[Year]

MODULAR WEAPON COMPONENT SYSTEM TOOL (MWCST) – USER GUIDE

VERSION 1.0

OWEN SIMS

CONTENTS

Requirements....1
Installation....1
Getting Started....2
- Setting Up Mesh Assets....2
- Creating a Modular Weapon Asset....3
  - Building the Weapon Tree....3
  - Affecting Weapon Function Using Property Deltas....5
- Creating a Weapon Blueprint....5

Blueprint Nodes....6
- Physical Actions....6
  - Fire....7
  - Reload....7
- Component Modification....7
  - Get All Option Pins....7
  - Set Selected Option Component....7
- Miscellaneous....7
  - Set Skin....7
  - Reset Skin....8
- Utility....8
  - Convert Fire Rate to Cooldown Time....8
- Getters....8

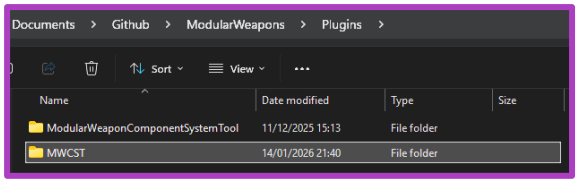
Additional Information....8

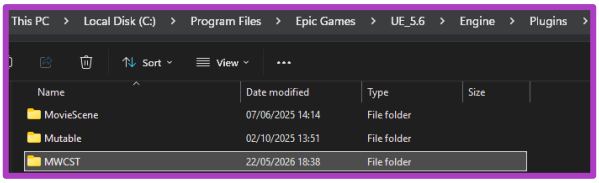
REQUIREMENTS

- Unreal Engine 5.6+
- MWCST Plugin
- Any modular weapon meshes

INSTALLATION

Extract the .ZIP archive provided to the plugins folder either in the project files or to your engine plugins folder. Enable the plugin in your project and restart the editor.



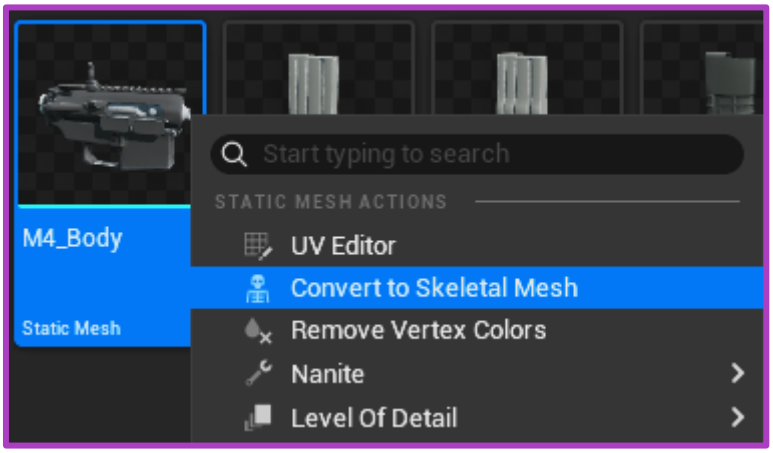


GETTING STARTED

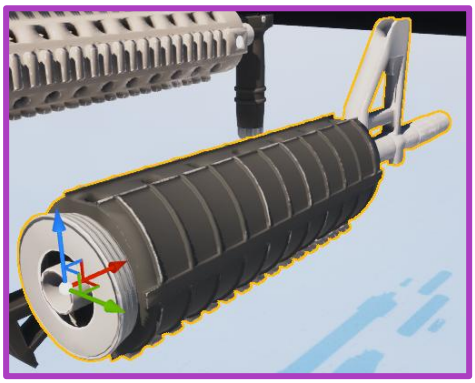
SETTING UP MESH ASSETS

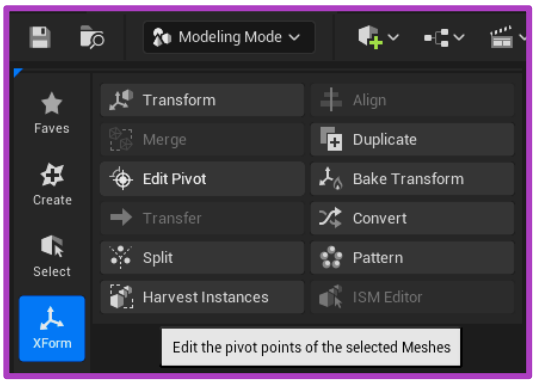
For your modular weapons to be built and to look as intended, it is important to ensure that the meshes are set up ready for usage. The requirements for this are:

1. As of the current version of this plugin, the root weapon component must have a skeletal mesh, and all other attachments must have static meshes. If your root weapon mesh is a static mesh, simply convert the asset to a skeletal mesh for the purposes of usage with this plugin. This can be done via the right-click context menu:

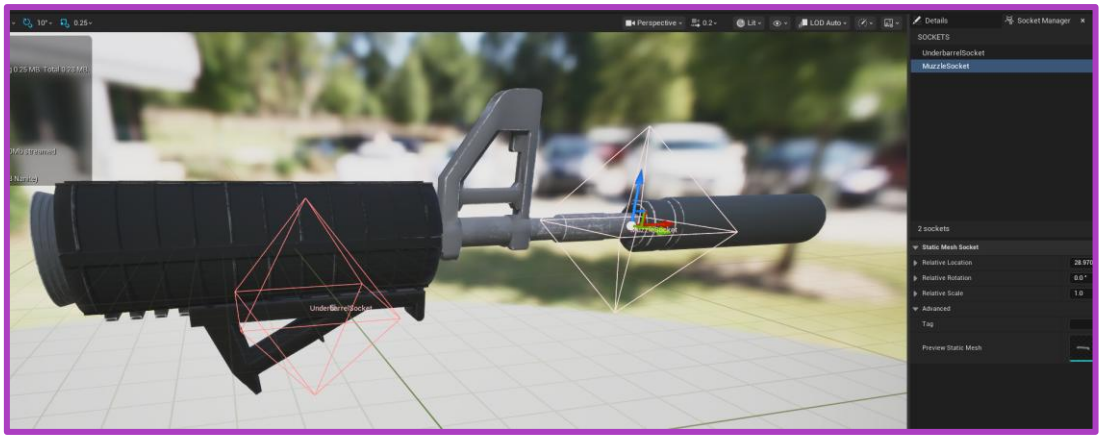


2. Ensure that the pivots for each weapon attachment are set to the point at which the component will attach to its parent. This will allow components to connect properly and can be performed on assets in engine via the modelling mode in the level editor:





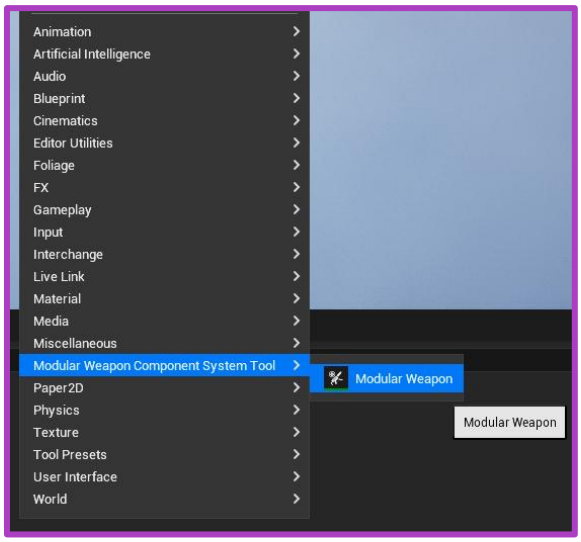
3. Add sockets to component meshes wherever attachments are possible. This applies for the weapon root, within which skeletal mesh sockets will be used, and for every other component, within which static mesh sockets will be used. You can test the visuals of the attachments here too by setting preview meshes.



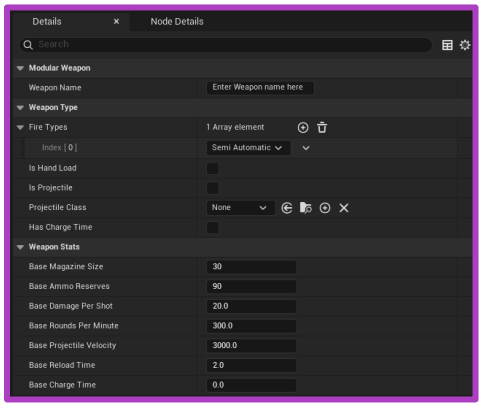
CREATING A MODULAR WEAPON ASSET

BUILDING THE WEAPON TREE

After setting up your assets, navigate to the content browser and right-click to bring up the context menu. Under the category ‘Modular Weapon Component System Tool’, click the ‘Modular Weapon’ asset type.

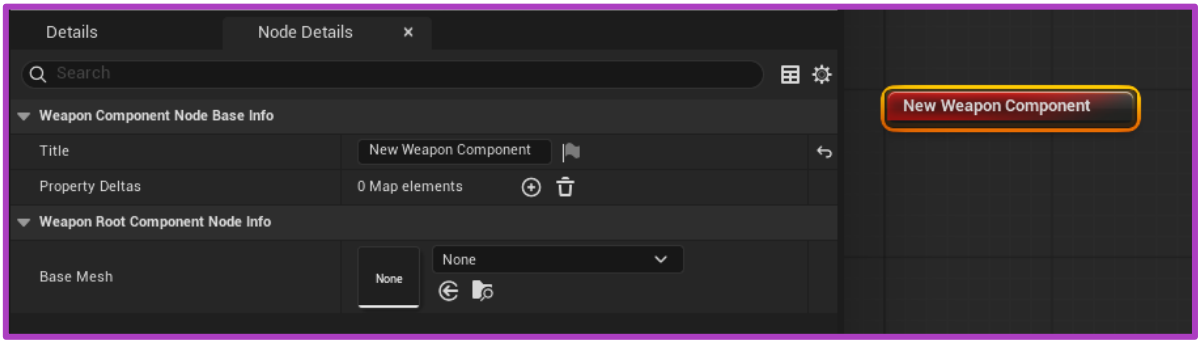


Open the asset. This will open the custom editor toolkit for your modular weapon asset. If the ‘Details’ panel has not opened by default, open it via the ‘Window’ menu.

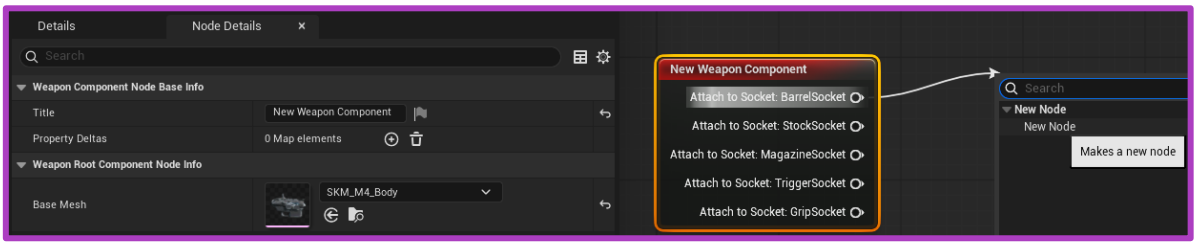


This defines the core aspects of the weapon which will be used by the in-built functions accessed via Blueprints to define weapon behaviour. Change these to define the weapon you desire.

By default, there will be an empty root component node. This represents the core of your weapon, corresponding to your root skeletal mesh. If the 'Node Details' panel has not opened by default, open it via the 'Window' menu.

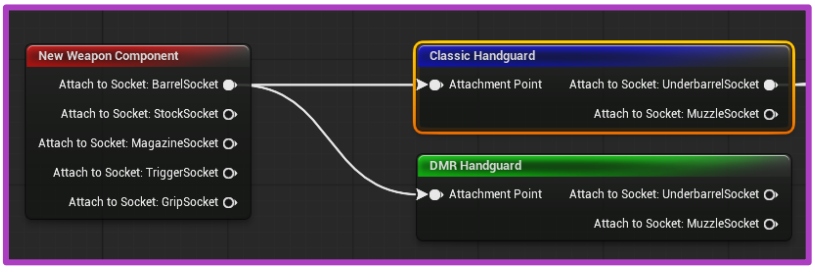


Select the root node. This will open the node's properties in the 'Node Details' panel, allowing you to change the properties of it. The node's name will update with the 'Title', and 'Property Deltas' allow for changing values depending on whether the component is enabled (explained fully later). Set your skeletal mesh asset in the 'Base Mesh' field properties, and this will populate the node with a pin for every skeletal mesh socket in the asset.



Dragging off from one of these nodes will allow you to create a new node, which will then connect to the previous one. These nodes act almost entirely the same as the root node, with the differences being the static mesh selection instead of skeletal and the ability to add and delete these nodes as desired.

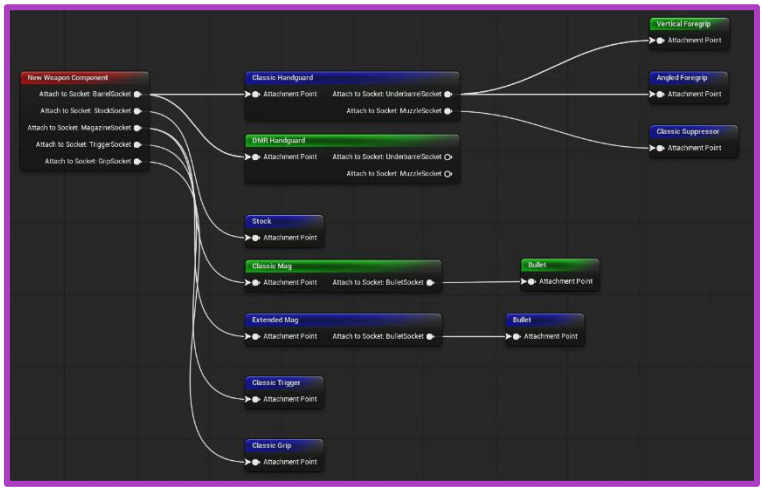
You will notice that the new node is blue if you created it and connected it to a previous pin. This means that this node is a 'default' node. Default nodes are what the weapon uses to build the base state of the weapon, e.g., setting what barrel will be used by the weapon before any runtime component swaps are made. Non-default nodes are green.



Every socket pin can have multiple nodes connected to it, allowing for component options to be defined. These can then be enabled and disabled via Blueprints later. For every socket pin, only one connected node can be the default node.

To make a non-default node default, right-click it and then click ‘Set Default’ from the context menu. This will turn the node blue to signify the change, and any defaults in other trees for the same socket pin will become non-default.

Build up your first weapon using the assets you have chosen:



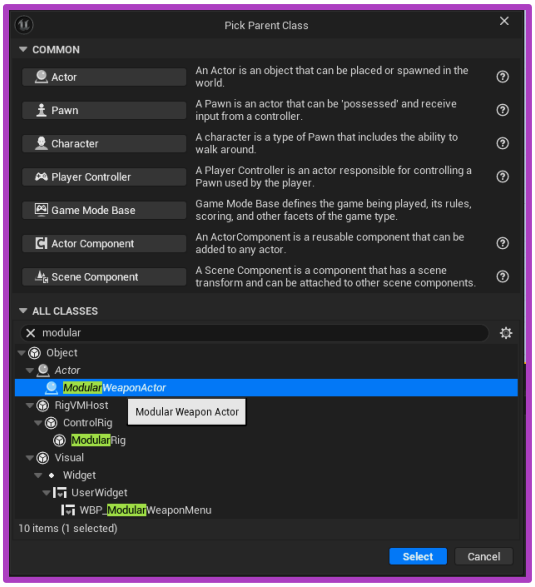
AFFECTING WEAPON FUNCTION USING PROPERTY DELTAS

To allow your weapon components to affect the function of the weapon outside of visuals, you need to define ‘Property Deltas’. This is a map within each node and is used to tell the weapon which variables should be changed when the component is active. Each property has its mapped component values summed whenever the weapon is ‘built’, and this will attempt to change a corresponding variable by name in the weapon Blueprint (see next section).

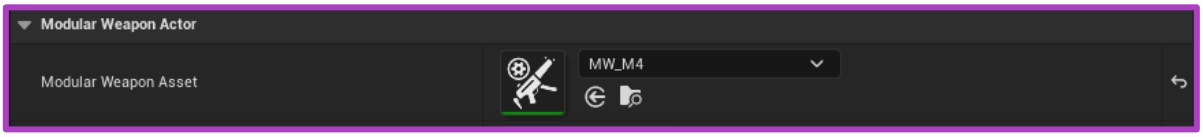
Each component can have any number of deltas. For now, this is limited to float variables, however these can be used to represent other data types and can be manipulated in the weapon Blueprint.

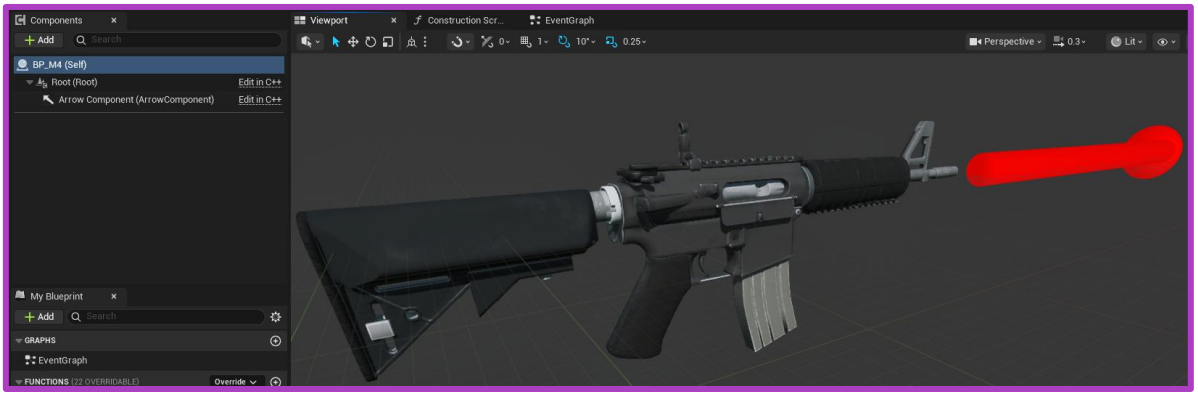
CREATING A WEAPON BLUEPRINT

Now that you have defined your modular weapon it is time to create the Blueprint which puts the asset into practice. Create a new Blueprint class and inherit from the ‘ModularWeaponActor’ class. This class contains the functionality for building any weapon from a given modular weapon asset and provides the necessary methods to define the weapon’s physical functionality.



In the details panel of this new class, set the weapon asset which you created. This will cause the weapon to build on compilation, and you can view the weapon with all the default components selected. For now, defaults are simply the first node the weapon finds which is connected to a socket, however in future this will be able to be set directly in the tree.





Create float variables for each of your 'Property Deltas' ensuring the names directly match the ones in the asset. You can then use these within your weapon functionality definitions.

Finally, create your weapon logic. The weapon actor class provides some methods and variables which can be accessed in Blueprint and are designed to make weapon logic as modular and easy to create as possible. This concludes the basics of creating a modular weapon, and the in-built methods are explained within the next section.

BLUEPRINT NODES

PHYSICAL ACTIONS

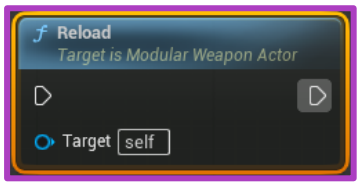
FIRE

The fire function fires the weapon using data from the weapon asset. It checks that the gun is not on cooldown, is not reloading and has sufficient ammo before firing. It then adjusts the look rotation depending on a test trace to allow projectiles to follow an accurate arc to the target if used.

If the weapon is a projectile weapon, the selected projectile class is used to spawn and fire. In future, there will be a pooling option for projectiles.

If the weapon is a hit-scan weapon, a simple line trace is used instead.

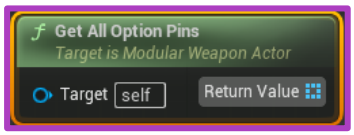
RELOAD



The reload function simply reloads the weapon from reserves. It checks that the weapon is not already reloading, there are reserves present and the magazine is not full before performing a reload, which takes time before placing the maximum possible number of reserves into the magazine.

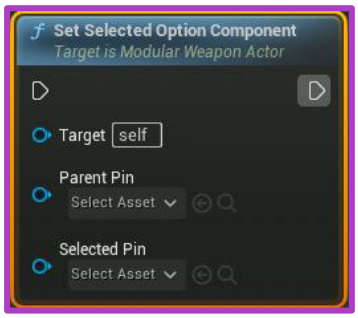
COMPONENT MODIFICATION

GET ALL OPTION PINS



This function returns every output pin in the modular weapon asset graph with multiple connections. This is useful for finding every component which may have options and thus building UI etc. to display these options.

SET SELECTED OPTION COMPONENT



By taking a parent pin and a selected option pin, this function enables the component which has the selected pin as its input pin. This ripples through the tree, disabling any attachments which were connected to the previous option.

MISCELLANEOUS

SET SKIN

This function applies a given material across the entire weapon and allows for passing in an array of every material slot you want affected by this skin.

RESET SKIN

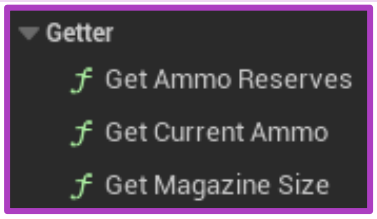
This function removes any weapon skins applied and returns the weapon's materials back to the mesh defaults.

UTILITY

CONVERT FIRE RATE TO COOLDOWN TIME

A simple helper function to convert rounds per minute into a cooldown period in seconds.

GETTERS



These functions return the related variable values which are not usually exposed to Blueprint for encapsulation.

ADDITIONAL INFORMATION

If you have any questions or requests regarding this tool, please contact me at any of the links provided in my website: https://owen-sims.dev/

I am always looking to improve the user experience of my tools, so please feel free to request any specific features you may desire.