripts in one scene.

There are two camera scripts and 4 scripts that move the character. The camera scripts consist of a normal top-down ARPG camera and a follow camera.

Scripts

The normal top-down ARPG camera (ARPGCameraController.cs)

A Script for an ARPG (Diablo, Path of Exile, Torchlight) style camera that also allows rotation on the y and x axis. It contains code that makes objects transparent if they are between the camera and target. The objects material needs to be a transparent shader so we can change the alpha value. Just drag the script on the main camera and set the parameters. Camera can be a child of the target.

Parameters:

- Target, the target the camera should look at and rotate around. You need a target (Character) for the camera to work.
- StartingDistance, distance the camera starts from target object.
- MaxDistance, the max distance the camera can be from target object.
- MinDistance, the min distance the camera can be from target object.
- ZoomSpeed, the speed the camera zooms in/out of the target.
- TargetHeight, the amount from the target object pivot the camera should look at.
- CamRotationSpeed, the speed at which the camera rotates.
- CamXAngle, the camera x Euler angle. This is the angle the camera looks at from the top.
- FadeObjects, enable objects of a certain layer to be faded.
- LayersToTransparent, the layers where we will allow transparency.
- Alpha, the alpha value of the material when player behind object.

To rotate around target you press the “Fire 3” button that can be changed in the input settings and moving mouse left/right.

The mouse scroll wheel is used to zoom in/out.

To change the rotation you look at press the “Fire 3” button that can be changed in the input settings and moving mouse up/down.

Follow camera (ARPGFollowCameraController.cs)

A Script for a follow style camera that allows rotation on the x axis and to zoom. It contains code that makes objects transparent if they are between the camera and target. The objects material needs to be a transparent shader so we can change the alpha value. Just drag the script on the main camera and set the parameters. Just drag the script on the main camera and set the parameters. Camera cannot be a child of the target.

Parameters that's different from former script:

- RotationDamping, how fast it should rotate to target angles.

The mouse scroll wheel is used to zoom in/out.

To change the rotation you look at press the "Jump" button that can be changed in the input settings and moving mouse up/down.

Player ARPG move script (ARPGPlayerMove.cs)

This script controls the player movement. Simple ray cast gets the target location and ignore layers provided, and then the player moves to target location stopping if it hits something. Can be easily modified to work with Unity Pro Nav Mesh or path finding projects.

This script requires Character Controller script, also uses an Animator but is not required.

Parameters:

- MoveSpeed, character movement speed.
- RotationSpeed, how quick the character rotates to target location.
- DistanceError, the distance where you stop the character, distance calculated between the difference of target position and character position.
- Gravity, gravity for the character.
- RayCastDistance, the ray casting distance of the mouse click.
- LayerMask, the layers the mouse click (ray cast) should ignore.
- GuiRect, the rectangle the HUD or GUI is in, will stop ray cast at that rectangle.
- GuiRectYFromBottom, if the HUD or GUI is calculated from Screen.heigt – y.

One can move the player by pressing "Fire1" or "Fire2" button that can be changed in the input settings on the desired location. The desired location must have a collider to hit like terrain.

Version 1.1 added two new public methods:

- MoveToPosition(Vector3 target)
- TeleportToPosition(Vector3 target)

Move player with keys and mouse rotation (AxisMousePlayerMove.cs)

This script controls the player movement. Use vertical axis to move forward and back, horizontal axis to strafe left and right, mouse to rotate the player. One can change keys in input settings.

This script requires Character Controller script, also uses an Animator but is not required.

Parameters:

- MoveSpeed, character movement speed.
- RotationSpeed, how quick the character rotates with mouse movement.
- Gravity, gravity for the character.
- JumpSpeed, jump height of the character.

Works best with top-down and follow camera.

Move player with keys and mouse position (AxisMouseTopDownPlayerMove.cs)

This script controls the player movement. Use vertical axis to move forward and back, horizontal axis to strafe left and right, mouse position on terrain to rotate the player. One can change keys in input settings.

This script requires Character Controller script, also uses an Animator but is not required.

Parameters that's different from former script:

- RayCastDistance, the distance the ray should check for mouse position.
- LayerMask, the layers the RayCast should ignore.

Works best with top-down and follow camera.

Move player in direction with keys (AxisPlayerMove.cs)

This script controls the player movement. Use vertical and horizontal axis to move in the desired direction. One can change keys in input settings.

Parameters that's different from "AxisMousePlayerMove.cs":

- MyRotationObject, the object we will be rotating when moving

Works best with top-down camera.

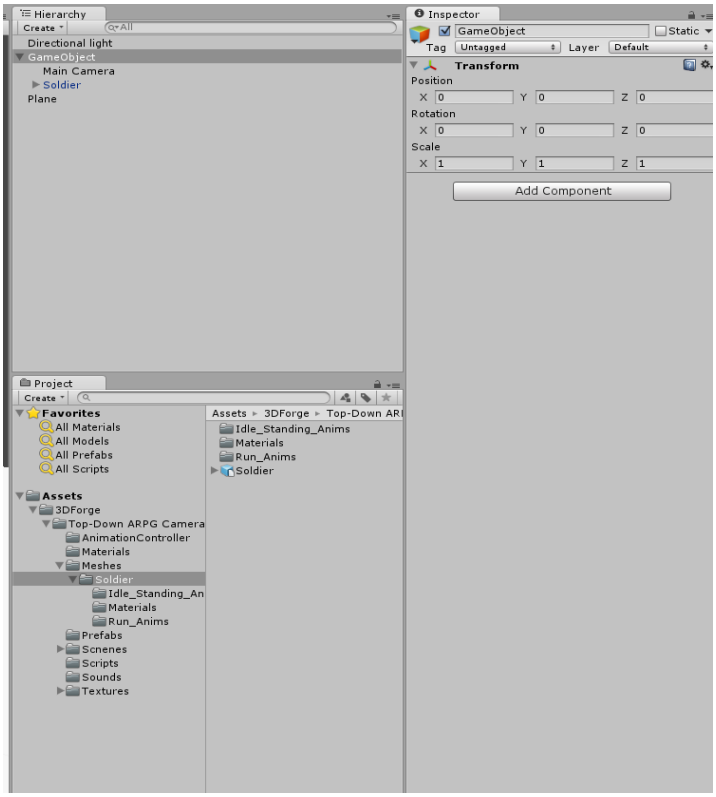
Move and rotate player with keys (AxisTurnPlayerMove.cs)

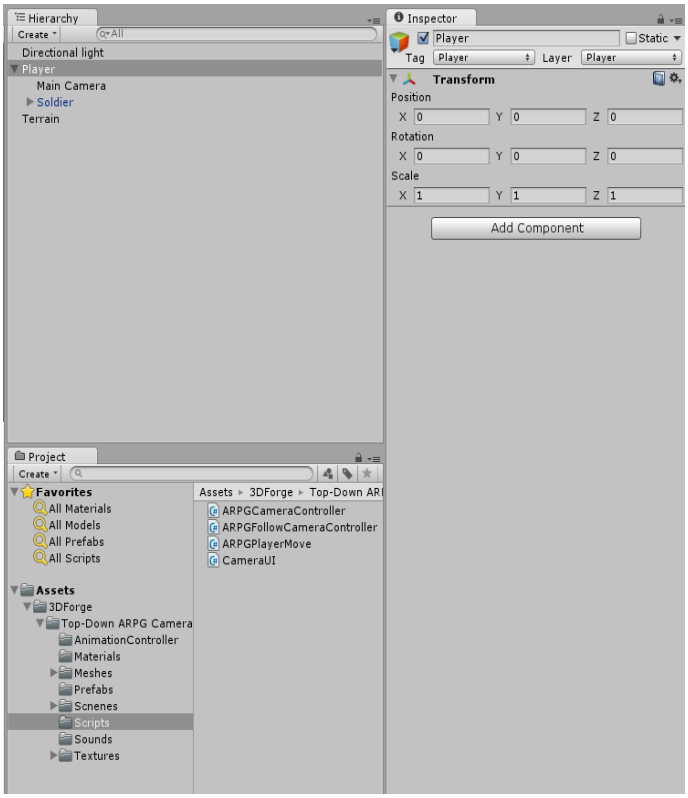
This script controls the player movement. Similar to "AxisMousePlayerMove.cs", just with keys.

Setup 1

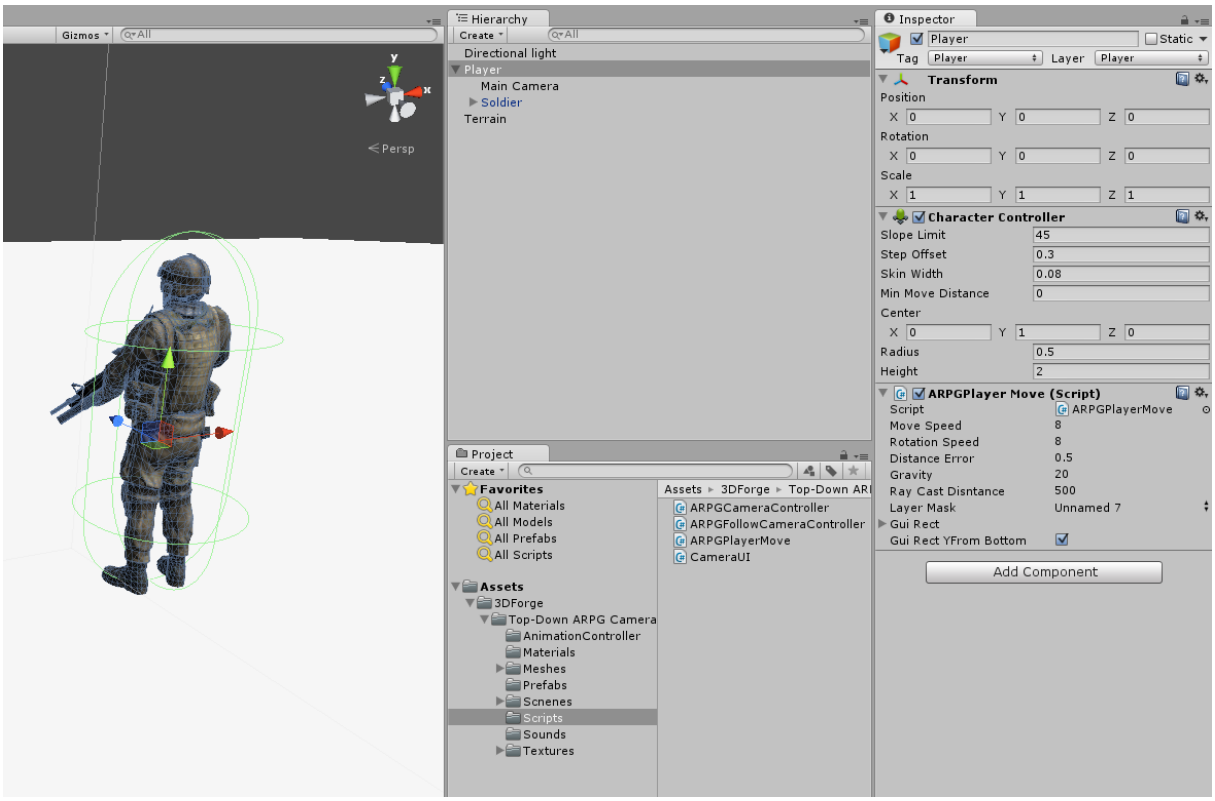
Setup guide for ARPG click and move, if you use the prefabs provided you can skip to step 10.

1. Create a new scene add a plane/terrain to the scene.
2. Create empty GameObject and move its position to (0, 0, 0).
3. Add your toon/character to the scene and move its position to (0, 0, 0);
4. Add your toon/character that is in the scene as a child object of the GameObject we added.
5. Also drag the mainCamera as a child object of the GameObject we added.
6. Now we can rename the GameObject and set its tag and layer (See 11) as player.

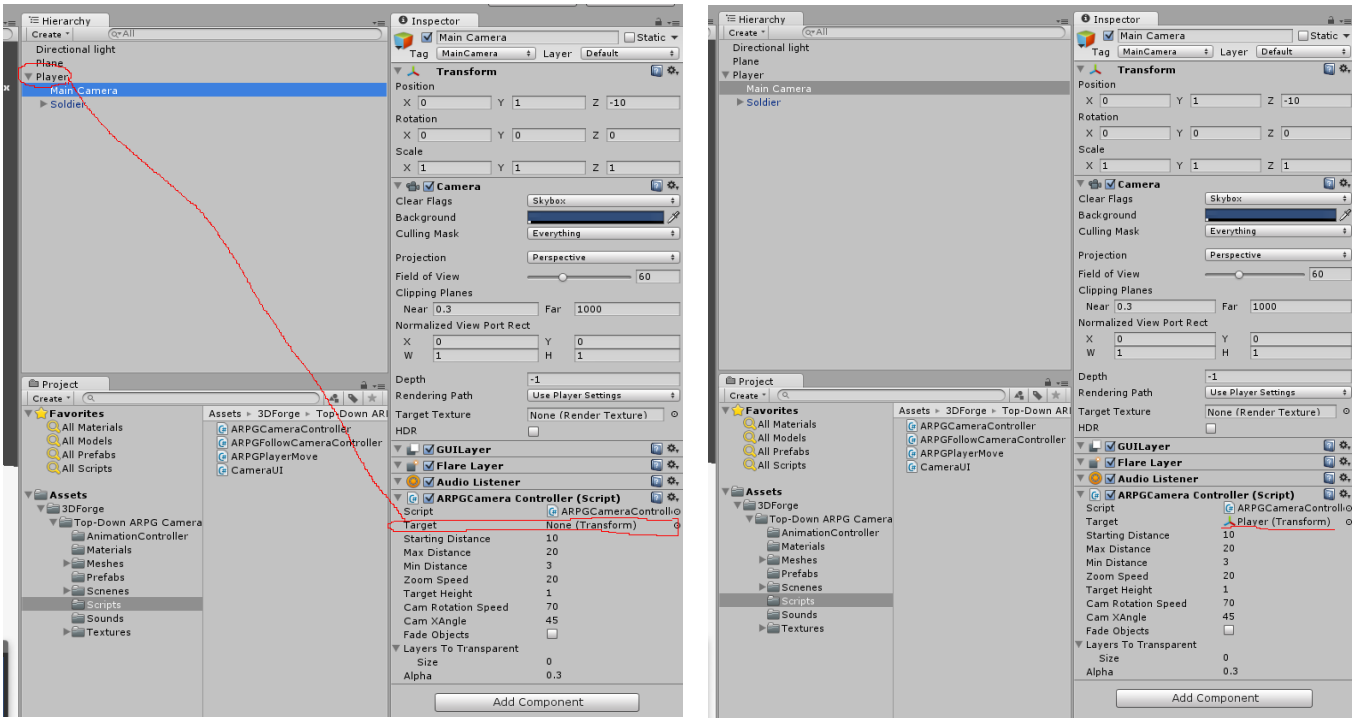




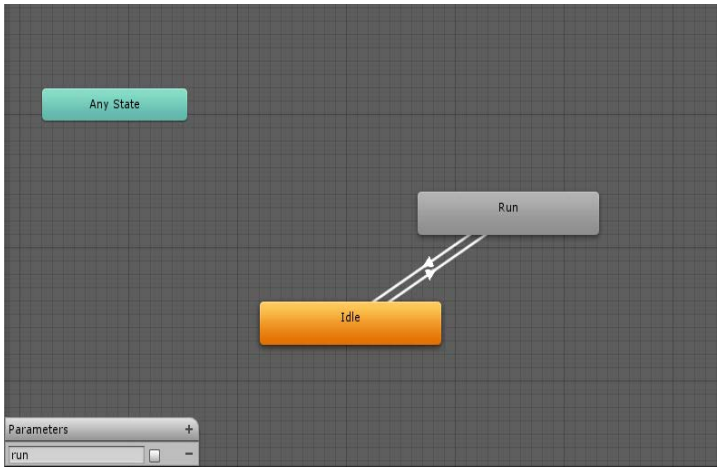
7. Now we can add the scripts, let's start with the player move script. You can find the scripts under Assets/3DForge/ Top-Down ARPG Camera & Controller/Scripts. We add the ARPGPlayerMove.cs to the Player (Parent) object. The Character Controller will be added automatically. Make sure the Character Controller is positioned correctly.

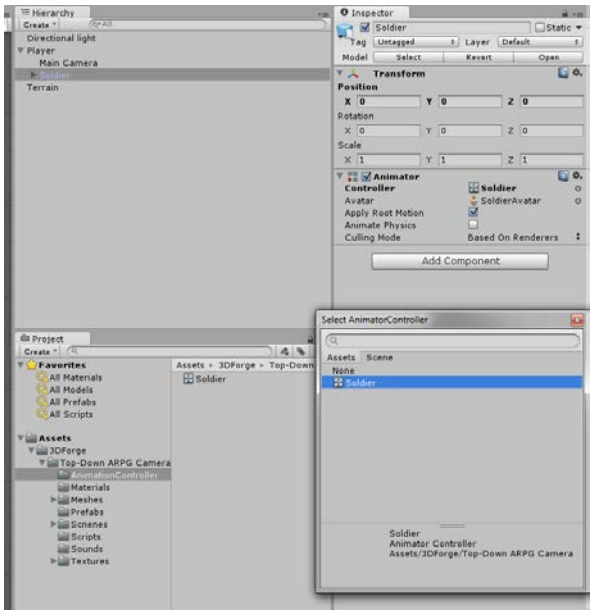


8. We then add the "ARPGCameraController.cs" (Object with script) to the Main Camera as we are building the Normal Top-Down camera. We can then set the target variable to the player game object.

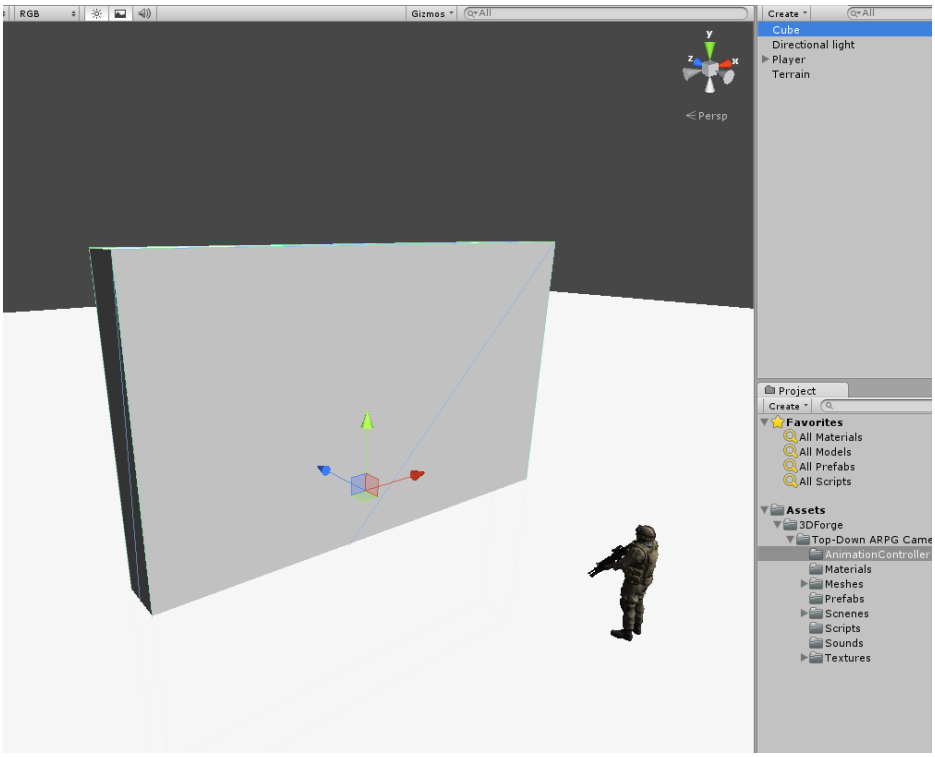


9. Player can move now but we need an AnimatorController for the character/toon to animate. The ARPGPlayerMove script only calls one animator variable “run” that is a Boolean. There is already one in the project under Assets/3DForge/ Top-Down ARPG Camera & Controller/AnimationController. Add AnimatorController to character/toon.

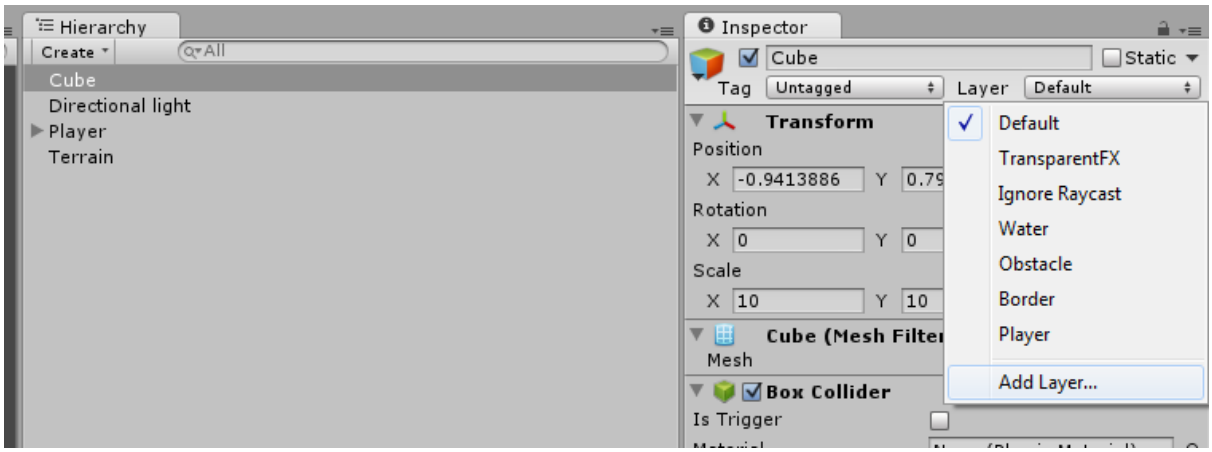


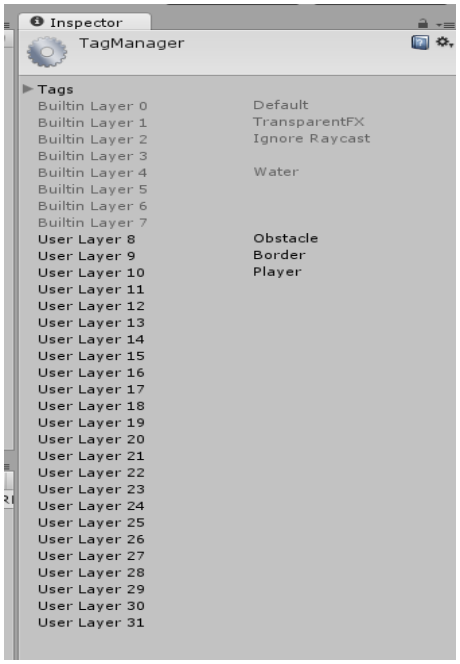


10. Now that we have the player running around looking awesome we can add some obstacles. Add a normal Cube to the scene and change its scale to (10, 10, 1).



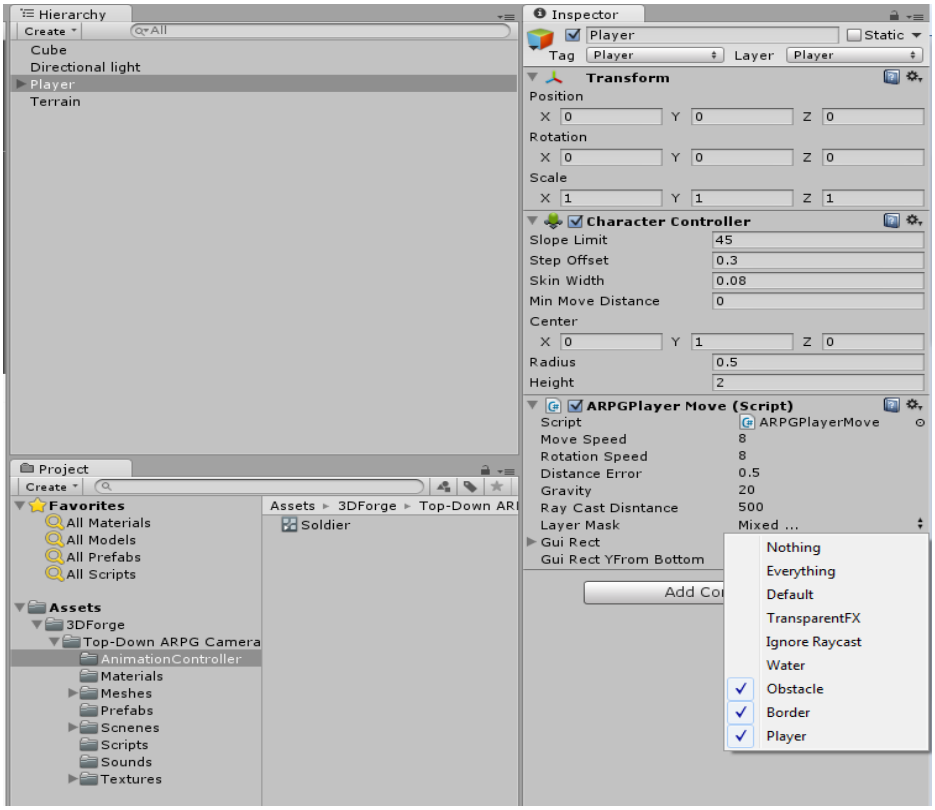
11. Add layers to project. I added Obstacle, Border and Player.



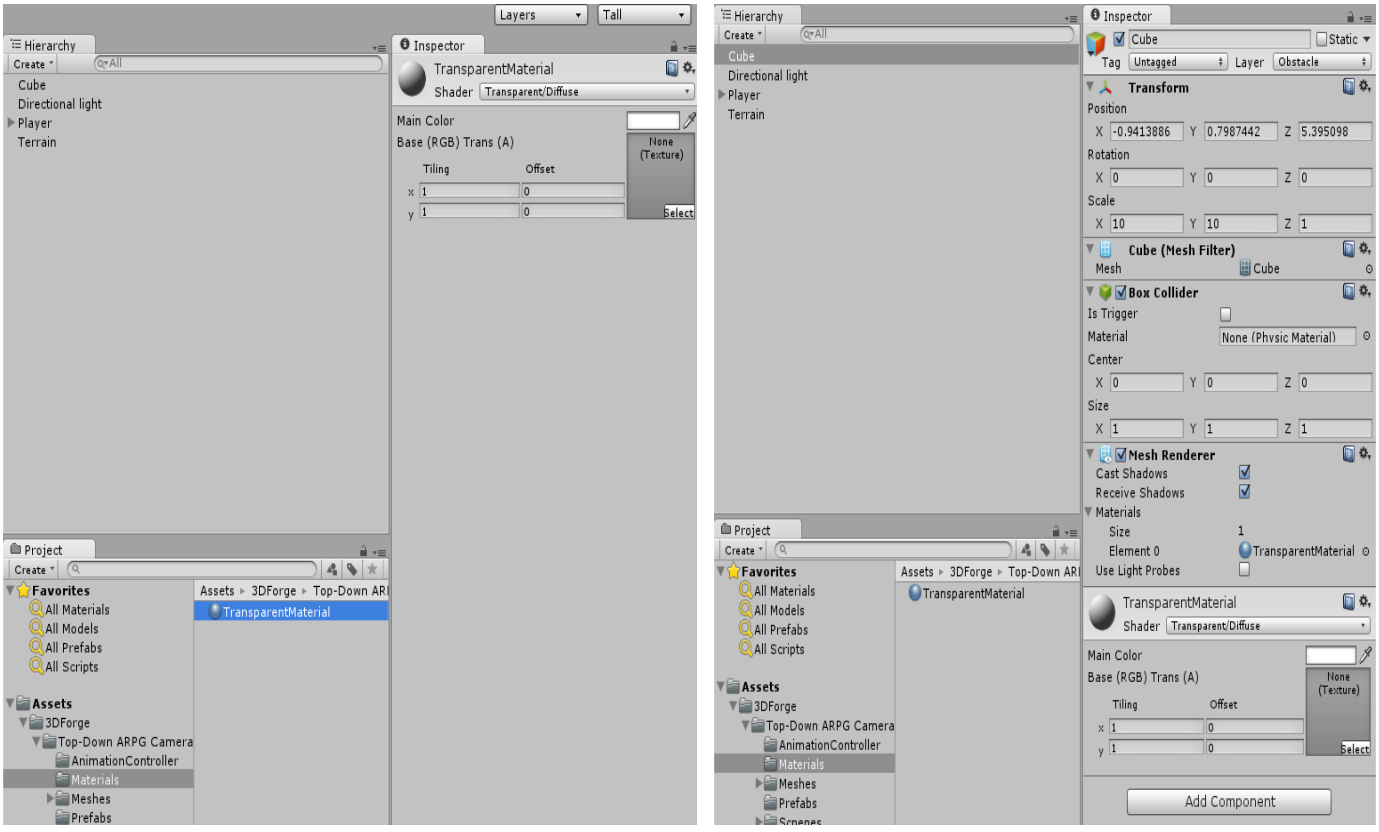


12. Now set the cube's layer to Obstacle.

13. On the player look at the ARPGPlayerMove script then on the LayerMask variable check the layers you wish the mouse clicks to ignore.

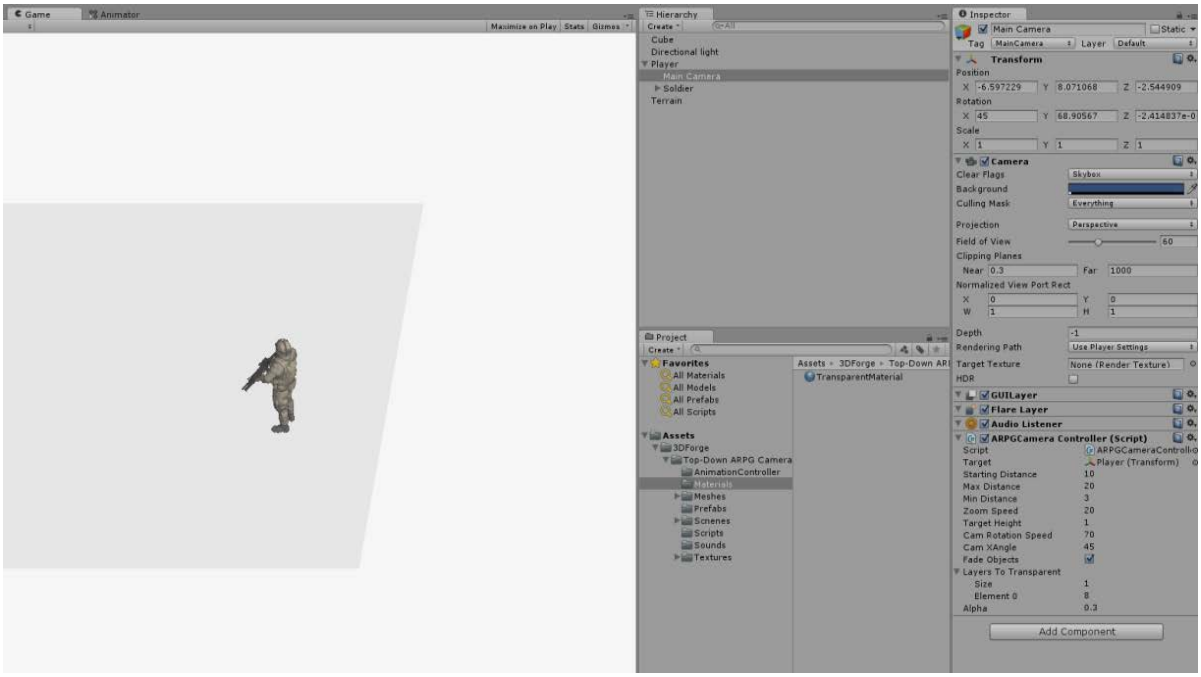


14. We can also set certain objects to go transparent using the layers we created, thirst we need to change the cubes materials to a transparent diffuse, create a transparent material then set the cubes material to the transparent one.

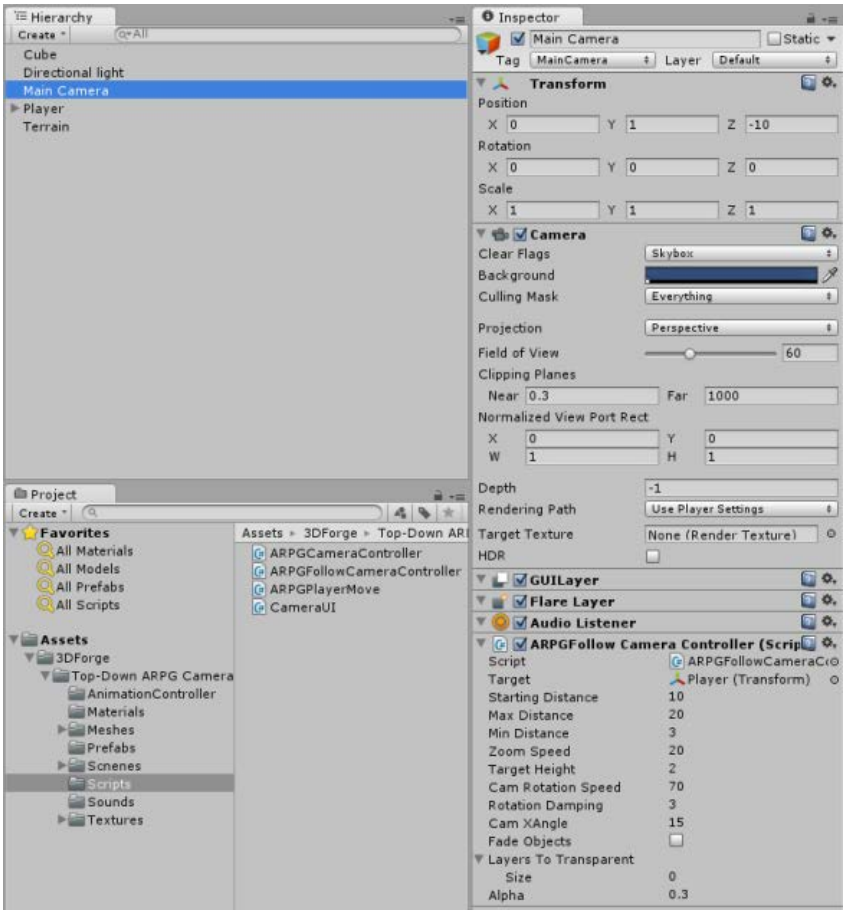


15. Now that the cube is setup we need to set the camera layers to test on, click on the Main Camera in the scene then on the script, check the Fade Objects variable.

    On the Layers to Transparent variable change the size to 1 and set Element 0 to 8 (Obstacle layer).



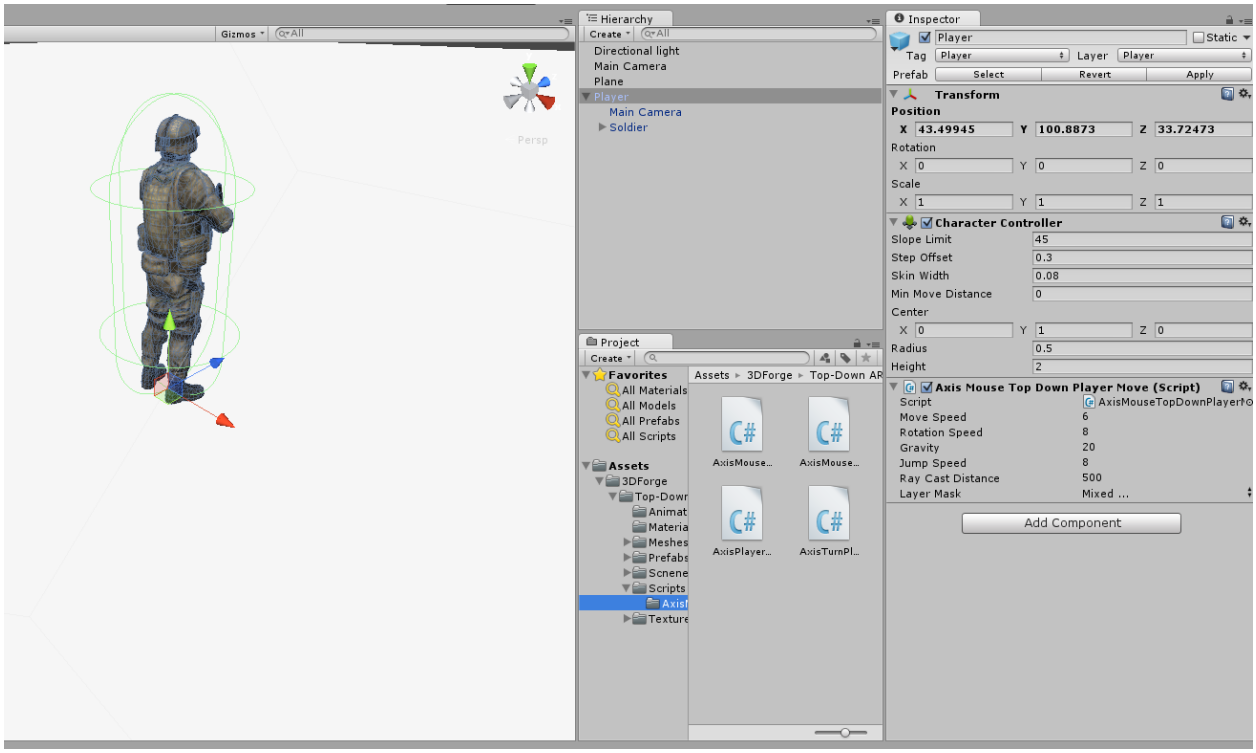
16. This setup was for the normal top-down ARPG camera, if you want to use the follow camera you just keep the Main Camera on the root.



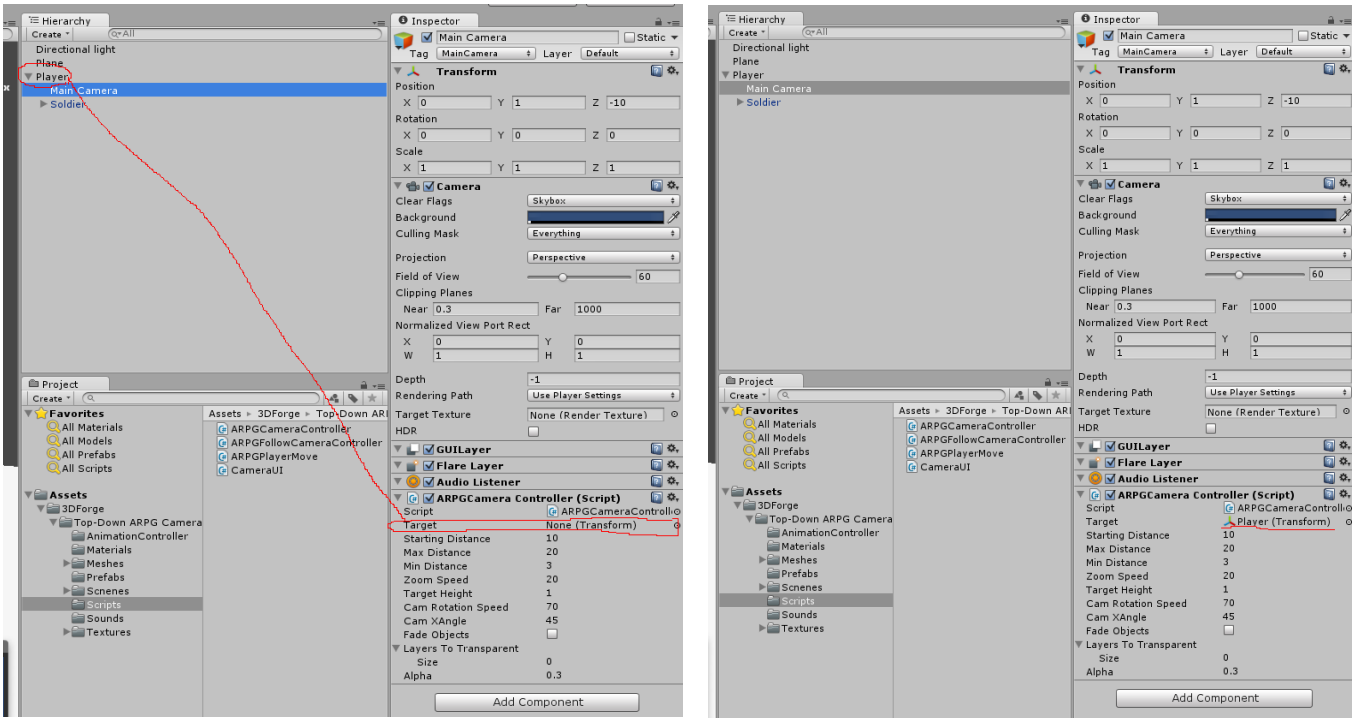
Setup 2

Setup guide for top-down key and mouse movement, this setup only shows you how to add “AxisMouseTopDownPlayerMove.cs” to the character.

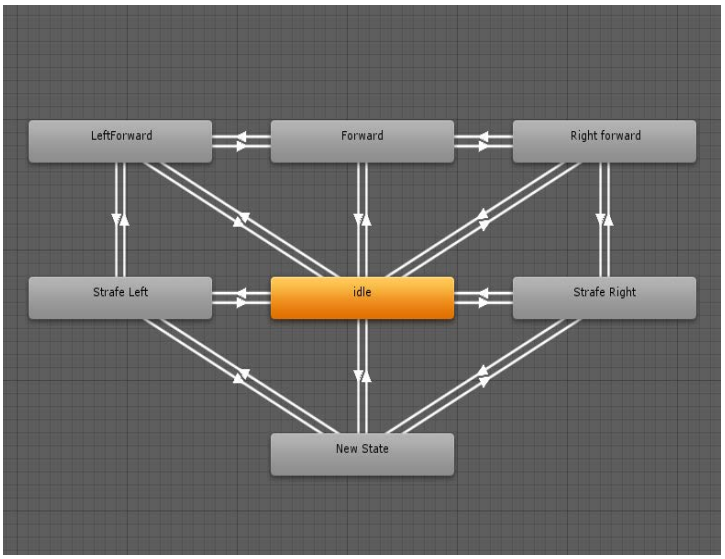
1. You can find the scripts under Assets/3DForge/ Top-Down ARPG Camera & Controller/Scripts/ AxisMoveUpdate. Add script to player object (see Setup1 1-6). The Character Controller will be added automatically. Make sure the Character Controller is positioned correctly.

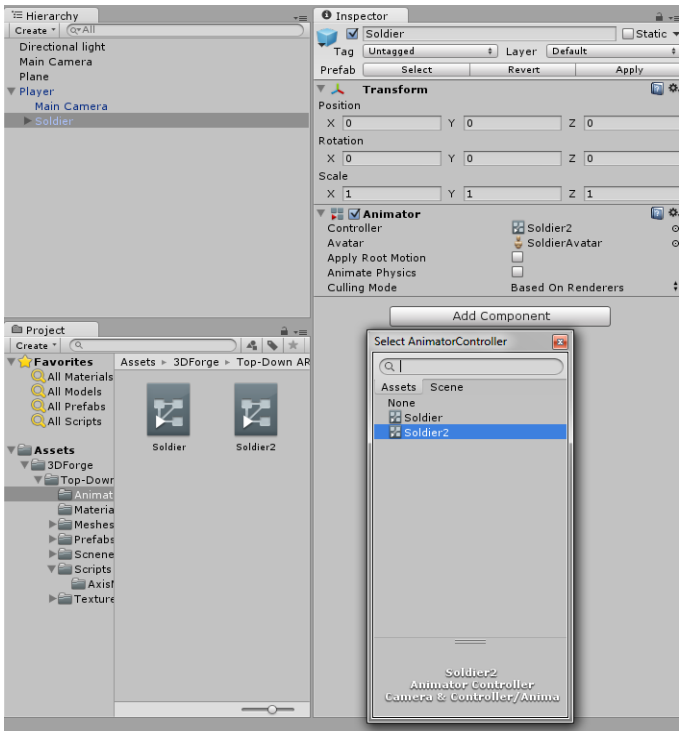


2. We then add the “AxisMouseTopDownPlayerMove.cs” (Object with script) to the Main Camera as we are building the Normal Top-Down camera. We can then set the target variable to the player game object.

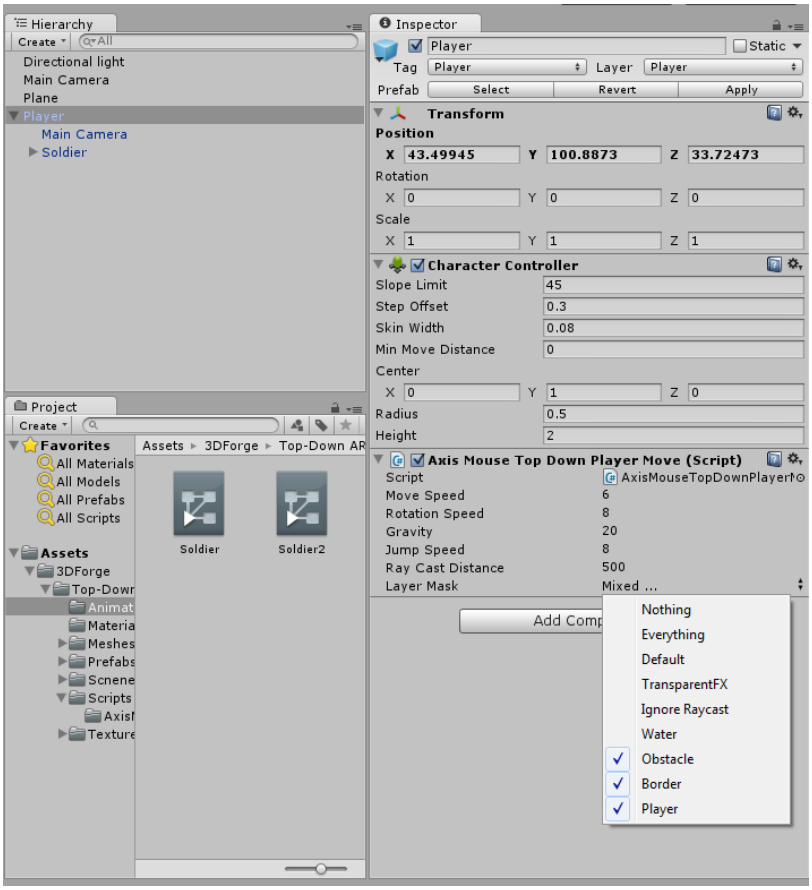


3. Player can move now but we need an AnimatorController for the character/toon to animate. The “AxisMouseTopDownPlayerMove .cs” script calls 2 animator variables “speed” and “strafe”, both are float values. These values are set in single method in the class, can easily be changed for your character and its animations. There is already one in the project under Assets/3DForge/ Top-Down ARPG Camera & Controller/AnimationController. Add AnimatorController to character/toon.





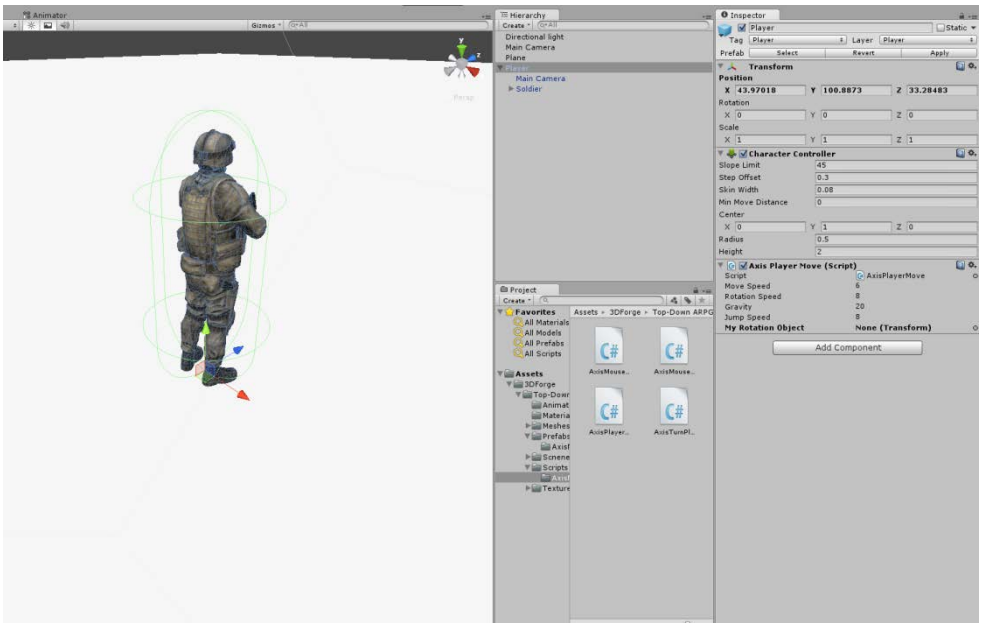
4. On the player look at the AxisMouseTopDownPlayerMove script then on the LayerMask variable check the layers you wish the mouse to ignore. Otherwise player will rotate to the colliders on that layer. Make sure you did Setup1 step 11.



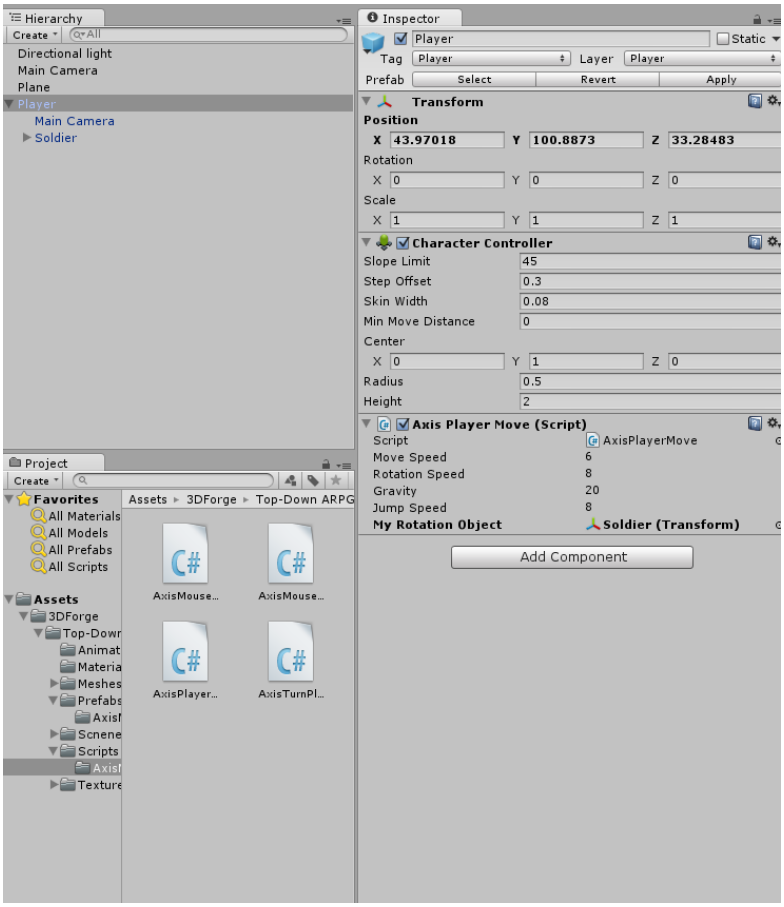
Setup 3

Setup guide for top-down key movement, this setup only shows you how to add “AxisPlayerMove.cs” to the character.

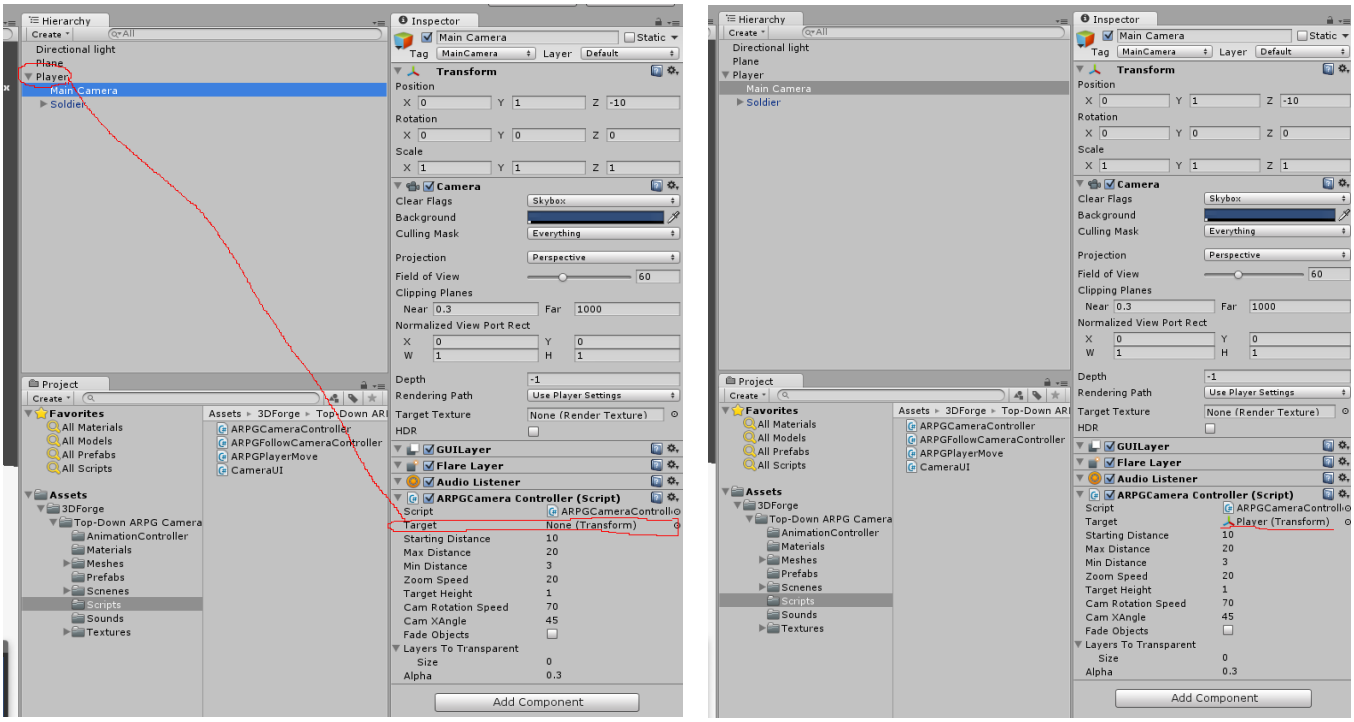
1. You can find the scripts under Assets/3DForge/ Top-Down ARPG Camera & Controller/Scripts/ AxisMoveUpdate. Add script to player object (see Setup1 1-6). The Character Controller will be added automatically. Make sure the Character Controller is positioned correctly.



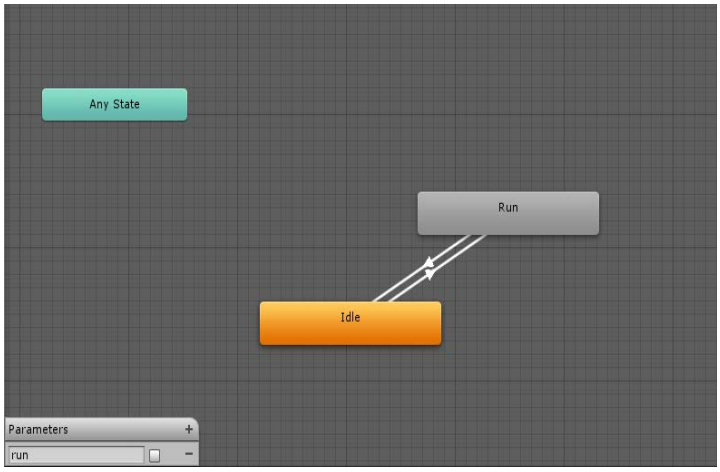
2. We need te set the object that will rotate, in this case the soldier object. Drag the Soldier object to My Rotation Object.

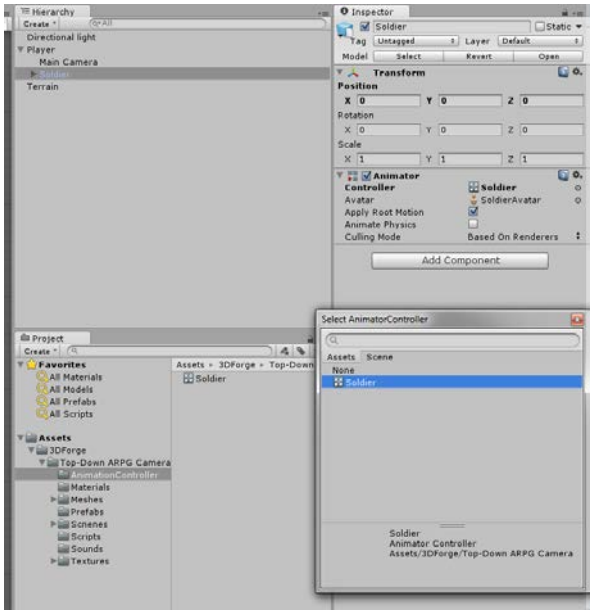


3. We then add the “AxisPlayerMove.cs” (Object with script) to the Main Camera as we are building the Normal Top-Down camera. We can then set the target variable to the player game object.



4. Player can move now but we need an AnimatorController for the character/toon to animate. The AxisPlayerMove script only calls one animator variable “run” that is a Boolean. There is already one in the project under Assets/3DForge/ Top-Down ARPG Camera & Controller/AnimationController. Add AnimatorController to character/toon.





Setup 4

For “AxisMousePlayerMove.cs” and “AxisTurnPlayerMove.cs”, we just need to do Setup 2 set 1-2.

There is no extra setup for these two.