
# Third Person Cover Shooter

## Video tutorials

Character setup : https://youtu.be/8aCr4eIMjFc
Add Pistol : https://youtu.be/CIes-mAj8q8
Add Rifle: https://youtu.be/HwN4rB-jPvo

## Covers

Walls usable for taking covers have to be marked by cover markers. A cover marker is any game object with Cover and Box Collider components attached. Markers can intersect, form a chain and act as if that chain is one big cover.

Cover orientation matters, the example scene contains markers with feet that mark facing directions.

There are two kinds of covers, low and tall. The kind is determined from the height of a BoxCollider attached to the marker. The height threshold is different for every character and is

defined by a Character Motor. In a chain of covers of different height character will correctly change its stance when transitioning between tall and low covers.

A corner of a cover with no adjacent covers nearby is treated as a corner characters can peek from. However, there can be unmarked walls and therefore character can attempt to take a peek in impossible situations. Such cases are handled by Open Left and Open Right properties inside the Cover component, setting a value to false marks that corner as unusable for peeking.

Low covers can be climbed or vaulted over. Every cover has the type of climbing defined in a Cover component. Vaulting should be enabled for low walls players can jump over and climbing for covers alongside higher ground.

# Character Motor

Characters must have a Character Motor component attached. It manages the character, it's movement, appearance and use of weapons. It handles gravity and therefore gravity should be turned off in the RigidBody component to avoid conflicts.

There is an IK (inverse-kinematics) system that handles aiming and recoil. It can be configured manually but for ease of use there is a button to set it up automatically inside the CharacterMotor inspector. It is recommended to reduce the amount of bones used by IK on non-player characters for performance reasons.

IK is calculated by adjusting bones until certain objects reach defined targets. Target objects must be part of the skeleton in order for changes to modify their transform. Character's sight is usually defined by a marker object that is part of the head, bones are transformed until marker's forward vector points towards the target.

Each character has a set of weapons in its disposal. To add a new weapon to the character you must create an object with a 3D model and a Gun component and attach it to a hand. Additionally, you can create a version of the weapon that is put into its holster. The motor automatically enables and disables weapon and holster objects.

# Gun

Guns raycasts bullet, manage clip and recoil.

For player characters bullets originate at camera in order for player to be able to fire on targets they can see, even if there is a small obstacle in front of the gun. The fire origin is set by a camera. Since AI do not have Third Person Camera attached their bullets are fired starting from the Aim marker, which usually is at the end of the gun.

Each weapon has two marker objects. Aim defines point of origin for AI bullets and is also used when rotating character's arms till the marker points towards the target. Left Hand object marks the position for character's left hand. Naming of left and right hands is incidental and character's handedness can be swapped.

The intended position of the left hand might differ in some animations, to handle that there are left hand marker overwrites you can use to set up IK for the left hand for some specific situations. Empty values are not used as overwrites.

Currently there are two kinds of weapons, pistols and rifles. The type defines character animations when using a weapon.

# Player Controller

Takes player keyboard and mouse input and translates that to Character Motor commands.

Weapons are controlled by the number keys. Pressing 1 hides a weapon and keys starting with 2 make the character equip a weapon. The order of keys and weapons defined in the Character Motor is the same.

# AI Controller

Makes the character walk and fire towards an enemy. Stays idle until provoked. AI tries to maintain a certain distance from its threat. The AI can use cover.

It reacts to the appearance of player in its line of sight. The field of view and distance of it can be controlled. Additionally, it notices characters within a certain distance in any direction. However, it ignores characters if they are sneaking behind the AI. The AI will notice friends getting hurt using its line of sight. If the friend is behind the character it will notice it if it is closer than Friend Hurt Distance.

In order to make gameplay fairer the AI has a small delay which can be controlled by adjusting the Reaction Time. It’s amount of time in seconds it takes for AI to notice threat or adjust its aim when the target is moving.

# Third Person Camera

Manages the camera object by setting an appropriate orientation depending on the target object's state. For camera to work you have to link it to the target object that has a Character Motor attached.

The camera component also maintains and draws a crosshair. It hides the crosshair if the character is unarmed or unable to fire at a wall because is too close. The visibility of crosshair also can be turned off manually by setting Is Crosshair Enabled value to false when your game needs so.

# Character Face

Reacts to character events and modifies blend shapes in a mesh. The set of faces can be expanded by modifying the script or making a new version of it.

# Character Platform

Keeps character on top of a moving platform. It doesn't require a Character Motor or any other component and therefore can be used on any object even if it's not physical.

# Additional Components

These components are added for example purposes. Users of the system might want to have their own implementations that fit their game better.

## Character Health

Manages health and sets Is Alive in Character Motor to false when it reaches 0. Registers damage done by bullets.

## Character Sounds

An example sound system that spawns sound instances upon various character events. Sounds are randomly picked from lists.

## Gun Effects

Similar to Character Sounds, Gun Effects spawns effects and sounds on various gun events. The effects are prefab objects the component uses to instantiate.

## Hit Effect

Spawns an effect upon a bullet hit. The effects are prefab objects the component uses to instantiate.

# Support

Ask questions, send ideas, report bugs: eduardas.ninja@gmail.com

Thanks,

RedBee Team