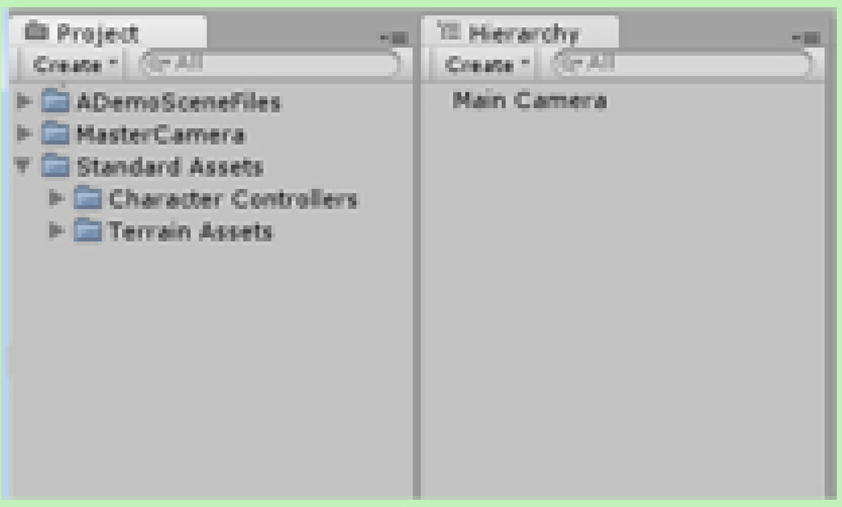
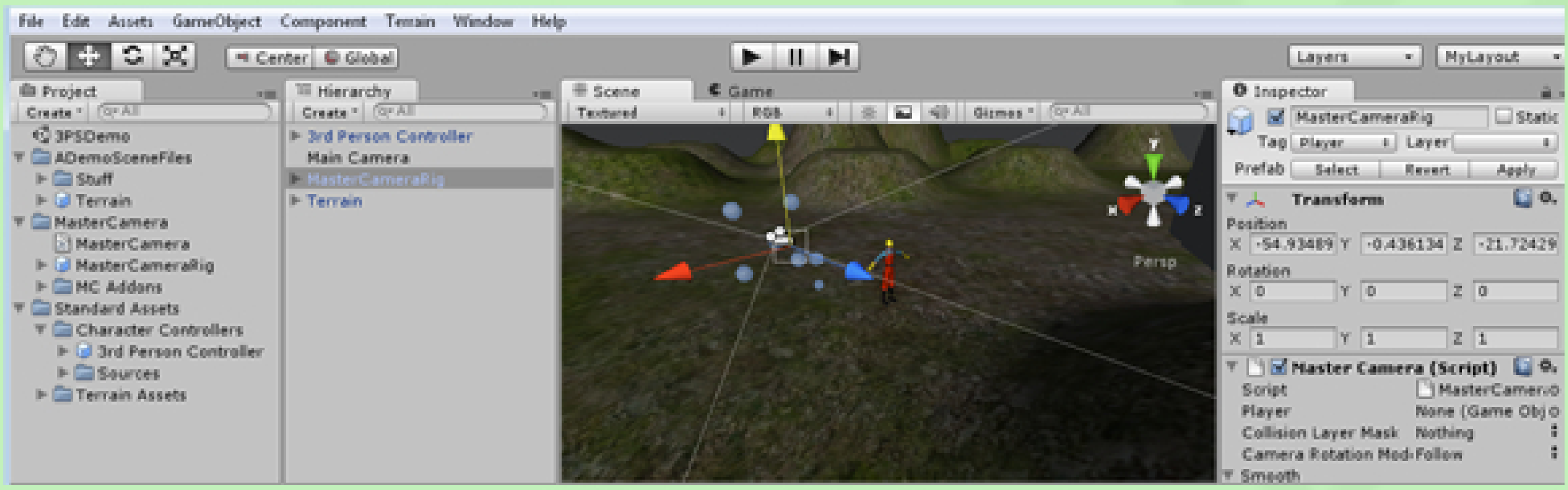
Master Camera

version 2.2

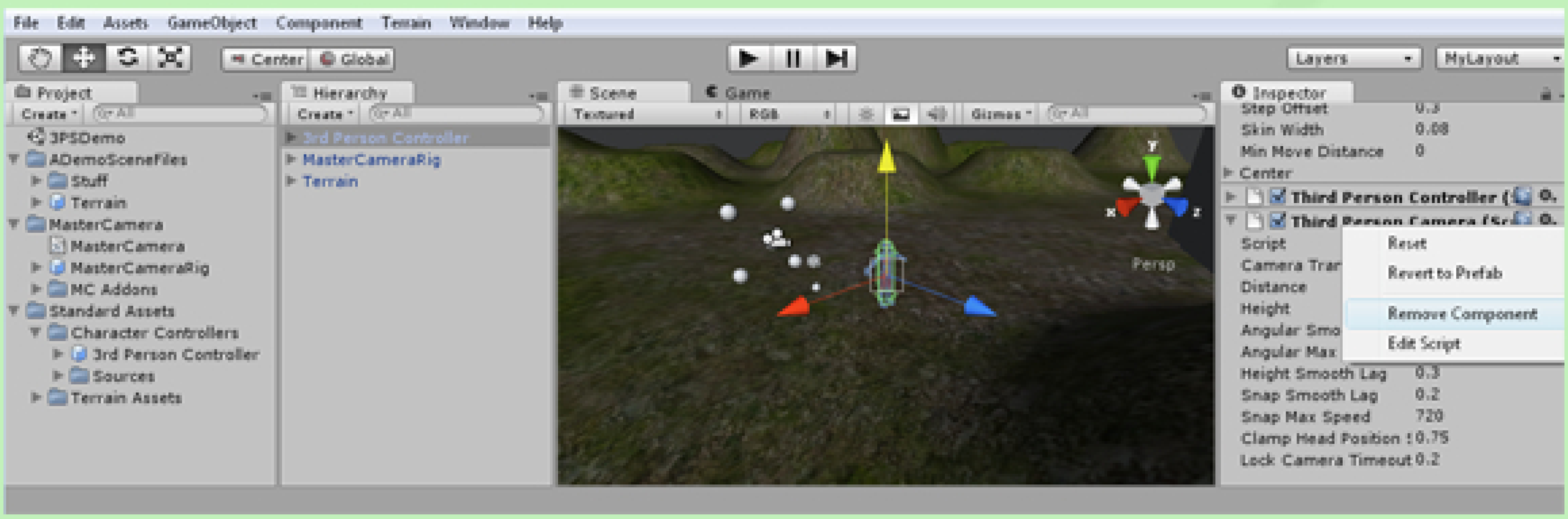
Initial Setup



For the purposes of this tutorial I have created a new project with the *Character Controllers* and *Terrain Assets* packages in the *Standard Assets* folder. We also have the *Master Camera* package and *ADemoSceneFiles* which contains a terrain object so we have something for the camera to interact with.



-Place the *Terrain, MasterCameraRig,* and *3rd Person Controller* prefabs into the scene.

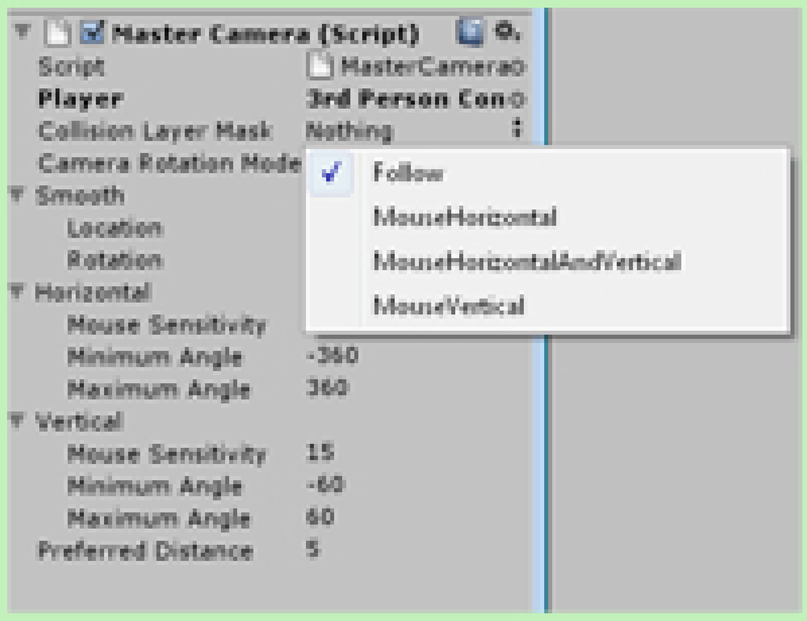


-Delete the *Main Camera* Object from the Heirarchy.
-Remove the *Third Person Camera (Script)* from the *3rd Person Controller* prefab.

Add the *3rd Person Controller* to the "Player" GameObject variable in the *Master Camera (Script)*.

We now have a basic camera that follows the Character,
But Wait! There's more.

Core Functions



Collision Layer Mask: If MCScroll script is added, the camera rig will collide and not clip through layers selected here.

Camera Rotation Mode:

Follow: The camera simply follows the character around.

MouseHorizontal: The camera responds to input from the mouse on the horizontal axis.

MouseHorizontalAndVertical: The camera responds to input on the mouse horizontal and vertical axis.

MouseVertical: Camera responds to mouse input on the vertical axis.

None: Camera does not rotate.

Smooth:

Location: How quickly the *MasterCameraRig* updates to the location of the Player.

Rotation: How quickly the *MasterCameraRig* rotates around the player in "Follow" mode.

Horizontal:

Mouse Sensitivity: How quickly the rig rotates on the horizontal axis via mouse input.

Minimum Angle: The minimum local euler angle on the Y axis of the rig's rotation object.

Maximum Angle: The maximum local euler angle on the Y axis of the rig's rotation object.

(Note: At Min - 360/ Max 360 the camera will spin infinitely in either direction on the horizontal axis.)

Vertical:

Mouse Sensitivity: How quickly the rig rotates on the vertical axis via mouse input.

Minimum Angle: The minimum local euler angle on the X axis of the rig's rotation object.

Maximum Angle: The maximum local euler angle on the X axis of the rig's rotation object.

(Note: Outside min of -90 and max 90, the camera will spin out of control as it looses the relative down)

Preferred Distance: How far away the camera is from the player.

Callable Functions

TestLOS (layerMask):

Tests if there is line of sight from the camera to the player, ignoring anything on the layerMask.

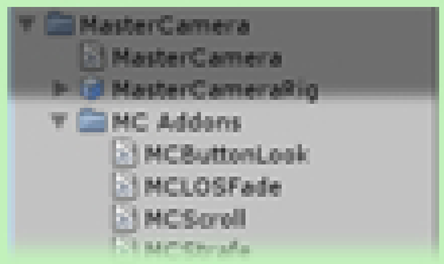
GetLocalRotationZ():

Returns the local z rotation (tilt) of the camera.

SetLocalRotationZ(float):

Sets the local z rotation (tilt) of the camera. This is useable for creating camera shake effects.

Addons



Addons are scripts that add additional functionality to the core *MasterCameraRig*. To use, simply drag and drop them onto the *MasterCameraRig* object in your Heirarchy.

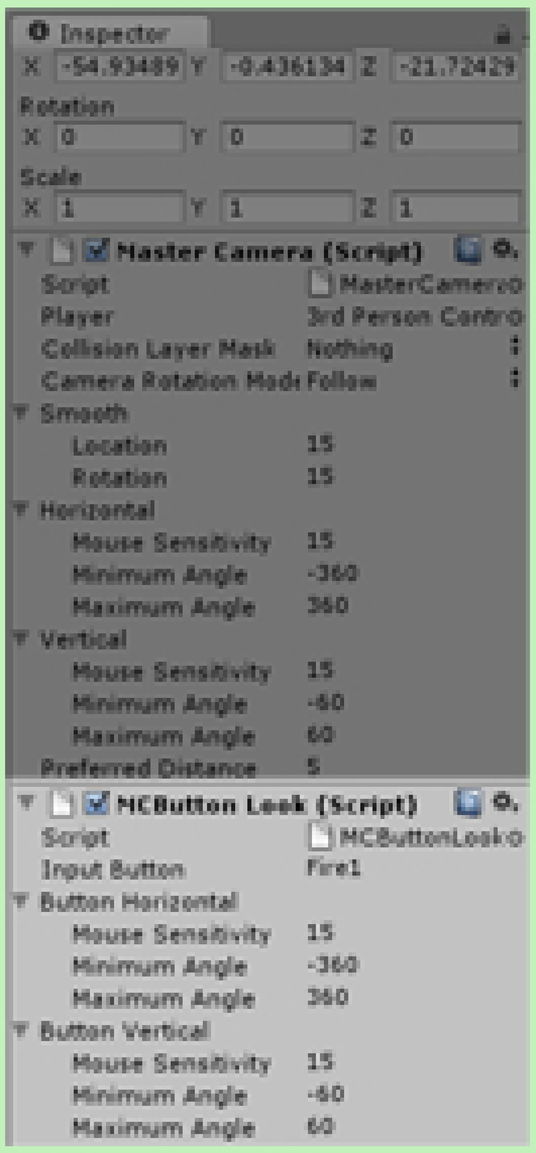
MCButtonLook: This allows you to designate an input via the Input Manager (setup by default to the “Fire1” or Left Mouse Click) which triggers mouse look functionality. Generally used on in “Follow” mode to let the player look around their surroundings.

MCLOSFade: Line of Sight Fade. This scripts causes object on designated layers to become partially transparent when they come between the camera and the player.

MCScroll: The camera is able to scroll in and out using the mouse wheel. The camera also collides with objects on designated layers and zooms in to maintain sight of the player.

MCStrafe: The camera can strafe back and fourth over the player’s shoulder, via keyboard inputs.

MCButtonLook



MCButtonLook: This allows you to designate an input via the Input Manager (setup by default to the "Fire1" or Left Mouse Click) which triggers mouse look functionality. Generally used on in "Follow" mode to let the player look around their surroundings.

Input Button: The input that needs to be held down to enable mouse look functionality

Button Horizontal:

Mouse Sensitivity: How quickly the rig rotates on the horizontal axis via mouse input.

Minimum Angle: The minimum local euler angle on the Y axis of the rig's rotation object.

Maximum Angle: The maximum local euler angle on the Y axis of the rig's rotation object.

(Note: At Min - 360/ Max 360 the camera will spin infinitely in either direction on the horizontal axis. Standard)

Button Vertical:

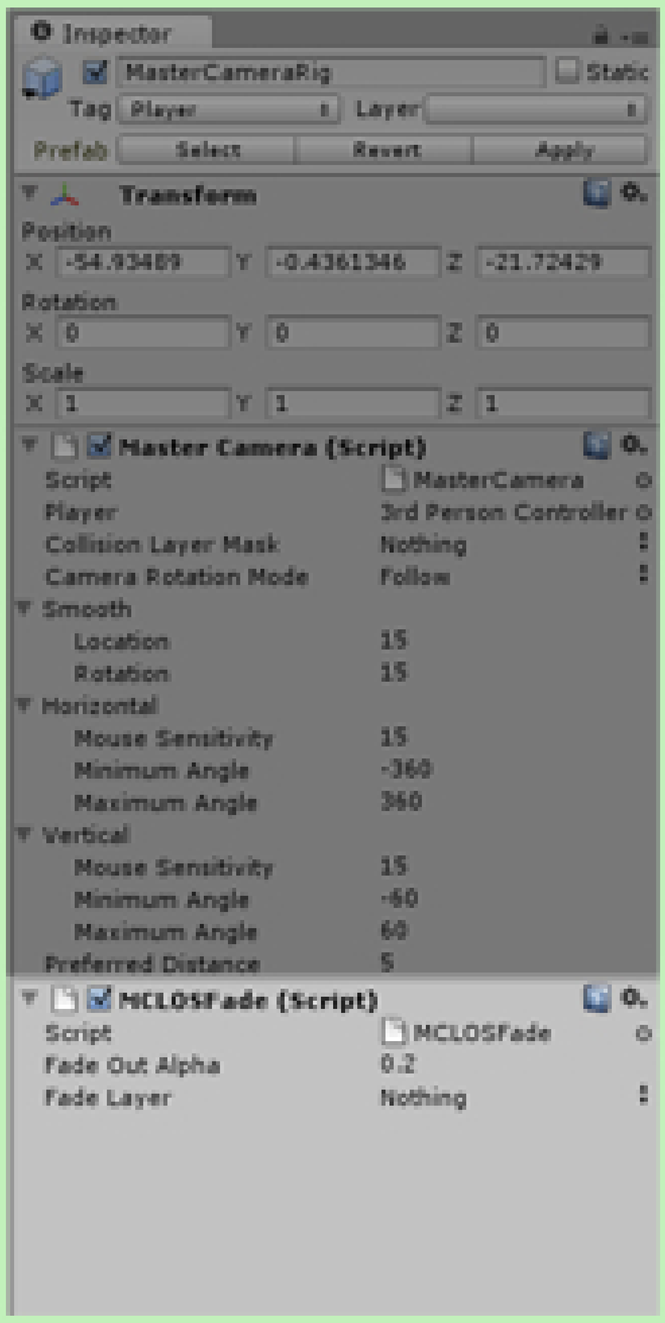
Mouse Sensitivity: How quickly the rig rotates on the vertical axis via mouse input.

Minimum Angle: The minimum local euler angle on the X axis of the rig's rotation object.

Maximum Angle: The maximum local euler angle on the X axis of the rig's rotation object.

(Note: Outside min of -90 and max 90, the camera will spin out of control as it looses the relative down)

MCLOSFade

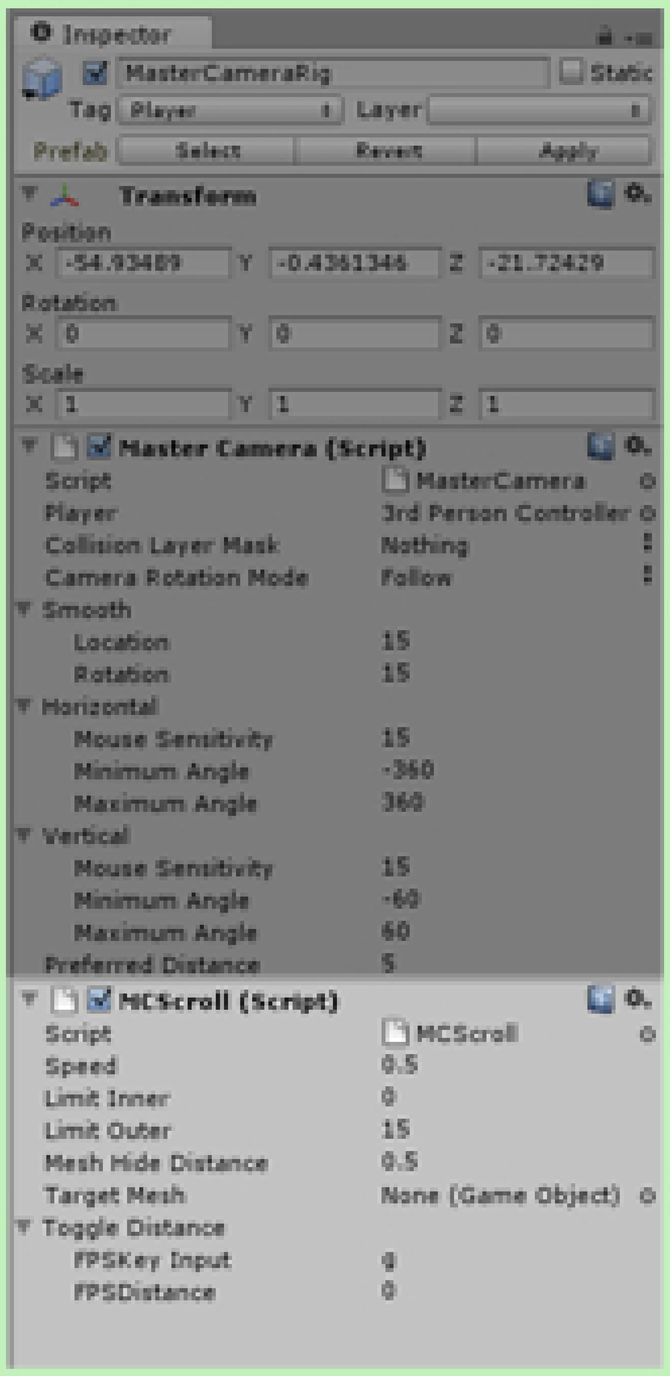


Fade Out Alpha: Alpha transparency on a scale of 0 to 1.

Fade Layer: Layer(s) of the objects to fade out.

(Note: If your camera uses scrolling and collision, the Fade Layers need to be removed from the Collision Layer Mask.)

MCScroll



Speed: How fast the camera scrolls in and out via mouse wheel.

Limit Inner: How close the camera can get to the player.

Limit Outer: How far away the camera can get from the character.

Limit Inner: How close the camera can get to the player.

Mesh Hide Distance: How close the camera can get to the player mesh before the player mesh gameObject is disabled.

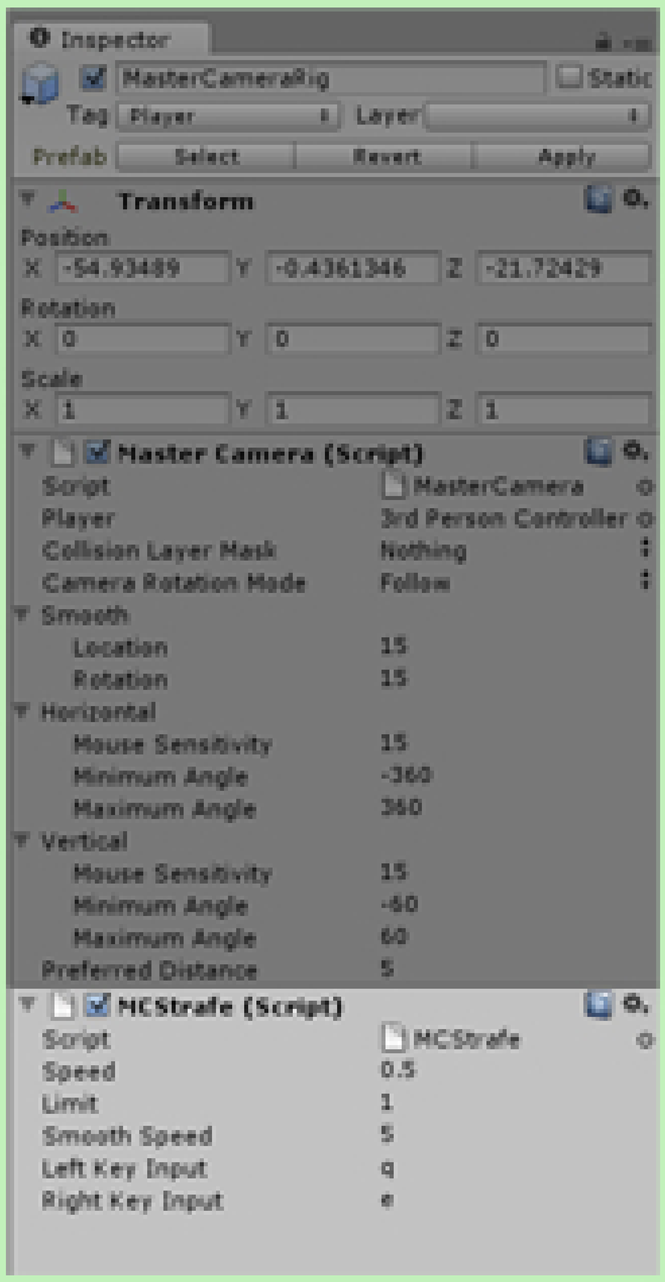
Target Mesh: Player mesh to hide based on Mesh Hide Distance.

Toggle Distance: Generally for 3PS to FPS toggling.

FPSKey Input: The keyboard key that toggles the action.

FPS Distance: The distance toggled to, not necessarily toggling to FPS, maybe a close 3PS or any distance.

MCStrafe



Speed: Speed that the camera moves over the shoulder.

Limit: Maximum distance the camera can mover from the center.

Smooth Speed: How quickly the camera catches up to the designated position over the character's shoulder.

Left Key Input: Keyboard key to move the camera toward the left shoulder.

Right Key Input: Keyboard key to move the camera toward the right shoulder.

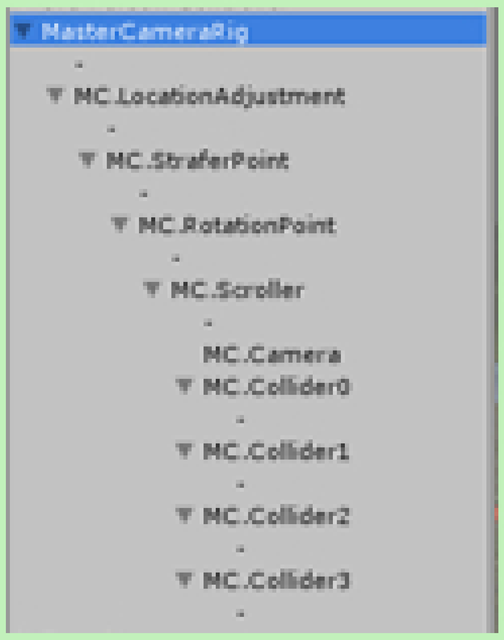
(Note: If the Left and Right Key Inputs are the same, the camera will start out over the right shoulder at the Limit distance, and toggle between the Left and Right limits when the Input key is used.)

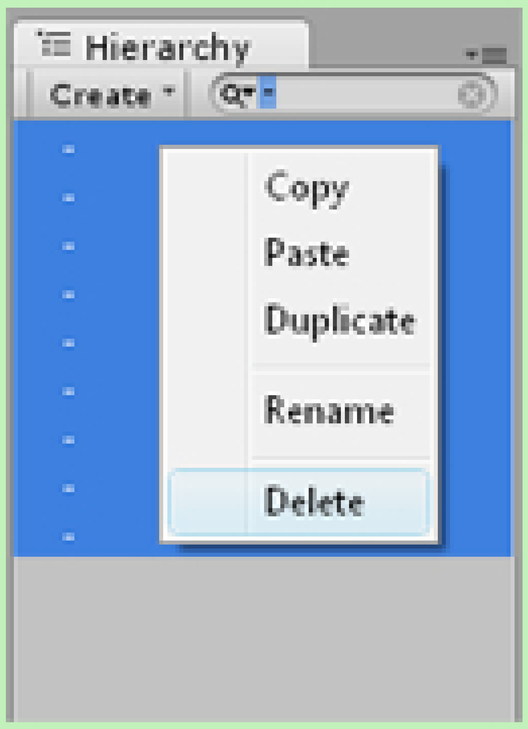
Callable Functions

StrafeSwap();

Toggles the location of the camera from it's current distance over one shoulder to the same distance over the other shoulder.

Master Camera Rig





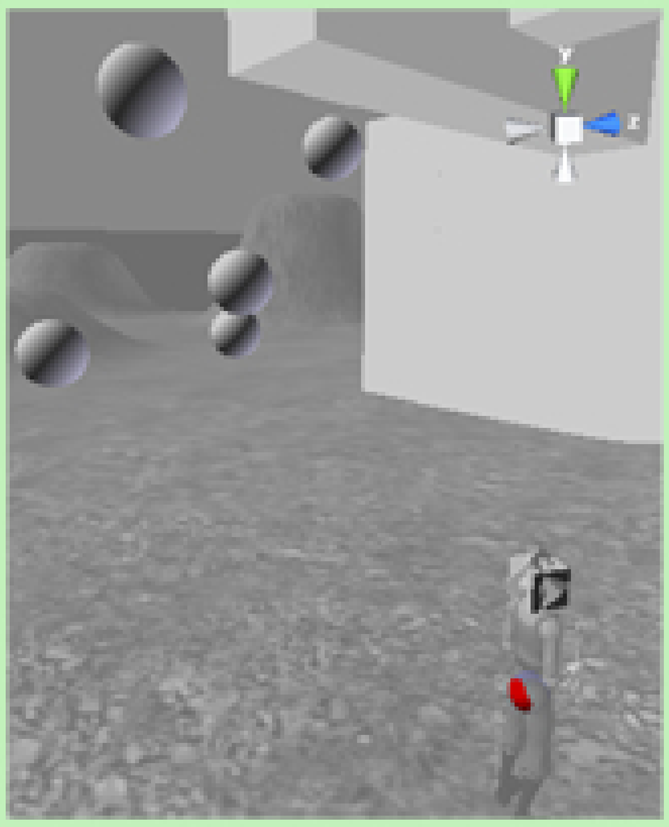
The Master Camera is a collections of game objects that allows the camera to function. Each game object's behavior is driven by the MasterCamera and Addon scripts.

Visualizations

At each level, under each game object, there is an additional game object named "-". These objects are for visual reference. They are just objects with a cube or sphere mesh renderer. They are not referenced or controlled by any script. They are all at the origin point of their parent object.

When all of settings are correctly adjusted for the camera rig, these visualization objects named"-" may easily be searched for and deleted.

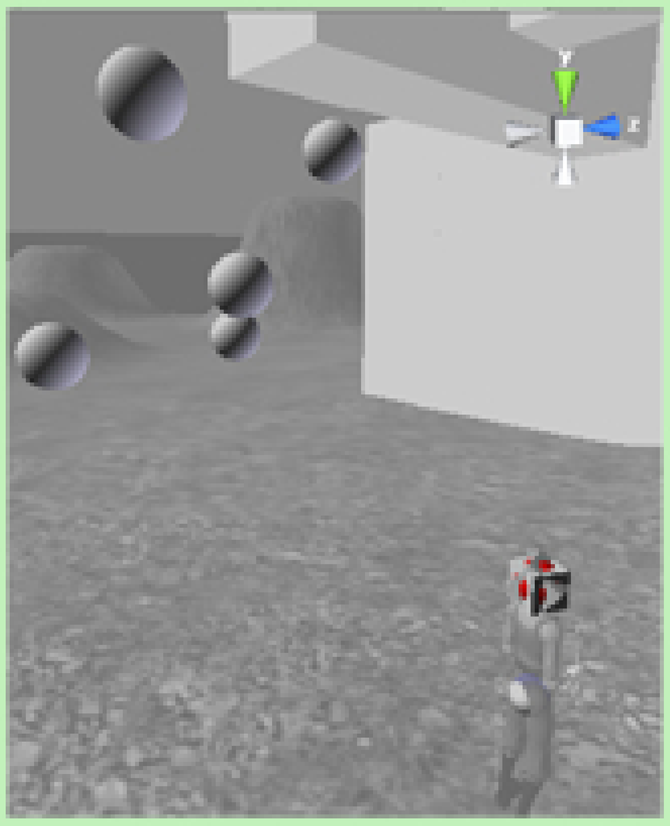
The individual mesh renderers, or the gameObjects themselves can be disabled to hide them from view, but keep them around for future testing.



Master Camera Origin

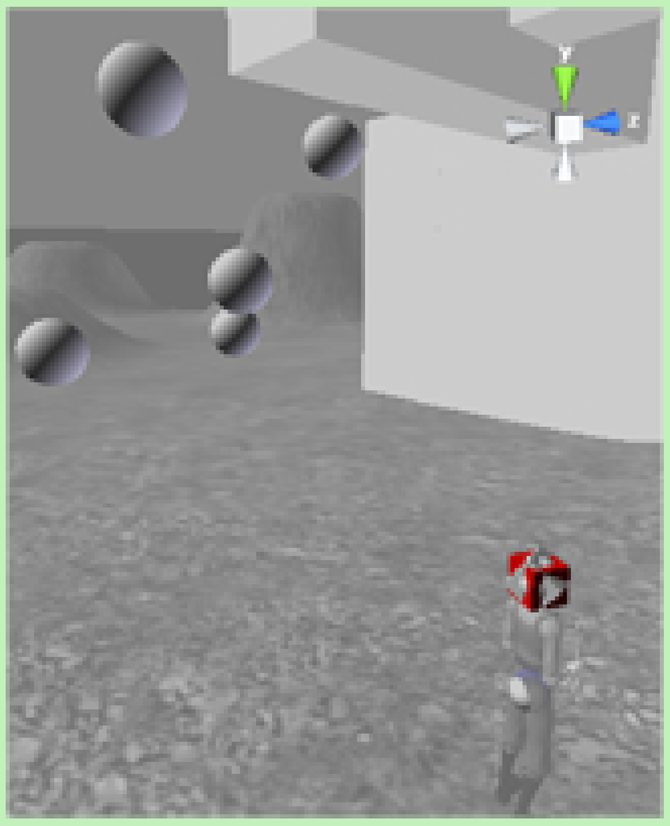
The origin point of the whole rig. This is the object that positions itself at the origin point of the *Player* variable and carries the rest of the rig with it.

Master Camera Rig
(continued 1)



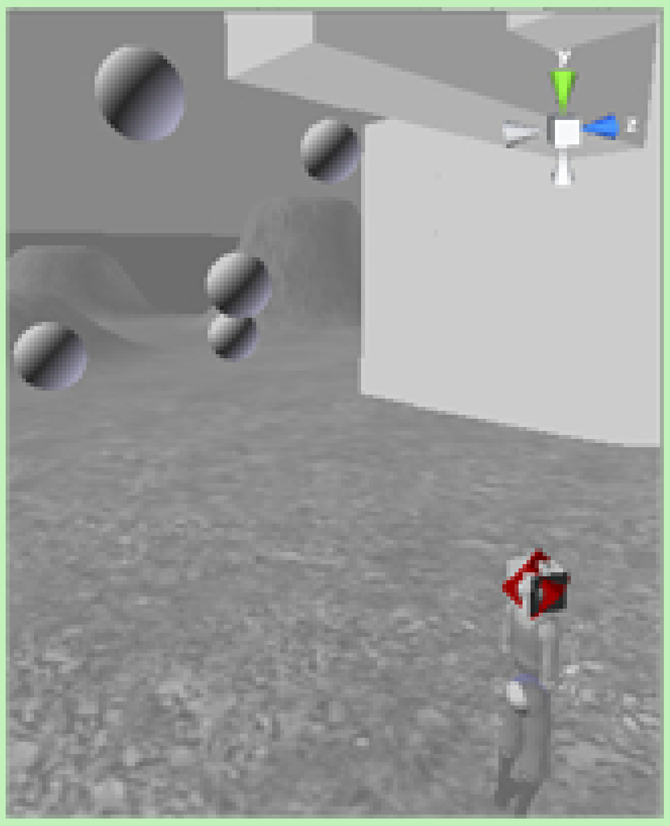
Location Adjustment
(sphere)

As the Master Camera origin goes to the origin of the *Player* object, an adjustment usually must be made to allow the camera to operate at or around the character model's head. The *Location Adjustment* object can be moved to wherever the camera is desired to operate around (e.g. head, shoulders, waist, above the head, etc.)



Strafer Point
(cube)

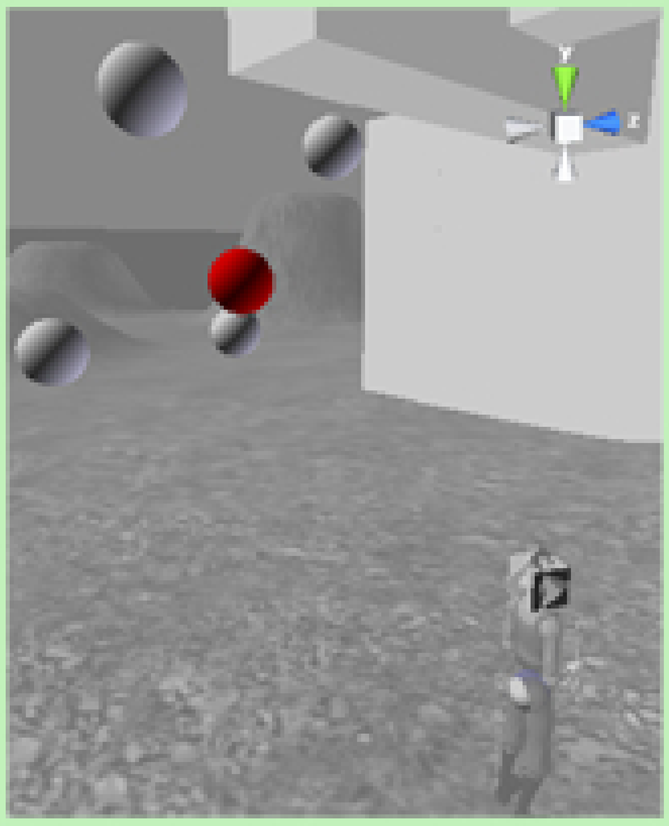
This is the object that moves back and forth over the character's shoulders when the *MCStrafe* script is added. It always faces the *Player*'s forward direction.



Rotation Point
(cube)

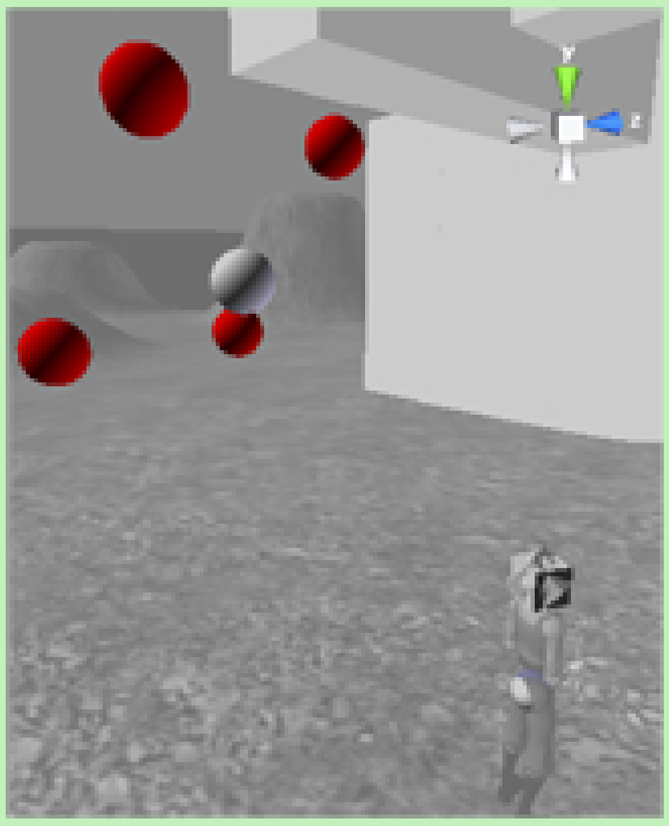
The object that rotates around the Strafer Point. It always faces away from the camera.

Master Camera Rig
(continued 2)



Scroller
(sphere)

This is the object that the camera is directly attached to. It is the object that scrolls in and out when the *MCScroll* script is attached.



Colliders
(spheres)

This are the objects that rays are cast to for testing whether or not the Scroller should scroll in to maintain view of the Player, when the *MCScroll* script is attached.