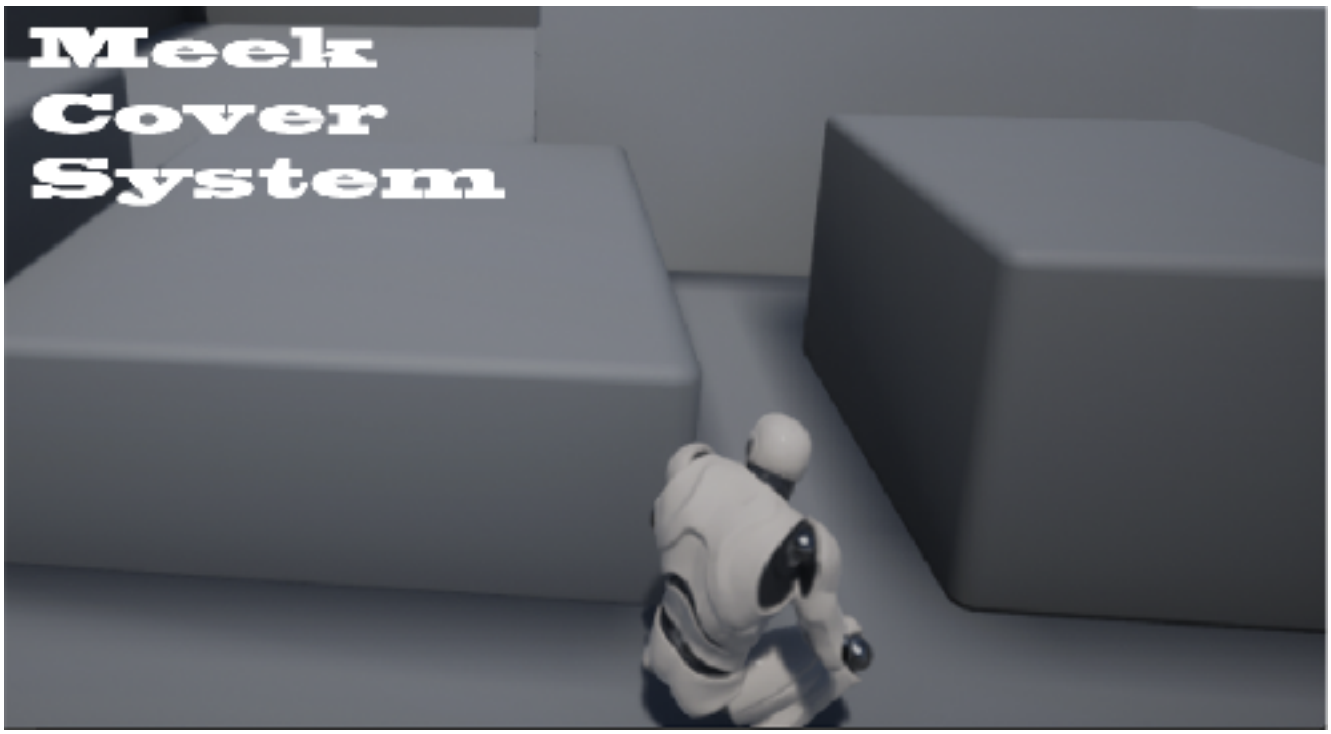
Meek Cover System

Version 1.00 for UE 4.18

Meeksoft LLC - December 21, 2017

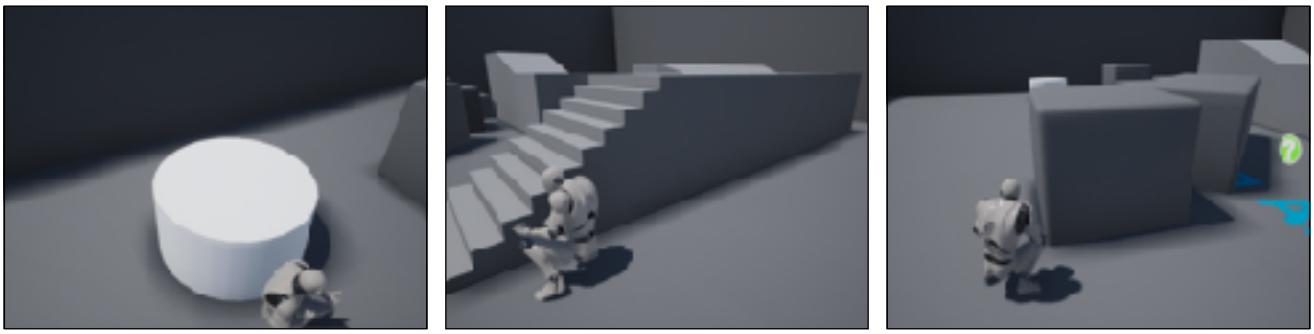


Introduction

Aloha! And Thank you for your interest in our Meek Cover System (MCS.) This is built from the **Third Person** Project setup. The entire purpose of MCS is to provide example and logic for your own cover system. We wanted a multiplayer cover system that didn't require special "walls." Unable to find suitable tutorials, examples, and Marketplace blue prints, we decide to implement our own. As time continued, we added other features, such as cover to cover. We hope you find this project very useful and instructive.

MCS is based on animations from the free UE4 Animation Starter Pack. Hence, our cover animations is the standard moving animation **BUT turned 90 degrees**, to face parallel with the wall. We then limit movement to following the wall.

If you have custom animations, you will may need to modify the 90 degree aspect. As of right now, we plan to, but do not provide a tutorial on that.



Cover against any surface normal

Quick list of features:

- Multiplayer support. Built and tested with the "Run as Server" mode.
- Hard Cover system. You need to press a button.
- Cover against any surface. No special "wall" materials required.
- Move from cover to cover.
- Move along adjoined walls.
- Move along curved walls.
- Move along concave/hard angled walls
- Very basic cover fire.

Built in Logic:

- Camera switch when in cover, vs when out of cover. You may/may not like this.
- Moving back, or away from wall, auto uncovers you.
- Auto crouch if wall is short. Based on wall height. Adjustable by a variable.
- Cover movement is based on camera. So forward while looking left, will move left.
- Cover to cover requires you going to edge of wall, then pressing cover.
- Curved walls has a minimum size. Else our line trace won't detect tiny walls.
- Cover fire was built as an example/filler for your real/animated cover fire.

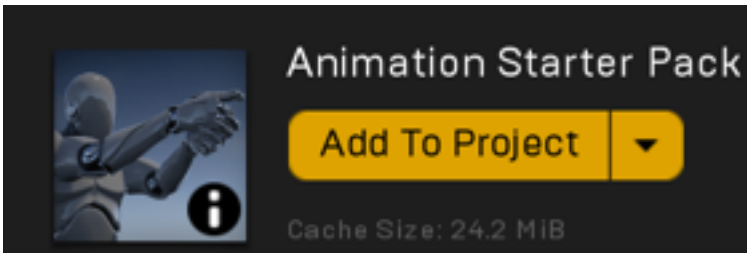
Playing

Move close to wall. Press "C" to enter cover. Move along the wall, being aware of where your camera is facing. Moving away from the wall, or pressing "C" will have you uncover. While at the edge of a wall, you can either Fire or move cover to cover. Press "C" to attempt cover to cover. Press and hold"LMB" to cover fire.

Files and Structure

There is only 1 file required, with all logic and functionality built into it; the ***MeekCharacter*** file. The ***MeekCharacter_BP*** Blueprint is an exact copy and rename of the ***UE4ASP_HeroTPP_AnimBlueprint*** Blueprint that is located in the **AnimStarterPack** folder.

MeekCharacter located in the Meek Folder



The **AnimStarterPack** is free and available through the UE4 Marketplace; as **Animation Starter Pack**.

New Actions

We added a total of only 2 new actions. **Cover** and **Fire**. **Cover** initiates the cover functionality, beginning with looking and placing the model into cover. The **Fire** action, is a placeholder, and basic logic, for your own **Fire** System. It includes firing from cover; when the model is on the end (aka **edge**) of the cover.

Event Graph Events

MCS is based on Third Person Project template that included the Animation Starter Pack; and you will notice changes in 2 places. The **Movement Input**, and **Crouching**. The **Movement Input** is explained in the function section. The **Crouching** was modified to allow multiplayer replication. Cover to cover is covered more in **Cover Movement** section.

We have added:

Cover: Entry point, when cover key is pressed.

Cover Set Direction: Used for multiplayer replication.

Cover Set: Multiplayer replication. Used to save prior to cover variables, and sets them back. Additionally we move the camera.

Cover Set Server: More multiplayer replication.

Cover Camera: Moves the camera when in cover. Either "more" to left or right.

IgnoreMovement: Movement Helpers, ignore move input. Used during cover to cover.

ResetMovementInput: Movement Helpers, resets input. Used during cover to cover.

Cover Move To Start: Cover movement using a timeline.

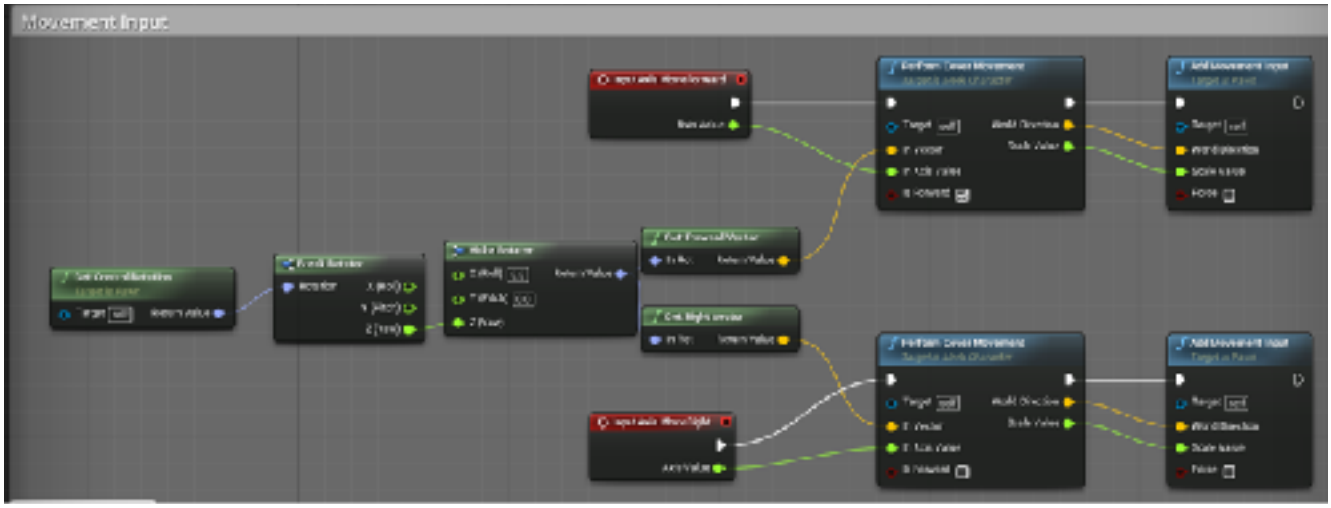
Cover Move To Stop: Cover movement.

Cover Move To Server: Multiplayer replication.

Cover Move To Multicast: Multiplayer replication.

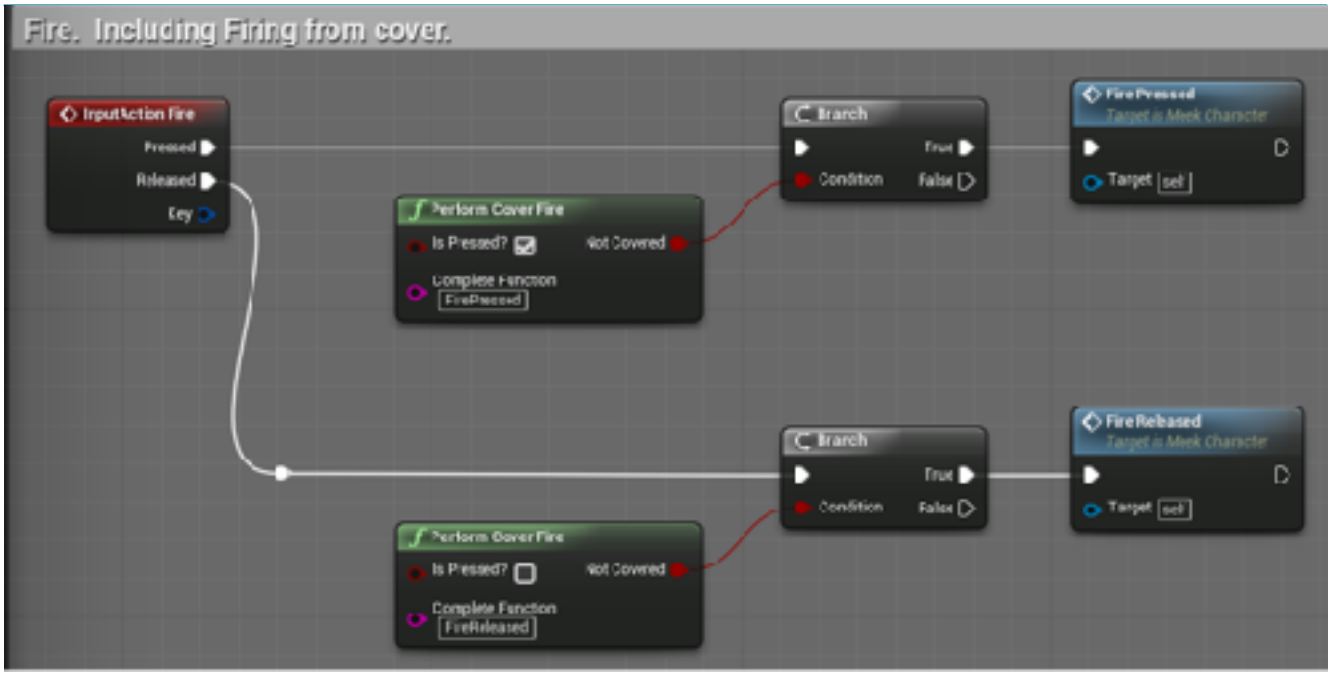
Functions

We have many cover functions. There are 3 functions that are used on the characters's Event Graph: **PerformCoverMovement**, **PerformCover**, and **PerformCoverFire**. The others are traces and checks The most important check is **CanMoveInCover**. It is used in **PerformCoverMovement**.



PerformCoverMovement "locks" the character's movement parallel to the wall.

PerformCover checks/moves the character into cover. And does cover-to-cover.



PerformCoverFire checks/performs cover fire. See Cover Fire Section.

CanMoveInCover

The real meat and potatoes of the entire MCS. Here we determine several checks:

1. Determine movement direction, based on camera location.
2. Determine direction to face. We use **CoverDirectionToFace** function.
3. Determine if upcoming cover is perpendicular. Using **StandardForwardTrace**.
4. Determine if there is more cover along the faced direction. Using **CoverTrace**.
5. Determine if we reached the edge of cover.
6. Determine if we need to crouch, because of low cover. See variable **HighCoverHeightPercent** in variables section.
7. Allow movement parallel (90 degrees) to wall.

Variables

There are 3 cover variables that you may set for each blueprint. These are **DistanceToCover**, **HighCoverHeightPercent**, and **CoverToCoverDistance**.

DistanceToCover is the distance a character must be, to enter into cover. And also the distance the character must maintain, to stay in cover. This is defaulted to ***90***, a random number I picked.

HighCoverHeightPercent is the percent, the cover must be under, to be determined as low cover. This is defaulted to ***0.7***, aka 70%. So any cover that is lower than 70% of the character, is determined to be low cover.

CoverToCoverDistance is the distance the next cover must be, from the character, in order for the character to move into the next cover. This is defaulted to ***150***, a random number I picked.

There majority of variables that are mainly for server replication and multiplayer support. A few for saving the character's status prior to cover; because we want to switch back to them when we uncover. Such as **SavedUCRYaw**, **SavedORTM**, **SavedCDR**, **ForcedCrouch?**, etc. A few variables are for cover fire, and discussed in the **Cover Fire Setup**.

Cover Move variables are used to track movement into and out of cover. See **Cover Movement** section.

Cover Movement

This section discusses how Entering into Cover and moving Cover to Cover is implemented. MCS uses a ***Timeline node*** to move and rotate the character from one location to another.

We use the 4 custom events: **CoverMoveToStart**, **CoverMoveToStop**, **CoverMoveToServer**, **CoverMoveToMulticast**. The last two are for multiplayer replication. **CoverMoveToStart**, begins the movement throughout the timeline. **CoverMoveToStop** interrupts, and stops the timeline. In both functions, we set and store variables.

We decided to use a timeline for several reasons. One reason is for multiplayer replication. For smoother transition. So characters don't teleport. And to check boundaries. We do a last "sweep" check when movement is finished.

Additionally, in **CoverMoveToStart**, you can pass function name (as string,) that you want to execute after the movement ends. MCS uses this call when cover firing. For example, we want to first move into cover fire position, and then start firing.

Cover Fire Setup

This is a place holder, a guide, and not intended to be a full implementation.

When cover firing, we rotate and move the character to "pop out" from cover. We store these rotations and locations in variables: **CoverActorLocation, CoverActorRotation, CoverMoveToStartLocation, CoverMoveToStartRotation**.

On the **Event Graph**, you may add your firing function/logic calls after the **FirePressed**/**FireReleased** custom events. You may also add your firing function/logic calls during the **FireServer** custom event; replacing the **Print Strings**.