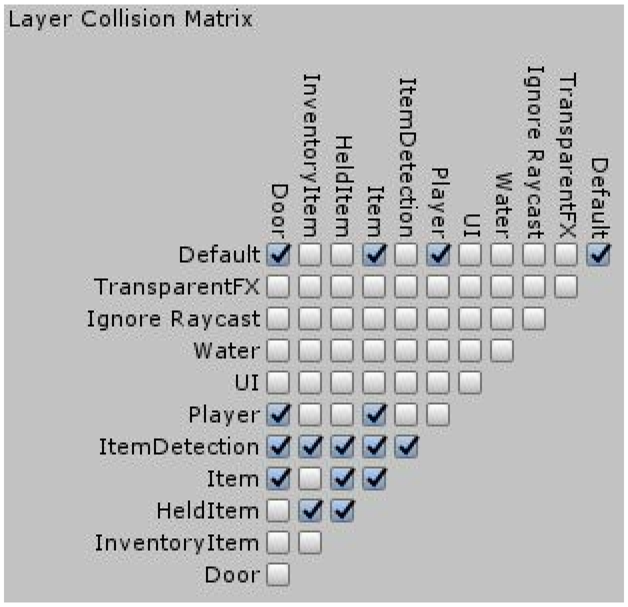
Setting up V Armory:

1. Import the V Armory package into your desired Unity project.

2. Import SteamVR into the project. Note, you do not need to import the SteamVR interaction system folder. A window will ask you “We’ve found a partial Steam VR input binding for ‘VArmory_Default_InputMap’ version ‘1’. Would you like to import it?”. Click “Import”. If you do not import the default input map you will have to follow steps 6,9, & 10 highlighted in red below.

4. Add the layers “Player”, “ItemDetection”, “Item”, “HeldItem”, “InventoryItem” and “Door”. Then set up the collision matrix in Edit > Project Settings > Physics


Layer Collision Matrix

	Door	InventoryItem	HeldItem	Item	ItemDetection	Player	UI	Water	Ignore Raycast	TransparentFX	Default
Default	✓	☐	☐	✓	☐	✓	☐	☐	☐	☐	✓
TransparentFX	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	
Ignore Raycast	☐	☐	☐	☐	☐	☐	☐	☐	☐		
Water	☐	☐	☐	☐	☐	☐	☐	☐			
UI	☐	☐	☐	☐	☐	☐	☐				
Player	✓	☐	☐	✓	☐	☐					
ItemDetection	✓	✓	✓	✓	✓						
Item	✓	☐	✓	✓							
HeldItem	☐	✓	✓								
InventoryItem	☐	☐									
Door	☐										

If you’re getting compiler errors at this point it’s likely the Steam VR or Unity version may be incompatible with the current version of V Armory. You can try an older version of Steam VR available on the Valve github. https://github.com/ValveSoftware/steamvr_unity_plugin

5. Add the tags “Slot” and “Interactable”.

Only follow the steps highlighted in red if you did not import an input binding with SteamVR from step 2.

6. If you can press play without errors you will now need to generate a Steam VR Input Action Set. Navigate to Window > Steam VR Input. It is recommended to have the inputs; Grip (boolean), Trigger (boolean), Menu (boolean), Touchpad (boolean), TriggerAxis (vector 1), TouchpadAxis (vector2), FireSelector (boolean), SlideStop (boolean), & EjectMag (boolean). The SteamVR_Behaviour_Pose component on each hand needs a pose action and input

source to ensure V Armory gets the correct left/right input. I prefer to call these LeftPose and RightPose in the input map.

Click “Save and Generate”.

7. Window > SteamVR Input > Open Binding UI. This will open a browser page. Bind the actions you created in the previous step to buttons on your respective controller in the opened browser page. Make sure you bind the LeftPose and RightPose pose actions. The hands will not function unless you do this.

8. ***Important information*** All interactable objects in the scene require the component “Interaction Volume”. I’ll refer to Interaction Volumes as IVs from now on. IVs require a collider to be detected by the hand. There are four primary fields for IVs; “Start Interacting Input ID”, “End Interacting Input ID”, “Start Interaction Mode”, & “End Interaction Mode”. These will need to be set on the IV before the object can be interacted with. The start/end input id should be set to any of the Steam VR boolean input actions. Ie grip, trigger, etc. The start/end interaction mode has 5 options, Press, Press Down, Press Up, Second Press Up, and none. For example if you wanted to hold an item when grip is held and drop it when grip is released you would set the start/end interaction input id to “Grip”, and the start interaction mode to “Press Down”, and the end interaction mode to “Press Up”. If you don’t want to drop the item the first time grip is released (toggle hold) simply set the end interaction mode to “Second Press Up”.

9. Multiple objects in the demo scene require IV input ids/modes to be set via the inspector.The “CharacterControllerMovement” components needs a Steam VR vector 2 action set. All “Item” components and components that inherit from “Item”. Firearm slides, magwells, and attachment sockets.

10. Firearms also have a few local inputs that need to be manually set in the inspector. Local inputs are inputs that only register once an Item is picked up. The “FireSelector” component (located on the root of every firearm) needs the “Fire Selector Input” set in the inspector. The “SocketSlide” or magwell has an “Eject Input” that is optional. Lastly the Firearm Slide component needs a “Slide Stop Input” set in the inspector if “Stop Action on Empty Mag” is set to true on the Firearm component.

11. All done. The demo scene should be functional at this point.

Setting Up Custom Models:

It is strongly recommended to keep the root of any object / item as well as any transform that has objects being attached to it scaled to 1,1,1. This will avoid distorting the scale of a transform when parenting/unparenting or rotating the object as a child. If you need to resize a mesh / model create an empty game object. Make sure the scale is 1,1,1. Attach the model to the newly created empty game object and resize as needed. Add all of the necessary components

(Item, IV, Rigidbody, Collider, etc) to the new game object instead of the scaled game object with the mesh.

Try to keep transform hierarchies as shallow as possible. Meaning keep the amount of objects with children of children and so on to a minimum for performance reasons.

- Every interactable item requires 3 components on the root object. A collider, rigidbody, and Interaction Volume. Simple grabbable objects require an Item component or one if its deriving classes (Firearm, Magazine, Bullet, Attachment, etc) on the same root object.

The first thing you'll need to set up in the inspector is the Interaction Volume which will be referred to as IV from now on. All IVs must be tagged with "Interactable" to be found. Any interactable object you create, not just items, will require one of these components. All IVs require a rigidbody and collider. IVs are responsible for receiving input from a hand and calling events which interactables subscribe desired behavior to. The most basic example of this is picking up an item. An item that subscribes a Grab method to the IVs start event will be grabbed when an IV receives the right input from the hand. A description of all the inspector variables is included below to help you set up the IV.

After the IV is done add a Firearm component to the same root object. Drag the IV you added in the previous step into the Primary Grip field on the new Firearm component. To set up a second grip position for two handed weapons, set up another IV and drag it into Secondary Grip field. To allow for items to be harnessed to inventory slots (auto equip to the harness slot when dropped) simply add another IV to the root object and drag it into Harness Input. At this point if you press play you should be able to pick up and drop the weapon. Use the Position and Rotation offset to correct the orientation of the item in the hand. This can be easily set up by clicking on the "Set Controller Offset" button at the bottom of the Item script in the inspector. The button will spawn a controller. Move the item relative to the controller as needed and click the inspector button again to save the offsets.

- Every weapon requires a few basic firearm parts as components including; fire selector, tigger, chamber, and muzzle.

To start create an empty game object and name it "**Chamber**". Now add a chamber component to the object. The chambered bullets position and rotation will be set to the chambers. Make sure the chamber is positioned and rotated correctly. An easy way to do this is to add the bullet that will be chambered, set its local position and rotation to zero, and then move the chamber in the correct location. Remove the bullet if it is no longer wanted. If you wish to add a bullet to an open chamber there must be a collider set as a trigger and the bullet needs to have the tag in the Bullet Tag field. If you want to be able to grab a bullet out of an open chamber add an IV to it.

Even if a firearm only has one firemode a **Fire Selector** component still needs to be added somewhere in the firearms hierarchy. Add the Fire Selector component to the respective mesh on the gun if there is one. If not add it to the root object. If a fire selector pivot does not rotate around the right position create an empty game object and position it to where you would expect the fire selector to rotate around. Make the fire selector mesh a child of the new object. Move the Fire Selector component to the new object as well.

If no separated **trigger** mesh exists on the weapon create an empty game object as a child of the weapon and add a Trigger component. If the trigger mesh's pivot doesn't rotate around the correct position follow the same steps as the fire selector.

The **muzzle** is not an actual component but a gameobject representing the position and rotation of the bullet when it is fired. Simply create an empty gameobject attached to the gun and rename it "Muzzle". The name is not cap sensitive.

Most Firearms require a slide and a magwell.

To add a **slide** select the respective separated slide mesh on the weapon. Start by adding an IV along with the necessary collider & rigidbody components. Then add the Firearm Slide component. Drag the IV into Interaction Volume. Move the slide to the desired resting position and click "Set Slide Start Position". This button will set the Start Position transform position to the slides current position. If the field is empty a game object will be created. "Set Slide End Position" functions the same but for the slide position when it's pulled all the way back. You can use the "Toggle Slide Position" button to make sure the slide positions are correct.

The **magwell** is set up in a similar way but instead uses a Socket Slide component on a child game object attached to the firearm. Add the magazine that will be used with this magwell. Move the magazine to the desired position and click "Set Loaded Position" / "Set Unloaded Position" respectively. You can use the "Toggle Slide Position" button to make sure the slide positions are correct. The Socket Slide component still needs the Accepted Tag field set to a tag the magazine will have. Loaded Center and Loaded Size are the size and center of the IV collider and need to be set and ideally should represent the collision of a loaded magazine. The collider size/center when unloaded will default to the size/center it had at start.

If a weapon is intended to have sound effects add a Firearm SFX_Manager component to the root.

If all of the editor inspector variables are set correctly you should now have a fully functioning firearm.

Index -

Interactables > Attachment Socket, Slide, Firearm Slide, & Socket Slide
Firearm Parts > Trigger, Fire Selector, Chamber, Hammer, & Revolver Hammer
Recoil > Recoil Axis, Recoil Struct, & Recoil Manager

CORE
{

Slot: Acts as a slot for items to be stored on. Slots can restrict what items are allowed to be attached to them by size or tag. Has a public reference to the stored item if one exists.

Inspector Variables:

Offset - The transform the stored item will be attached to.
Size Limit - Items with a size greater than this limit cannot be stored on this slot.
Accepted Tags - Only items with one of these tags can be stored on this slot.
Pose Item - Calls the items Pose method if set to true.
Mesh - The mesh renderer of the interaction sphere.
Highlight Material - The material the interaction sphere is set to when there is a potential interaction.

Exposed Variables:

Has Item - Get only boolean property that returns true if an item is stored on this slot.
Stackable Stored Item - Returns true if the stored item is stackable. Get only.
Stored Item - Reference to the stored item on this slot if one exists.
Stacked Items - A list of the stacked items stored in this slot.
Stack Is Full - Get only boolean property that returns true if the stack is full.
Stack Is Empty - Get only boolean property that returns true if the stack is empty.

Exposed Functions:

Unstore Item Light - Sets the stored item variable to null.
Unstore Item - Calls the DetachSlot() method on the stored item.
_Highlight - Highlights the inventory slot.
_Unhighlight - Un-highlights the inventory slot.
SetVisibility - Disables/Enables the slot mesh.
Valid Item - Returns true if the items is allowed to be stored on this slot
AddToStack - Adds an item to the stack on this slot.

Hand: Inherits from slot. There are two Hand components attached to each of the controllers in the Steam VR Camera Rig. These are responsible for collecting any IV that enter the controllers sphere trigger. The collected IV are separated into groups by input. Each group is sorted by how viable an interaction is. Input is then attempted on the most viable IV from each input group. If the input is successful the IV will be saved in the Hands Interacting Volume variable. Interactions will be culled if they are restrained (see Item & IV) or if they are currently being held and overlap (see IV) is set to false.

Inspector Variables:

Tomato Presence - Hides the controller and interaction sphere when interacting.
Drop Grab - Allows the hand to interact with items directly under it even if the item isn't inside the interaction sphere trigger.
Point Grab - Allows interaction with items at a distance via a line from the hand.
Trigger Input - Steam VR boolean action for local trigger input on items.
Touchpad Input - Steam VR boolean action for local touchpad input on items.
Trigger Rotation - Normalized rotation of the trigger.
Touchpad Axis - Returns the position of the touchpad axis.
Auto Grab - Auto grabs secondary grip positions on items if an empty hand is near.
Auto Drop - Auto drops secondary grip positions on items when the primary hand rotates away from the secondary or the secondary hand moves too far away from the grip position.

Exposed Variables:

Input Source - Reference to the Steam VR input source.
Sibling - Get only reference to the other hand.
Interacting Volume - The IV the hand is currently interacting with.

Exposed Functions:

Get Closest Valid Slot - Returns the closest valid slot to store the provided item onto.
Grab From Stack - Starts interacting with the item at the top of the stack in this hand.
Add To Stack - Add a new item to the stack on this hand if possible.

Interaction Volume (IV): The interaction volume (IV) is responsible for receiving start and end input from the Hands and relaying that input as events (_StartInteraction & _EndInteraction) to interactables that have subscribed functions. The interaction volume saves a reference to the hand currently interacting with it.

Inspector Variables:

Highlight - This gameobject will be enabled / disabled to highlight the interactable.
Priority - Increasing this value will make it easier to interact with when overlapping other IV of the with the same start input.

Overlap - The IV's current interacting can be overridden when this is set to true.
Start Input ID - The Steam VR action input to begin interaction with this IV.
End Input ID - The Steam VR action input to end interaction with this IV.
Start Interaction Mode - The type of input that must be detected to start interaction. (press, down, up, second up, none)
End Interaction Mode - The type of input that must be detected to end interaction. (press, down, up, second up, none)
Early Exit Condition - If the hand leaves a bounds *(iv collider, sphere, box)* the interaction will end. Ignored when set to *none*.

Exposed Variables:

Hand - A reference to the hand currently interacting with this IV.
_StartInteraction - The event called when a hand begins interacting with this IV.
_EndInteraction - The event called when a hand ends interacting with this IV.
Restrained - The IV cannot be interacted with while this is set to true.

Exposed Functions:

Attempt Interaction (Hand hand) - Starts interacting with hand if the required input is detected.
Force Start Interaction (Hand hand) - Bypasses the input required to start interacting.
Stop Interaction - Stops the interaction.

Item: Item is the base class for every interactable object that should be able to be grabbed and dropped. Item contains two main interaction volume fields; Primary Grip and Secondary Grip. The Items Primary/Secondary Grab/Drop methods are subscribed to the IVs _StartInteraction & _EndInteraction events respectively. Items are responsible for saving the offsets (position,rotation,aligned) relative to the controller when held.

Inspector Variables:

Primary Grip - An IV responsible for interacting with the items primary grip.
Secondary Grip - An IV responsible for interacting with the items secondary grip.
Harness Input - An IV responsible for harnessing the item to the slot it's stored on.
On Attach - The physics settings applied when the weapon is grabbed.
On Detach - The physics settings applied when the weapon is dropped entirely.
Position Offset - An offset applied to the positional pose of the item relative to the controller.
Rotation Offset - An offset applied to the rotational pose of the item relative to the controller.

Aligned Rotation Offset - An offset applied to the rotational pose of the item relative to the controller when the item is being held by both hands and the poseType is set to Two Hand Aligned.
PoseType - The way in which the item will be posed. Static: the item will not move relative to the controller when grabbed. Position: The item will move to its positional offset but the item will remain in the same rotation relative to the controller when it was grabbed. Position and Rotation: The item will move and rotate to its positional and rotational offsets. Two Hand Aligned: Acts like position and rotation when held by the primary grip only and like static when held by the secondary grip only. When both grips are held the item will align to the direction between the primary hand and the secondary hand.
Set PositionSpeed - Scales the speed at which Set Position Curve is evaluated.
Set Rotation Speed - Scales the speed at which Set Rotation Curve is evaluated.
Set Two Hand Rotation Speed - Scales the speed at which Set Rotation Curve is evaluated when the item is held with two hands.
Set Position Curve - Animation curve used to set the position of the item to its positional offset over 1 second / Set Position Speed.
Set Rotation Curve - Animation curve used to set the rotation of the item to its rotational offset over 1 second / Set Rotation Speed.
Auto Drop Dot - The dot product the secondary hand needs to be away from the z axis of the primary hand to auto drop.
Size - If this item is bigger than a slots sizeLimit it will not be able to be stored on that slot.
Storable - Is the item able to be stored on slots at all?
Stackable - Toggles the ability to stack this item. There is no support to avoid stacking different items.

Exposed Variables:

Rb - The rigidbody attached to this item
Col - The collider attached to this item
Velocity - The averaged velocity the item is going over x frames
Primary Grip - The primary IV used to grab and drop this item.
Primary Hand - The hand interacting with the primary grip IV.
Secondary Hand - The hand interacting with the secondary grip IV.
Slot - The slot this item is stored on.
Restrained - This items primary and secondary grip IVs will not be able to be interacted with while this is set to true.

Exposed Functions:

Set Physics - Sets whether or not; the collider is enabled, the is a trigger, the rigidbody is kinematic, the rigidbody uses gravity, the item is parented to the controller.

Set Kinematic - Sets Rb to kinematic.
Attach(Hand) - Force starts an interaction between this item and the provided hand.
Attach(Slot) - Forcibly stores the item on the provided slot.
Detach Without Storing - A way to drop the item without storing it on a slot.
Detach - Stops interaction with both the primary and secondary grip IVs.
Detach Slot Light - Only removes references of the item from the slot and the slot from the item.
Detach Slot - Detaches the item from the slot it's currently stored on.
Pose - Animates the item to its position and rotation offsets.

}

ITEMS (The following classes inherit from Item)
{

Firearm: Holds references to and connects each firearm part described further in the documentation. Subscribes functions to the magwell and slide interactables.

Editor Variables:

Stack Trigger Pulls - Will fire as soon as the slide rests if the trigger was released and once again pulled while the weapon was firing.
Eject Mag With Local Input - Ejects the magazine if touchpad is pressed.
Muzzle Velocity - The force applied to the projectile. Uses the muzzle velocity on the bullet if this is 0.
Spread - Random force applied to the bullet.
Slide Eject Velocity - The velocity the slide must be moving at to eject bullets.
Slide Stop Function - Either turn on, turn off, or toggle the slide stop.
Slide Stop Touchpad Direction - The direction the touchpad must be pressed to call the Slide Stop Function.

Magazine: Capable of storing and dispensing rounds. Maximum Rounds dictates the magazines capacity. Has a tag variable to ignore different types of bullets. The tag must be set in order to store rounds. Saves the physical round and the order it was loaded to be fired later.

Editor Variables:

Eject - The position and rotation of the bullet when it's ejected from the magazine.
Bullets - Mesh renderers that are enabled based on the number of rounds loaded.
Bullet Tag - The tag bullets must have to be loaded into this magazine.
Bullet Clone - Used to spawn new bullets if saved rounds count is less than current rounds. Called on start.
Swipe To Eject Distance - The distance needed to be swiped to eject a round.
Max Rounds - Capacity of the magazine.

Current Rounds - Amount of rounds currently loaded into the magazine.
Require Held - Do bullets need to be held to be loaded?

Exposed Variables:

MaxRounds
CurrentRounds
Full
Empty

Exposed Functions:

Update Bullets - Enables mesh renderers from bullets to match the amount of loaded bullets.
Load Bullet - Loads the provided bullet into the magazine

Bullet: Not the projectile itself but the Item containing it (has a variable that's a reference to the projectile attached to the bullet gameobject to enable when fired). A reference to a spent and live mesh is toggled between when fired. By default setting a bullet to spent also sets the bullet to restricted and destroys it shortly after.

Editor Variables:

Spent - The bullet has already been fired when set to true
Casing Mesh - The mesh the bullet will be assigned when it is spent.
Cartridge Mesh - The mesh the bullet will be assigned when it is not spent.
Projectile - The projectile enabled when the bullet is fired.

Exposed Variables:

Spent - Get only boolean property. The bullet cannot be fired if this returns true.
Chambered - A chambered bullet cannot be loaded into a chamber. Prevents overlapping chambers from "stealing" a neighbors bullet.

Exposed Functions:

Fire(Transform muzzle) - Sets the bullets mesh to Casing Mesh and enables the projectile at the position and rotation of the muzzle argument.

Projectile: The projectile is a disabled gameobject on the a bullet. When the bullet is fired the projectile is enabled and a force equal to Muzzle Velocity is applied. Impact Effects holds the effect (prefab) to be spawned when the projectile hits an object with a specified tag. The projectile is a physical object. Use On Collision Enter to call functions on objects hit by the projectile. For example, in On Collision Enter you could get the health component of the object hit and call an apply damage method.

}

INTERACTABLES

{

An IV variable will allows interactables to subscribe functions to the start and end methods of said IV.

Attachment Socket (AS): Attachment Sockets are restrained (see IV) when no attachment is equipped. When an attachment enters the trigger collider on an empty AS a few checks are performed before attaching. The attachment will have to be with in x distance to the AS and two axis on the attachment must match two axis on the AS. These axis can be set (forward, back, left, right, up, down) through PrimaryAttachDotAxis and SecondaryAttachDotAxis on the AS and attachment respectively.

Slide: Moves between a start and end position based on the position of the interacting hand. Slide Position describes the position of slide between the start and end point. 1 being at start and 0 being at end. 0.5 would be in the middle. Events _OnReachStart and _OnReachEnd are used by the Firearm to chamber bullets from a loaded magazine, eject the chambered bullet, and load/eject mags from mag wells. These events use a hysteresis meaning the event won't be called until the slider is moved a certain distance away from the event (start / end) position and once again back to the event position. Contains coroutines to animate the slide back, forward, and both.

Editor Variables:

> **Interaction Volume -** The IV responsible for grabbing / releasing the slide.
> **StartPos -** The transform that defines the start of the slide.
> **EndPos -** The Transform that defines the end of the slide.
> **Slider Position -** The position of the slider along the slide from 0 to 1. 0 being the end pos and 1 being the start pos
> **End Hysteresis -** The position the slide must move away from the end position before the _OnReachedEnd event can be called again.
> **Start Hysteresis -** The position the slide must move away from the start position before the _OnReachedStart event can be called again.
> **Continuous -** By default if the hand extends beyond the limits of the slide it will have to be moved back to the limit before the slide will move again. If this is set to true and the

hand is extended beyond the limits of the slide, as soon as the hand begins to move a direction the slide will follow.

Exposed Variables:

_OnReachedEnd - Called when the slider reaches the end position of the slide.
_OnReachedStart - Called when the slider reaches the start position of the slide.
_OnFullCycle - Called when the slide reaches the start position and then the end position of the slide.
Slide Object - The gameobject being moved by the slide component.
Min Slider Distance - Limits how close the slider can get to the end of the slide.
Max Slider Distance - Limits how close the slider can get to the start of the slide.
Interaction Point - The transform used to move the slider.

Exposed Functions:

Force Stop - Forcibly stop the interaction with the slides IV.
Force Start - Force start an interaction with the provided hand and the slides IV.
Grab Slide - Starts moving the slide by the hand interacting with the IV.
Grab Slide(Transform newInteractPoint) - Moves the slide by the provided transform.
Detach Slide - Sets the interaction point to null. Overridden.
Set Slide Position - Sets the position of the slide between the start 1 and end 0 position.
Animate Slide - Animates the slide back and forward again.
Animate Slide Back - Animates the slide towards the end pos.
Animate Slide Forward - Animates the slide towards the start pos.

Firearm Slide: Added functionality of a Slide Stop that limits how close the slide can get to the start position. An event _PullPassedSlideStop is called when the slide position is moved passed slideStopPosition.

Editor Variables:

Animate Forward On Release - Does the slide return to the start position when released?
Slide Forward Speed - The speed at which the slider is animated towards the start position of the slide.
Slide Back Speed - The speed at which the slider is animated towards the end position of the slide.
Drop Slide Forward Speed - The speed the slider is moved towards the start position when released. Typically this is set to a lower value than Slide Forward Speed.
Slide Stop Position - The position at which the slide will stop when Slide Stop is true.

Catch Bullet Position - If there is no bullet in the loaded magazine when the slider passes over this point along the slide then no bullet will be chambered when Chamber Bullet From Magazine is called on the Firearm.
Slide Stop Object - The gameobject used to represent the slide stop.
Slide Stop Rotations - The rotations for the on and off position of the slide stop.

Exposed Variables:

Slide Stop - A set only property that enables / disables the behavior of the slide stop.
_PulledPassedSlideStop - An event called when the slide is pulled passed the slide stop position.
_CatchBullet - An event called when the slide passes over the catch bullet position of slide.
_RestingOnSlideStop - Called when the slider is stopped by the slide stop.
Resting On Slide Stop - Get only boolean property that is true when the slide was stopped by the slide stop and not moved since.

Exposed Functions:

Grab Slide - Overridden behavior of Slide only it now stops any current animations.
LockSlideStop - Sets Slide Stop to true.
Unlock Slide Stop - Sets Slide Stop to false.
Toggle Slide Stop - Toggles Slide Stop.
Animate Slide - Animates the slide back and forth.
Stop Slide Animations - Cancels any animations currently running on the slide.
Detach Slide - Overridden behavior from Slide so the slide drops forward when released.

Socket Slide (Inherits from Slide): Insert / remove an item into / from the slide. If the _OnReachEndHysteresis is called the attached item is detached and added back to the hand. If _OnReachStartHysteresis is called the item will be “loaded” and the hand will be forced to stop interacting with the Socket Slide. Eject Slider is used to remove an attached item with out directly interacting with the Socket Slide.

Editor Variables:

Accepted Tag - The tag the potential slider must have to be attached.
Primary Attach Dot Axis - The first axis that will be compared to for attachment.
Secondary Attach Dot Axis - The second axis that will be compared to for attachment. These axis can be as follows; forward, back, up, down, left, and right.
Primary Attach Dot Threshold - The dot product of the primaryAttachDotThreshold of the socket slide and potential slider must be greater than this value to attach.

Secondary Attach Dot Threshold - The dot product of the secondaryAttachDotThreshold of the socket slide and potential slider must be greater than this value to attach.
Attach Distance Threshold - The potential slider has to be closer than this distance.
Detach Slider Position Threshold - A threshold the slider can be away from the unloaded position of the slide before considering an early detach.
Detach Slider Distance Threshold - When detachSliderPositionThreshold is passed the distance away from the unloaded position of the slide before the attached slider is auto detached and reattached to the hand.
Load From Slot - Can items stored in slots be attached and loaded into this slider.
Eject Speed - The speed the slider will be ejected at when EjectSlider is called.
Loaded Center - The center of the trigger collider when a slider is loaded.
Loaded Size - The size of the trigger collider when a slider is loaded.
resetSlideOnLoad - Sets the collider as if it is unloaded by calling Clear Slider.

Exposed Variables:

_OnDetach - Called when the slider is detached.
_OnLoad - Called when a slider is attached.
_OnGrab - Called when a slider is loaded and the socket slide is interacted with.

Exposed Functions:

Grab Slide - Starts moving the slide if the interaction volume is being interacting with.
Clear Slider - Sets the collider as if it was unloaded. Doesn't actually do anything with the slide besides clear references.
Eject Slider - Ejects the slider.

Firearm Double Slide: Starts dual interaction with firearm Slide when firearm Slides position is closer to start (1) than itself. Used by the M4 to allow the bolt to move separately from the handle when firing or being stopped by slide Stop but still "catch" the bolt when pulling the handle back.

Hinge: Acts much like a slide but the value of slide is converted into rotational movement instead of translational movement.

Editor Variables:

Interaction Volume - The IV that drops / grabs this hinge.
Inverse Pull - Invert the direction the hinge is rotated when pulling up/down.
Axis - The axis the hinge rotates around.

Pull Distance - The distance to fully pull the hinge from its max rotation to its minimum one.
Max Angle - The max rotation the hinge can be set to.

Exposed Variables:

Hinge Hand - Set only Hand property. Starts moving the hinge when this is set.

Free Swing Hinge: When not being interacted with and swung around an axis the hinge will gradually close scaled by the how fast the hinge is being swung. Contains two events, _Open and _Close which act very similar to Slides own _OnReachStart and _OnReachEnd.

Editor Variables:

Inverse Free Swing - Invert the direction the hinge rotates when swinging Hinge around the rotation axis.
Open Speed - The speed at which the hinge is rotated open.
Close Speed - The speed at which the hinge is rotated closed.
Hinge - The parent item used to detect free swing rotation.
Locked - Is the hinge currently locked from rotating?
Angle Delta Threshold - The threshold difference in angle Hinge has to be from itself in the previous frame for the hinge to free swing.
Lock Hinge - Does the hinge lock when closed?

Exposed Variables:

_Open - Called when the hinge is opened.
_Closed - Called when the hinge is closed.
locked - The hinge is locked when this is set to true.

Exposed Functions:

Unlock - Unlocks and opens the hinge.

}

FIREARM PARTS

{

Trigger: Has a rotation at which the trigger pull is started and ended. Also positions / rotates the trigger by inputScale. Contains a separate start / end rotation if the weapons hammer is already cocked.

Editor Variables:

Trigger Release - Trigger rotation is mapped from 0 -1 and this value should be set between that range. This is the rotation the trigger calls the _TriggerReleased event.
Trigger Pull - Trigger rotation is mapped from 0 -1 and this value should be set between that range. This is the rotation the trigger calls the _TriggerPulled event.
Double Action Trigger Release - When the trigger is given a reference to a hammer and the hammer is cocked this will replace trigger Release as the rotation the _TriggerReleased event is called.
Double Action Trigger Pull - When the trigger is given a reference to a hammer and the hammer is cocked this will replace trigger Pull as the rotation the _TriggerPulled event is called.
Pose Type - *Rotation or Position*. Is the trigger rotated or translated when pulled?
Pose Axis - The axis being rotated or translated when pulled.
Pose Input Scale - How much with the trigger be rotated/translated when fully pulled?
Hammer - A reference to the hammer associated with the trigger.

Exposed Variables:

_TriggerPulled - An event called when the trigger passes passed the trigger Pull position.
_TriggerHeld - An event called when the trigger is passed the trigger Release position.
_TriggerReleased - An event called when the trigger passes the trigger Release position.
Released Trigger - Has the trigger been released?

Exposed Functions:

PoseTrigger(float input) - Must be called for the trigger to function.

Fire Selector: Create a list of available fire modes (safety, semi-auto, burst, full-auto) that will be cycled through when the fire mode input is pressed. There is another list of rotations for each respective fire mode. It is common for slides to be restricted when the fire mode is set to safety. SafetySlideLimit will limit slide movement to that value when safety is enabled.

Editor Variables:

Fire Mode - The current fire mode.
Available Fire Modes - Fire modes that will be cycled through when SwitchFireMode is called.
Fire Selector Rotations - A list of respective rotations for each available fire mode.

Safety Slide Limit - When the gun is on safety limit the slide from moving back passed this point.
TouchPad Direction - The direction the touchpad must be pressed to call SwitchFireMode.

Exposed Variables:

Fire Mode - The current fire mode. Get only property.

Exposed Functions:

Switch Fire Mode - Cycles to the next fire mode in available fire mode from the current.
Next Fire Mode - Returns the fire mode after the current one.

Chamber: Holds a reference to the bullet. Allows for bullets to be placed directly into the chamber. When a potential bullet is found the event _LoadBullet that takes a parameter of chamber will be called. The chamber will pass itself as the argument. Firearm subscribes Chamber Bullet(Chamber chamber) to the event. Chamber Bullet calls Load Potential Bullet on the chamber passed to it. This might seem like an unnecessary work around but Chamber Bullet is meant to be overridden if there is a situation where you don't want the chamber to be loadable. Like when the slide is closed on a bolt action or shotgun. Or when the revolver hinge is closed.

Editor Variables:

Bullet Tag - The tag the bullet must have to be directly loaded into this chamber.
Load Bullet Distance - The max distance away from the chamber the bullet can be before loading.
Require Held Bullets - Do the bullets have to be held to be loaded directly?
Eject Force - Force applied to the bullet when it is ejected.
Eject Torque - Torque applied to the bullet when it is ejected.
Eject - The bullets position and rotation will be set to this transform before ejecting.
Chamber Dot Axis - The local axis of the chamber to compare against BulletDotAxis before loading a bullet.
Bullet Dot Axis - The local axis of the bullet to compare against ChamberDotAxis before loading a bullet.
LoadBulletDot - Tolerance for how closely rotated the bullet must be to the chamber before loading.
Eject Direction - The direction of the force to eject the bullet as a local axis of the eject transform variable.
Torque Direction - The axis the torque is applied around when ejecting the bullet as a local axis of the eject transform variable.

Exposed Variables:

_LoadBullet - An event called when a bullet is loaded into this chamber.
Bullet - The bullet loaded into this chamber.

Exposed Functions:

Chamber Bullet(Bullet bullet) - Chambers the provided bullet.
Eject Bullet - Removes and ejects the bullet from this chamber.
Eject Bullet(Vector3 additional Velocity) - Applies an additional velocity when ejecting.
Load Potential Bullet - Attempts to load the potential bullet.

Hammer: RestHammerSpeed is how fast the hammer will fall when not being either in the cocked position or held by touchpad input. HammerMaxRotationAngle is how far the hammer must be rotated to reach the cocked position. Clicking down the touchpad and pulling trigger will decock the hammer. A double action hammer will be pulled with the trigger. This class is only intended to be used with Hammer Action Pistol and its children because Fire needs to be overridden to release the hammer and fire when it reaches its resting position instead of firing immediately.

Editor Variables:

Rest HammerS peed - The speed at which the hammer falls to its resting rotation.
Max Rotation - The max rotation the hammer can be pulled back.

Exposed Variables:

Hammer Rotation - The current rotation of the hammer scaled between 0 and 1.
_Fire - The event called when the hammer is fired and reaches its resting rotation.
_Lock - The event called when the hammer is pulled to its max rotation and locked.
Firing - Is the hammer currently falling after being released by a trigger pull?
Cocked - Is the hammer cocked back?

Exposed Functions:

Fire - Releases the hammer and calls _Fire when it reaches its resting rotation.
Lock Hammer - Locks the hammer back in a position to be fired.
Pull Hammer(Vector2 touchpad Axis, float slide Axis) - Moves the hammer.

Revolver Hammer (Inherits from Hammer): Adds the responsibility of rotating the cylinder along with pulling the hammer. Only intended to be used with Revolver.

Editor Variables:

Hinge - The revolver cylinder hinge.
Cyl - The transform used to represent the rotating cylinder.
End Cyl Rotation - The rotation the cylinder will be in when the hammer is pulled all the way back. Should be set to 360 / chambers before start (Assuming the cyl hasn't been rotated).

Exposed Variables:

_OnCockHammer - An event called when the hammer is locked back.

Exposed Functions: Same but overridden functions of Hammer. Only now they handle rotating the cylinder.

Muzzle: Not an actual component but it is critical an empty gameobject named "Muzzle" positioned where the bullet would fire and the z axis being the direction the bullet travels exists as a child of the weapon.

}
RECOIL
{

Recoil Manager: Has two Recoil (see RECOIL > Recoil Struct) variables. Rotation and translation. The local position of the position Offset is set to the current value of each respective Recoil Axis current value in translation. The same is done to the rotation Offset using rotation instead.

Editor Variables:

Decrease While Increasing - Will the target recoil be reduced while target recoil is higher than the current recoil?
Increase Chance - Chance for the increment to negative instead of positive. Used to simulate horizontal side to side recoil.
Limits - The target is clamped between the y (min) and x (max) values.
Increment - How much the target is increased every time the weapon is fired.
Decrease Target Speed - How fast the target is decreased per second.
Increase Speed - How fast recoil will reach the target when the target is greater than the current recoil.
Decrease Speed - How fast the recoil will reach the target when the target is less than the current recoil.
Two Hand Max Scale - The scale applied to the limits when the weapon is held with two hands.

Two Hand Increment Scale - The scale applied to the increment when the weapon is held with two hands.
Two Hand Decrease Target - The scale applied to the decrease target speed when the weapon is held with two hands.
Two Hand Increase Scale - The scale applied to the increase speed when the weapon is held with two hands.
Two Hand Decrease Scale - The scale applied to the decrease speed when the weapon is held with two hands.

Recoil: A struct containing a x,y, & z RecoilAxis.

Recoil Axis: A recoil axis is an abstract way of keeping track of recoil. Every time the weapon is fired Increase Recoil is called increasing target by increment.Target is decreased over time by decrease Target Speed by calling the function Decrease Recoil. Current is what the rotation / translation (whatever this recoil axis describes) should be set to. Current is being move towards the target every frame. If the target is greater than current increase Speed is used to increase current towards the target. Decrease Speed is used to decrease current if the target is less than current.
}