

Combat - User Manual

The combat system is a crucial system in the game.

By default, players, AI zombies, and defence towers are combatable entities that naturally carry attributes for combat such as blood, damage capacity, and so on. Custom combat targets are dependent on targetable entities and are tied to concepts such as assisted targeting and attackability.

Flexible use of the combat system allows you to create a customised set of battles that go even beyond the limitations of a shooter. However, shooter battles using FF's native guns are more fully supported, and if you want to make a battle that doesn't use guns (such as Sword & Sorcery's Battle combat), it may require more effort and resources. So this article is mainly about the combat system with the premise of fighting with officially provided guns, but maybe you will get more inspiration.

In this article, the following will be presented in order:

1. combatable entity
2. Customised combat entities
3. How to create combat entities.
4. How to make a battle example.
5. Expanded: homemade targets and guns that fire AOE ammo.

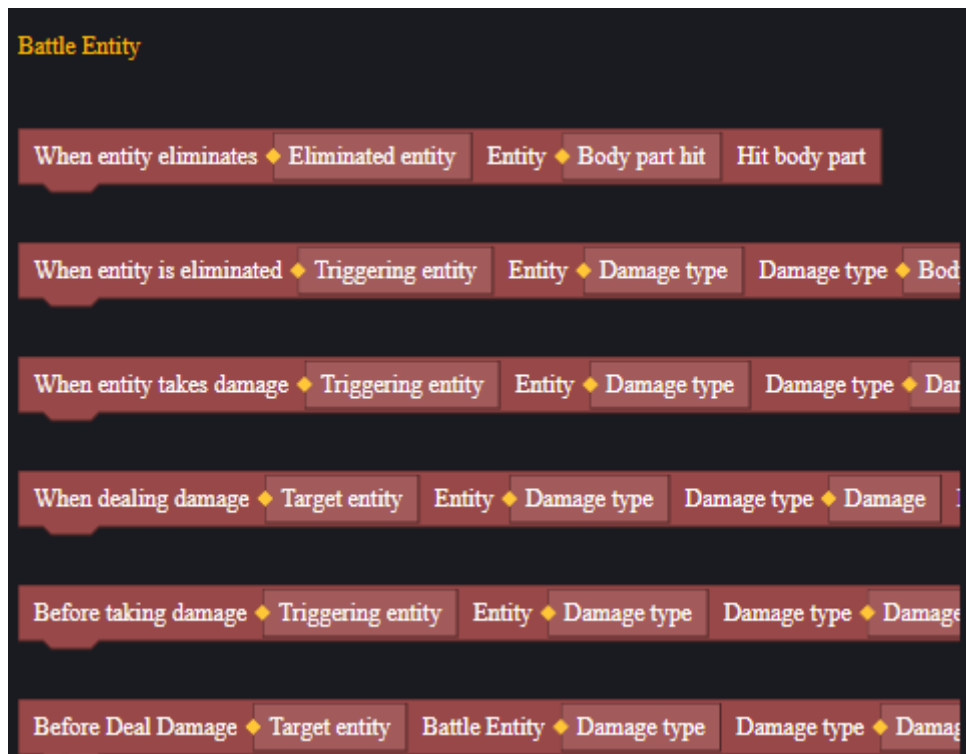
combatable entity

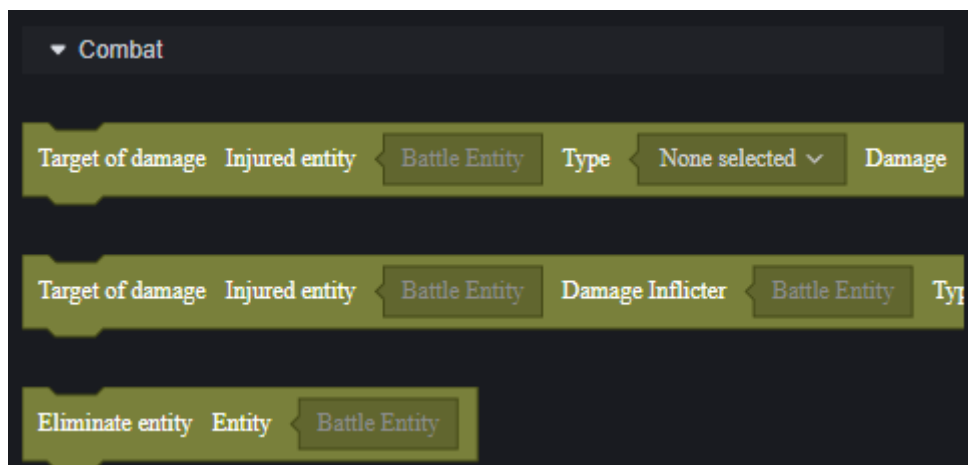
Combatable entities are entities that will participate in combat by default, currently only the player, AI zombies, and defence towers are combatable entities. Combat related events and interfaces can be used, and the Combatable Entity component does not support custom additions.

Combatable entities all have the ability to inflict damage and all have the blood attribute, and are killed/destroyed when their blood level reaches zero.

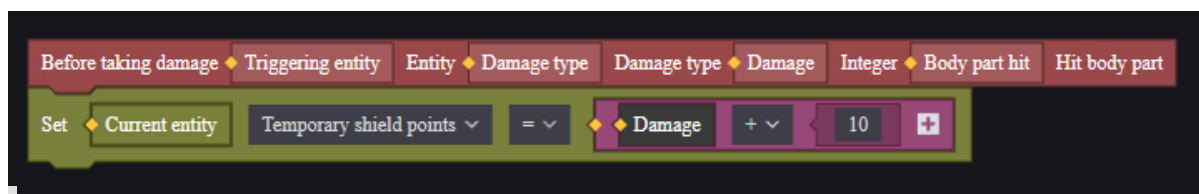
scripts

Combatable entities can use a number of events and interfaces that depend on themselves.





Players, AI zombies, and defence towers can also each use events and interfaces that depend on other entity components they have. Flexible use of these scripts allows for a degree of customisation and expansion of FF combat content.



Adds a temporary shield to yourself before each damage, the amount of damage is the value before the increase in damage reduction is applied, so it can't fully protect against headshot damage.

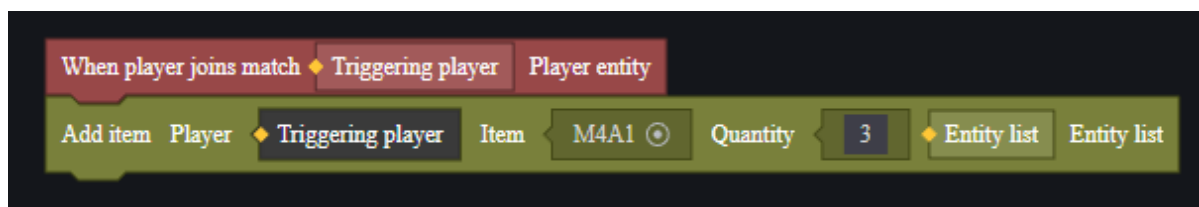
Firearms and props

Guns and props are indispensable to combat, and the vast majority of damage dealt by the player is dependent on them. Only the player can use guns and props directly in combat.

Firearms and props issued

Guns and props can be issued to players via scripts, and each player can carry two primary weapons, a secondary weapon, a melee weapon, and several props by default.

Props that exceed the carry limit will be dropped at the player's location when issued.



In addition, weapons and props or their generators can be placed directly on the field using level objects. It is also possible to sell weapons or props to the player through the shop. The method of distributing items to players depends on the game design.

Property Configuration

For guns and props placed in the scene via generators and level objects, the properties panel can be configured directly before starting the game.

▼ Weapon Generator...

Weapon type

Rifle

▼

Weapon

M4A1

▼

▼ Weapon entity...

Damage

29

^

▼

Effective Range

220

^

▼

Burst

0

^

▼

Magazine Storage

30

^

▼

Armor Penetration

0

^

▼

Minimum damage

11

^

▼

Firing interval

0.172

^

▼

Supports Akimbo

Not dual-wieldable

▼

Armor DMG multiplier

0

^

▼

Movement Speed

1

^

▼

Body shots DMG ratio

0

^

▼

Headshots DMG ratio

0

^

▼

Limb shots DMG ratio

0

^

▼

Extra DMG

0

^

▼

Extra damage on the downed

0

^

▼

Single shot bullet number

1

^

▼

Reload Speed

0.9

^

▼

Weapon skin ID

None added

⊕

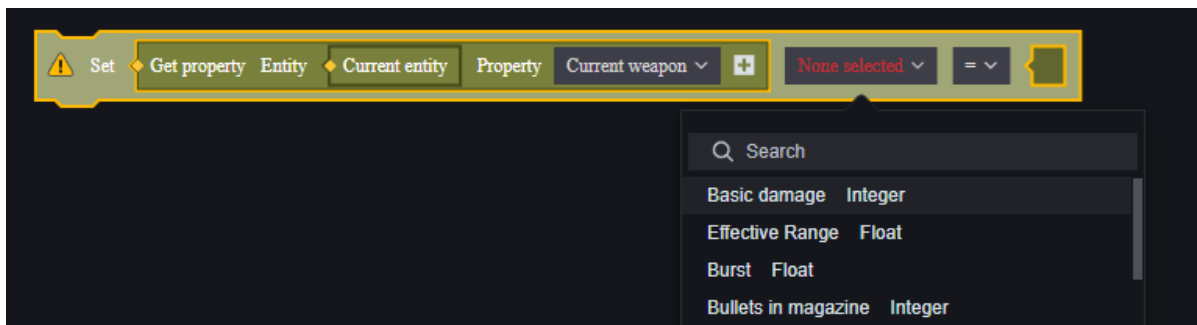
Prioritize player weapon skin

☒

Write weapon skin attribute

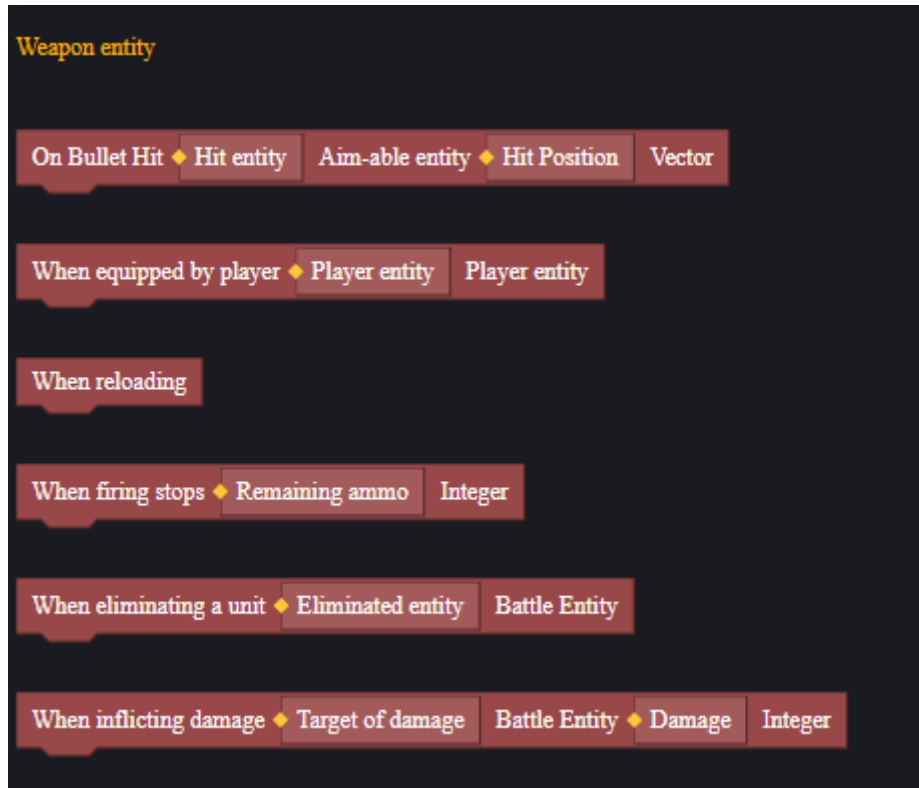
☒

For the need to adjust the properties of guns and props after the game has been run, this can be done through the script's setting properties.



Modify player's main weapon attributes


When attempting to modify gun properties via scripting, care needs to be taken to fetch the correct gun target. The weapons themselves also support some scripted events:



The script needs to be mounted on the weapon entity when using these events. This will be illustrated in action in the exploding bomb example below.

player (of computer games)

For players in combat, the player module provides the following configurations:

 Player

Description

Settings

Jump Height

100%

Respawns

☐

▼ Combat

Damage Multiplier

100%

Damage Taken Multiplier

100%

This determines the ratio of damage dealt and taken by the player.

In addition to the configuration of the battle, the player's basic attribute configuration will also have an impact on the battle, such as life value, energy, movement speed, and whether or not they can respawn.

▼ Basic Settings

Maximum HP

200

Maximum EP

200

Initial EP Proportion

0%

Movement Speed

100%

Jump Height

100%

Respawns

☒

Respawn Waiting Time

3s

Similarly, the player's attributes can be modified while the game is running by setting the attributes.

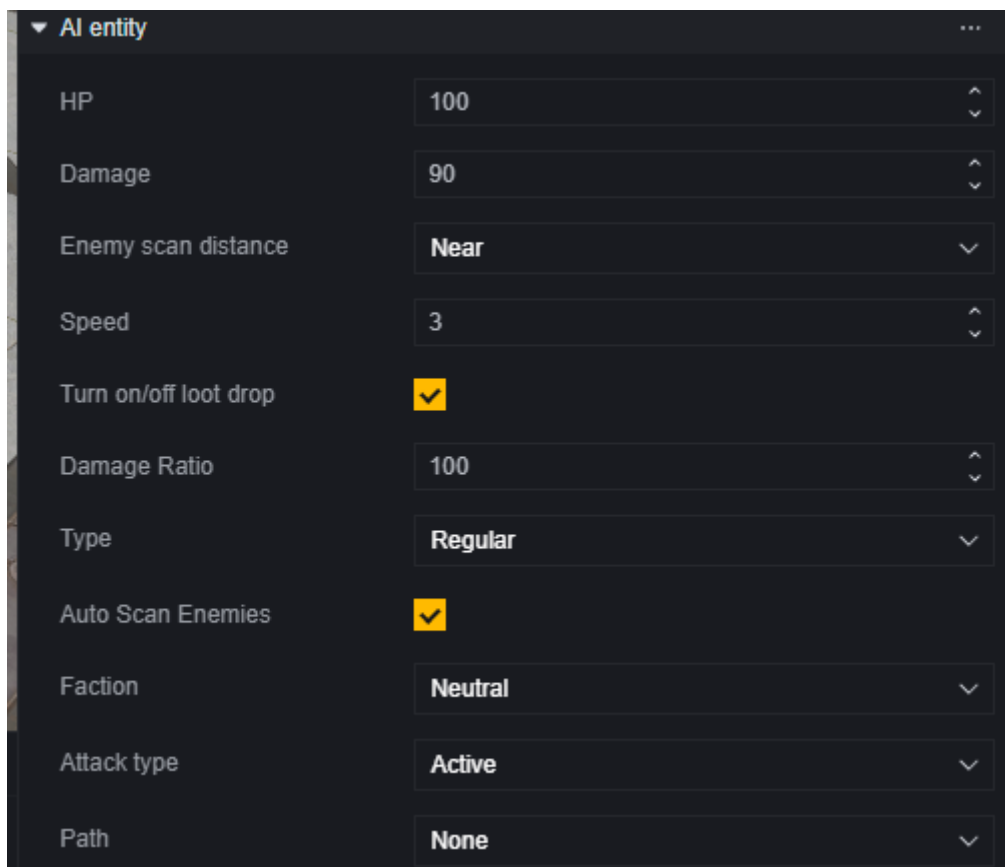


It is also possible to receive trigger signals from players involved in a battle or perform damage actions on players via battle-specific events.

Setting attributes in such a way as to modify the amount of blood is different from inflicting damage, and cannot trigger damage-related events, nor does it ignore invincibility, damage reduction, and other statuses. Other player-related instructions can be found in the corresponding documentation.

AI Zombie

An AI zombie is a monster that has a set of attributes exclusive to AI entities.

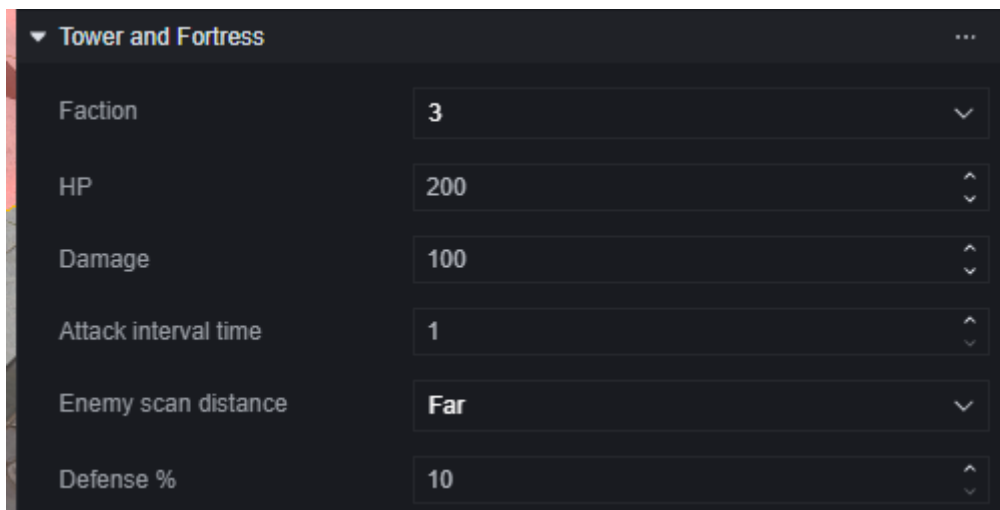


AI zombies will actively attack players with different formations, but will not attack other zombies or defence towers.

For other descriptions of AI zombies, please refer to the corresponding documentation.

defensive tower

Defence towers are similar to AI zombies and also have their own unique attribute configuration.

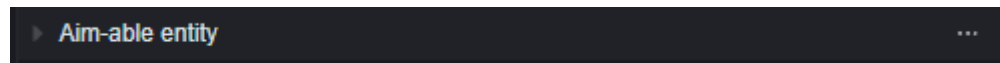


Defence towers will likewise only attack players with different formations, not another defence tower or zombie.

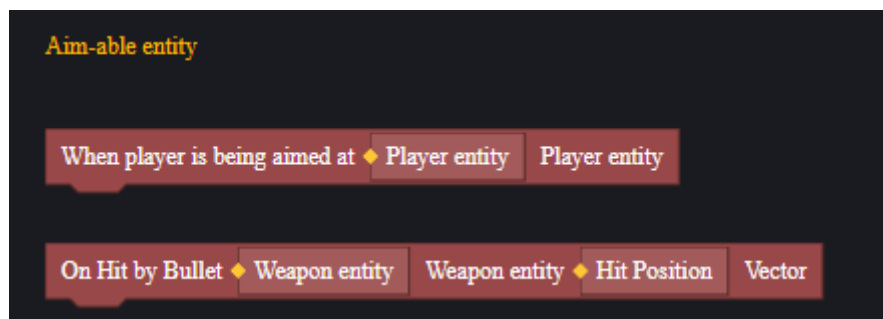
Customised combat entities

targetable entity

Without considering completely customising a combat system from scratch, fighting with officially supplied firearms, units that can be hit in combat need to have a **targetable entity** component mounted.

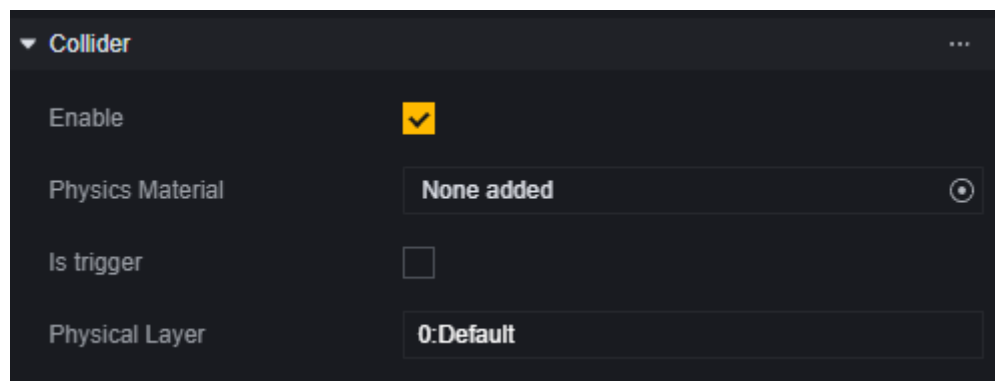


Units with targetable entity components mounted can trigger relevant events, which in turn complete the combat logic.



Players, zombies, and defence towers, all combatable entities, naturally carry this component.

The event implementation of the targetable entity component actually relies on collisions, so make sure you add a collision component and turn on collisions.



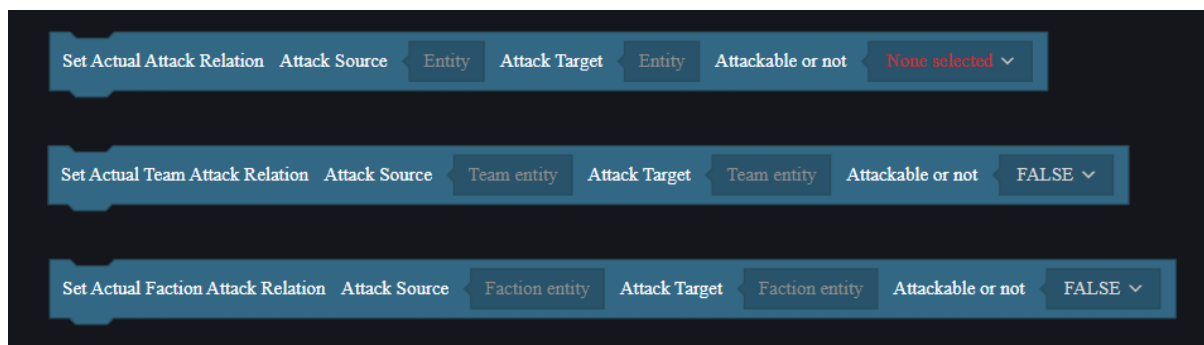
Aiming-assisted entities

For the player, an in-game aiming aid is provided. When firing, the collimator is automatically attached to the aimable entity. The aimable entity component is dependent on the **aimable entity** component.

However, for the targetable entity component to take effect, it is required that the targeting entity must be a

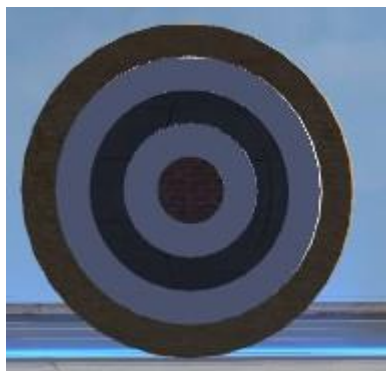


player attackable entity. For AI zombies, other players and defence towers, it is the current player attackable entity as long as **the camp** is different; for other targetable entities, the attackable relationship needs to be set manually in the script.



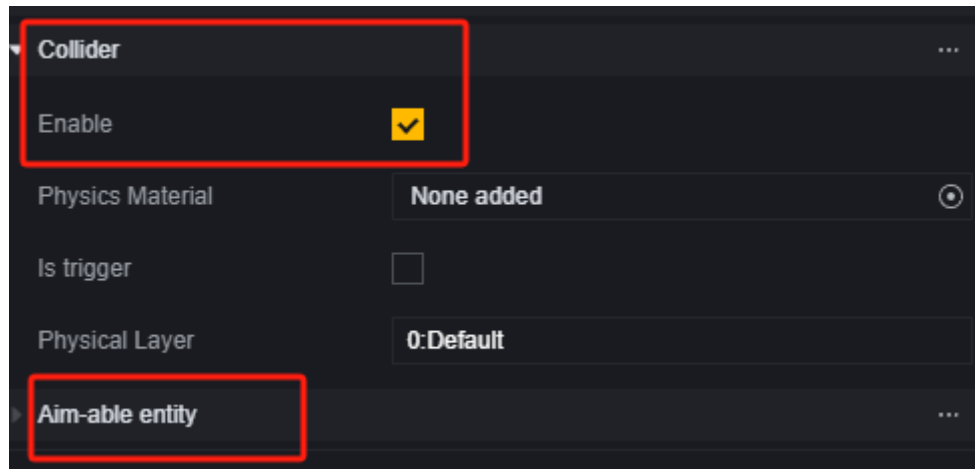
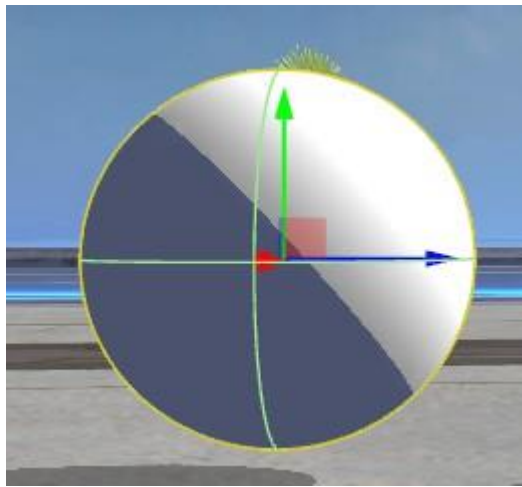
How to create a combat entity

In addition to the default combatable entities, you may want to add some targets to the map that can be shot at, such as markers.

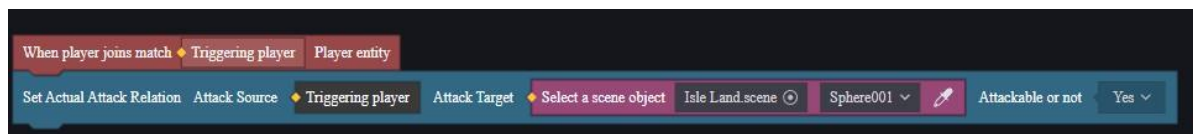
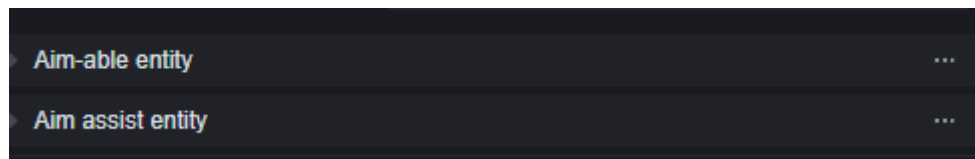


How to create a marker with functionality will be described later.

Using a basic sphere as an example, you will need to add targetable solid components to this object after you create it and make sure collisions are in effect.



If the object is needed to trigger the player's assisted targeting, an additional assistable targeting entity will need to be added and its relationship to the player's attack will need to be handled in the script.

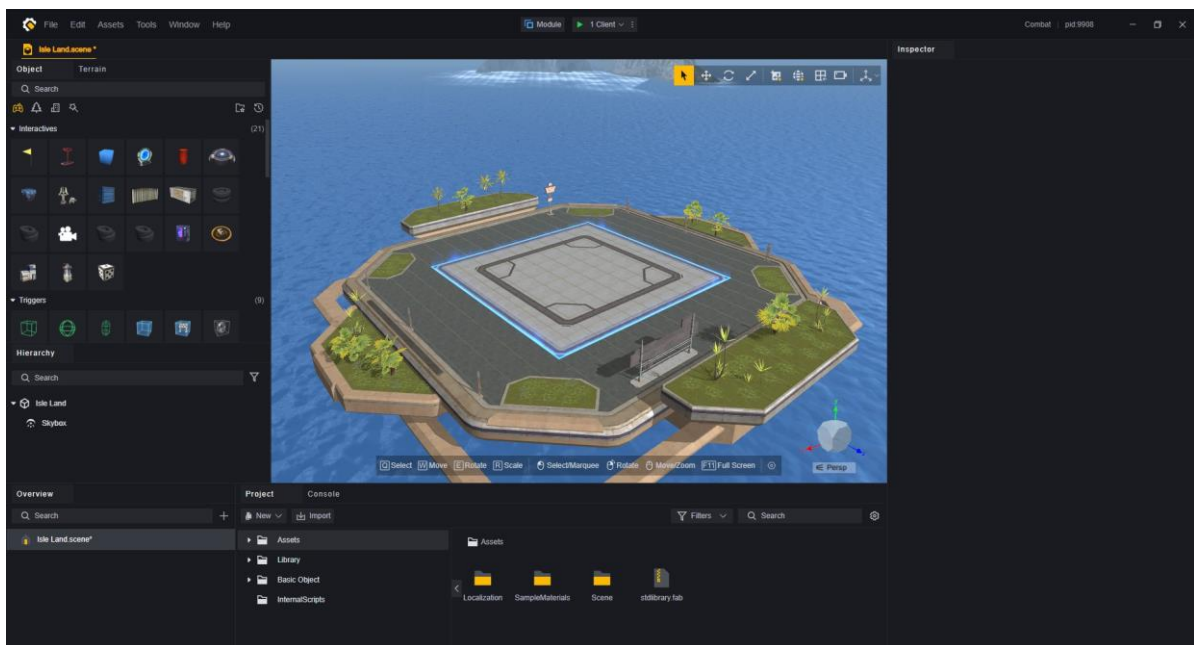




How to make a combat example

Below is a simple example of how to create battle content.

1. Create Project

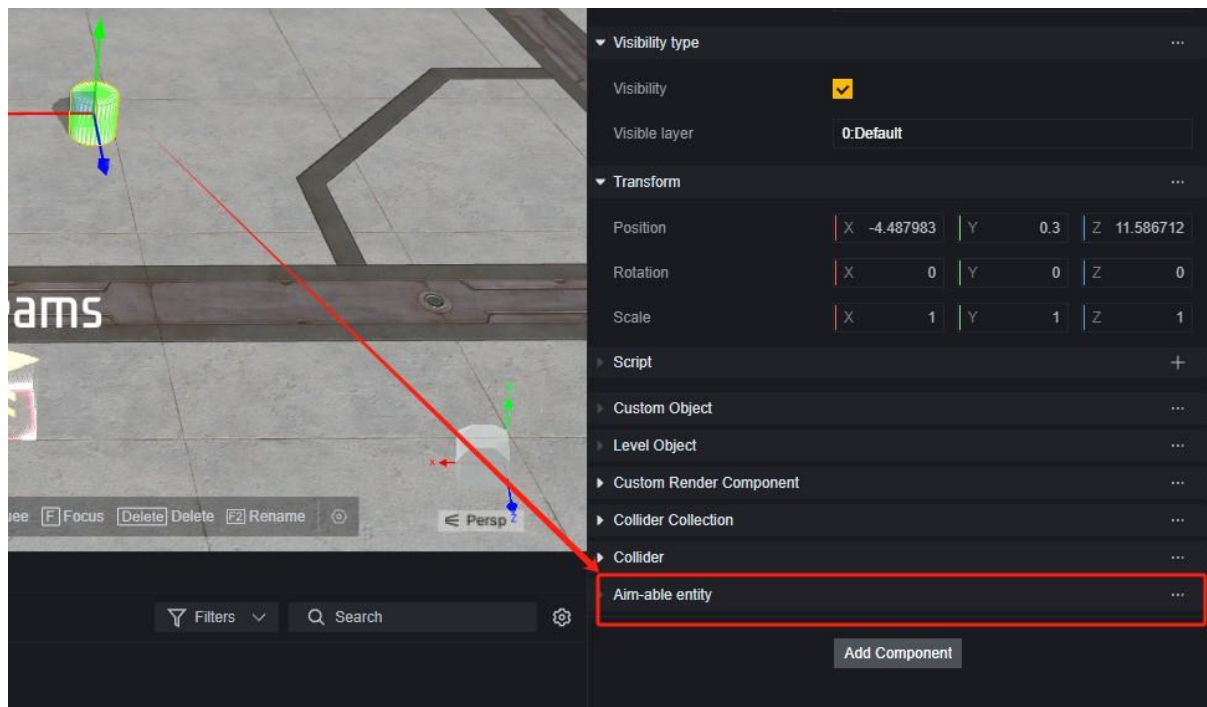


2. Setting up scene objects

We need a birth point to specify the player's birth location, some zombies, and some custom combat objects.



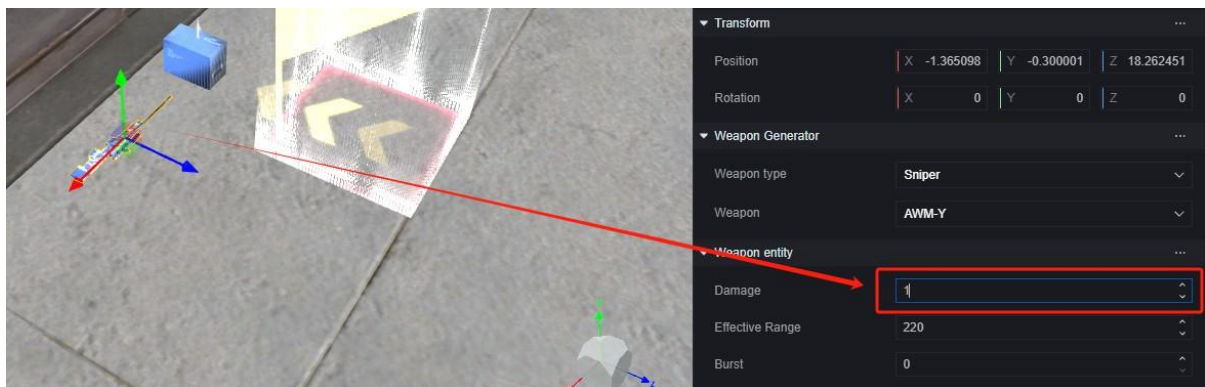
Add targetable solid components to custom combat objects.



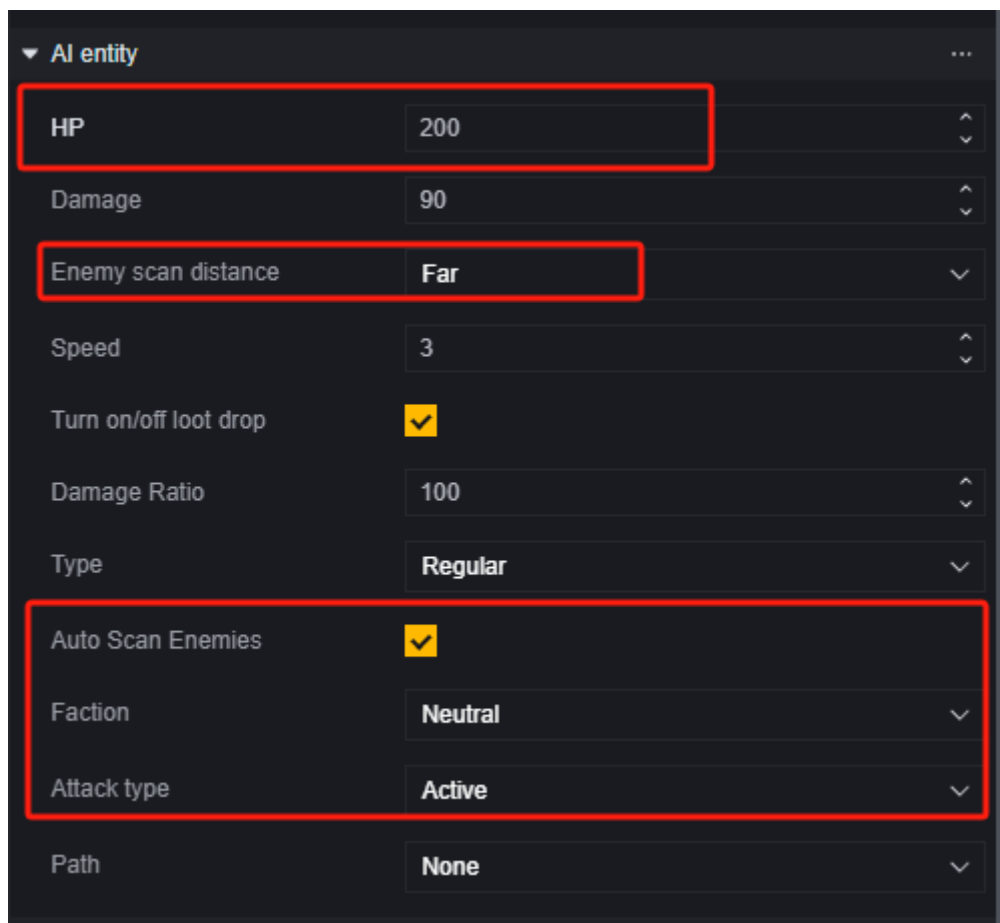
Using a weapons and props unit, set up a gun and ammo in front of the player's birth point.



Adjust the gun's properties so that it can only deal 1 point of damage per shot.



Adjust the zombie's attributes so that its camp is neutral, it has a long tracking distance, and it has as much as 200 life points.



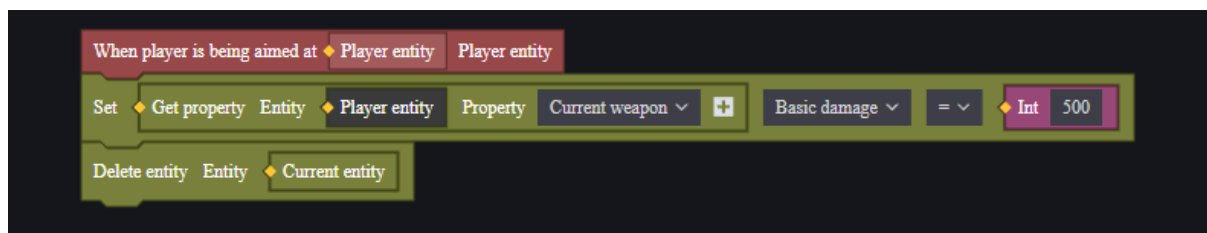
Neutral camp zombies will actively attack players in range

At this point you should have noticed that there is no way for the player to destroy the zombies via firearms, we are now going to add some features to the custom combat entity: cones: destroys itself after adjusting the damage of the player's current primary weapon to 500 when targeted by the player.

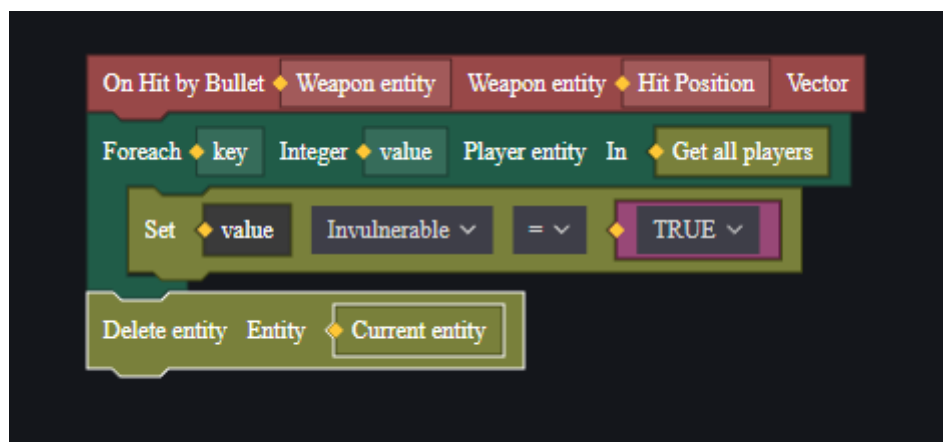
Cube: Gives all players invincibility and destroys itself after being attacked by a bullet.

Column: destroys itself after dealing 500 Oil Drum Blast type damage to all **combatale** entities when attacked.

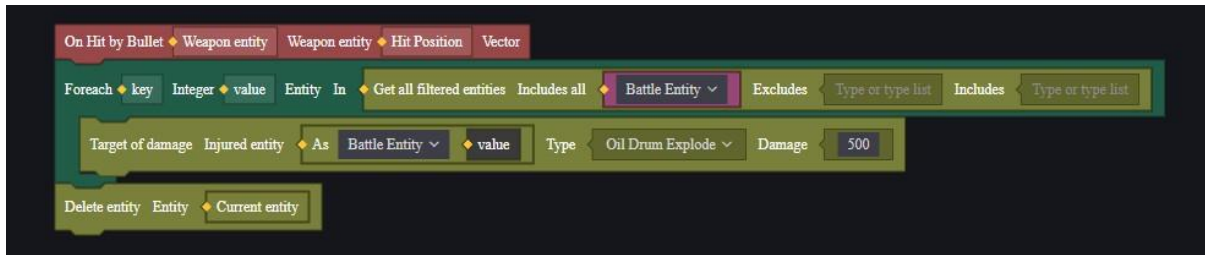
Cone Script:



Cube Script:



Cylinder Script:



Attack the zombies directly:



Practical approach 1:

Gain a damage boost by aiming at the cone and kill all zombies.



The cone has been destroyed by aiming.





Practical approach 2:

Gain invincibility by attacking the cube and destroy all zombies by attacking the column.



Gain invincibility after destroying the cube.

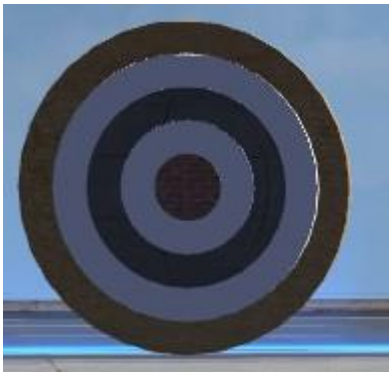


Destroy all the zombies after destroying the columns.

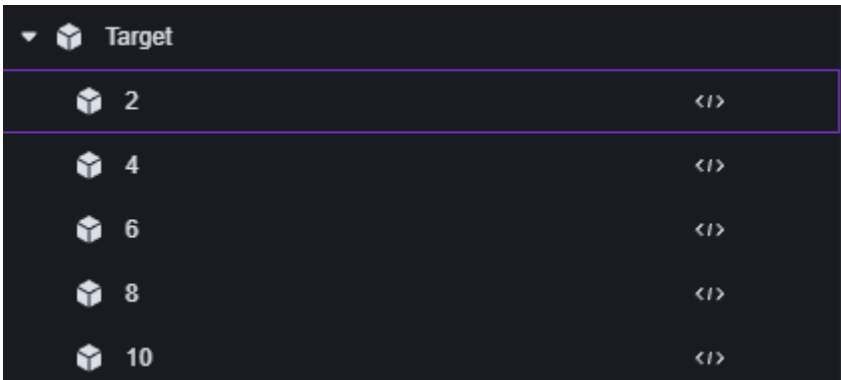
Expanded Example

easy target

Make targets in the scene.

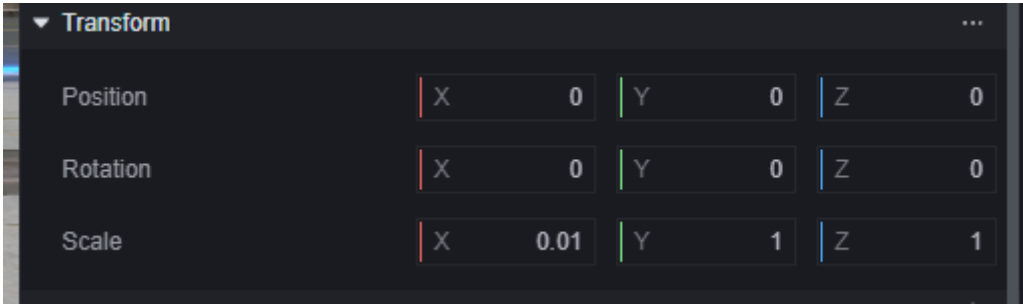


Create a parent object and drag in five basic spheres below it, representing 2, 4, 6, 8, and 10 rings.

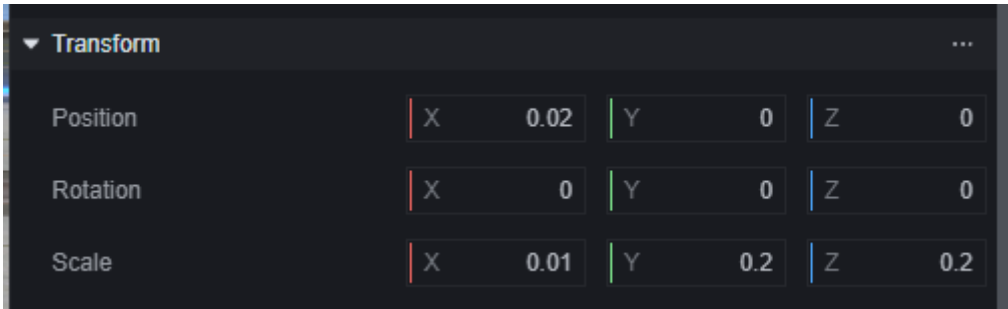


The sphere scaling was adjusted to X = 0.01, and both Y and Z were 1, 0.8, 0.6, 0.4, and 0.2 in that order. This yielded five discs for detecting the number of rings struck by the bullet.

And adjust the x-value of the position so that the larger ring count is further forward.

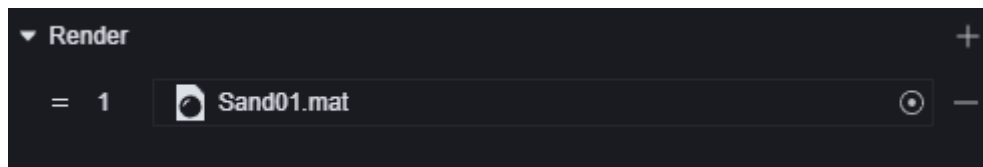


2 Transform of the Universal Body

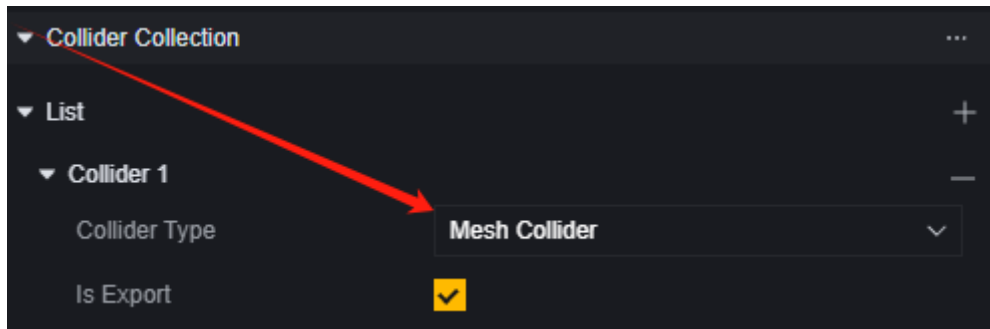


10 Transform of the Universal Body

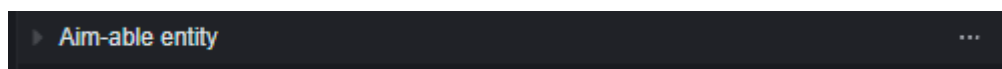
Modify the sphere rendering so that two adjacent rings behave differently.



Modify the collision type of all sub-objects to Mesh, so that the collision fits the shape of the object tightly.



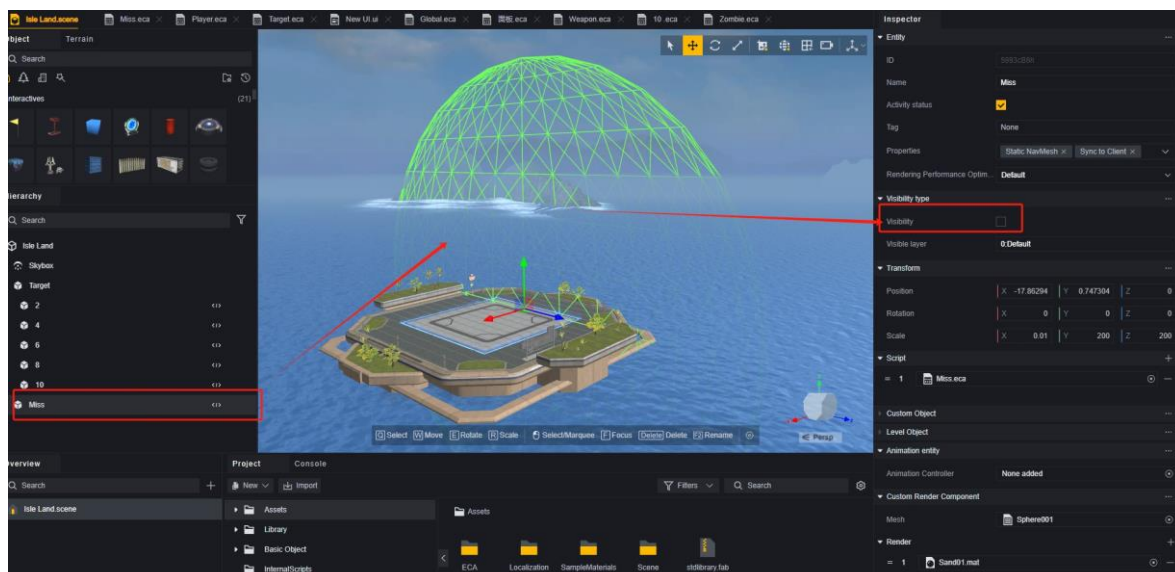
Add targetable entity components to all sub-objects.

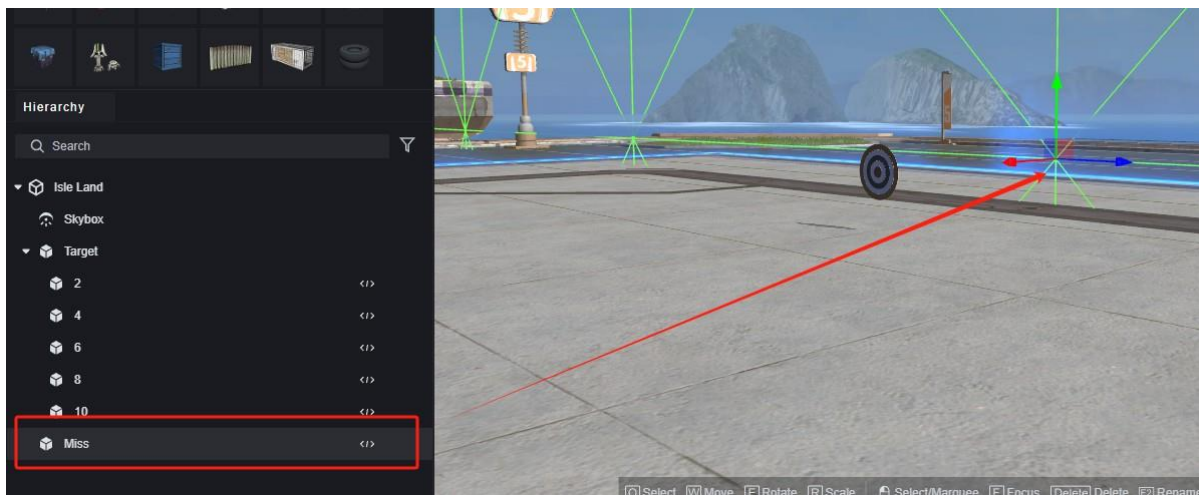


This gives us a target object, where the larger the ring number the more forward the collision detection, and the bullet hit will preferentially detect the colliding body with the largest ring number at the current position.

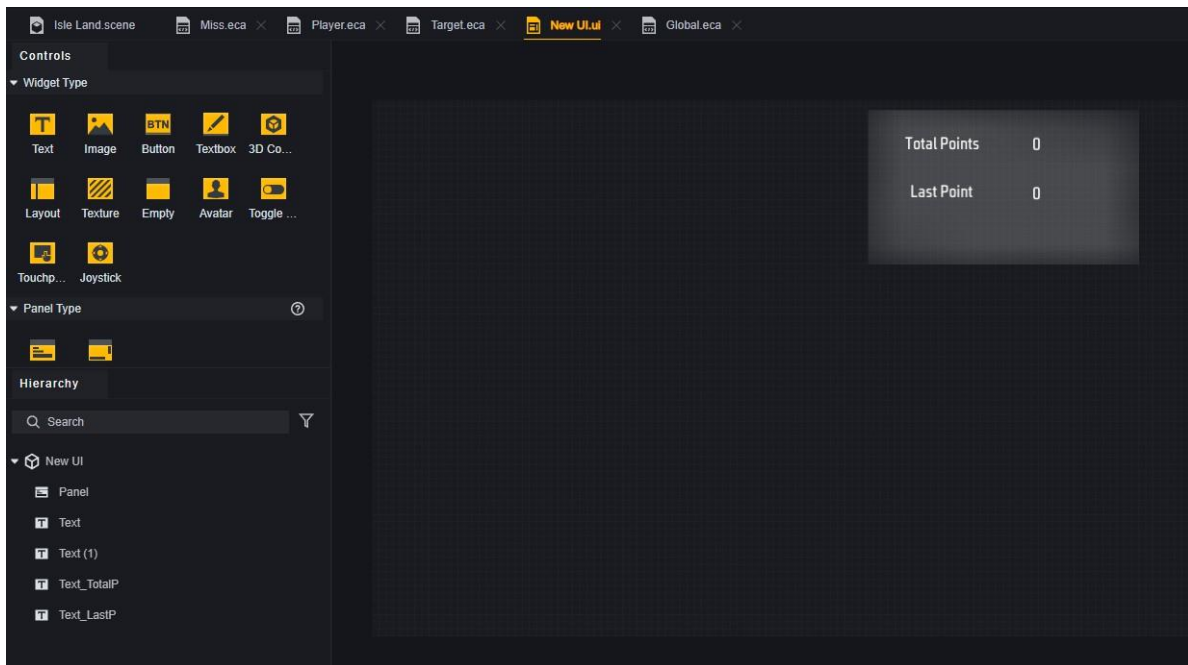


A large background is placed behind the target, turning off its visibility and keeping only collision and targetable entities as off-target detection.

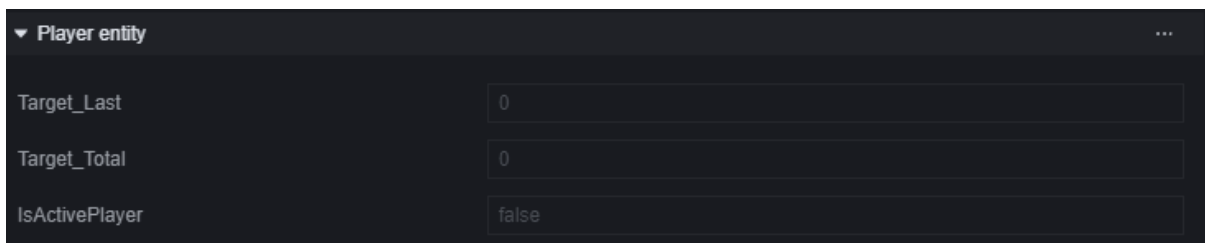




Create a UI that will indicate the current player's score.

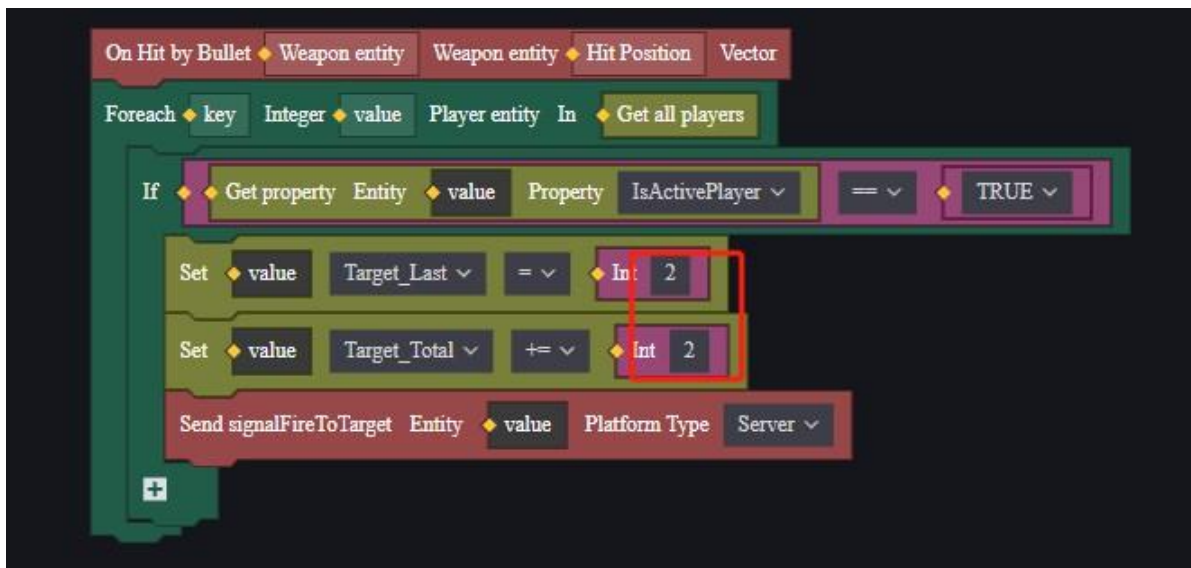


Add three custom attributes to the player: current target score, total target score, and whether or not they are an active player.

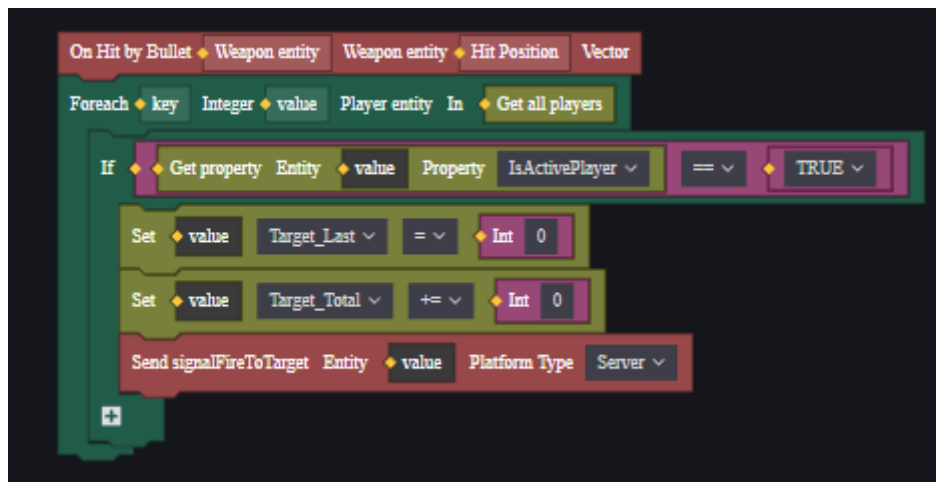


Whether or not it is an active player is used to determine which player is the one who is hitting the target when there is more than one player; target shooting by other non-active players will not be counted in the score.

Add scripts for all target objects and objects used to determine off-target, which have a similar script structure, differing only in the value of the Modify Player's Target Points attribute:



Scripts for 2-ring subobjects



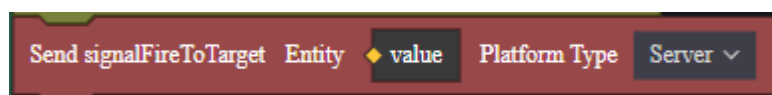
Scripts for off-target backgrounds

Add scripts for global modules, set to activate players when they join a match, and create custom UI.



The actual application can be modified to activate the player according to the conditions.

An event is sent to the player when the target or background is hit by a bullet.



Add a script for the player to modify the text on its own custom UI when it receives a target hit event.



Place the gun in the scene and enter the game to test it.

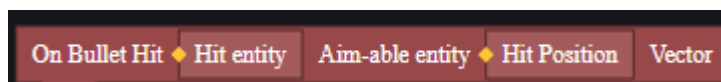


Under this example, off-targeting due to attacking the ground is not handled.

explosive bomb

Here's how to make a gun that fires AOE damage bullets and has an explosion effect.

Firstly, to analyse this, the damage inflicted and the hit effect should all occur when the bullet actually hits the target, so the When Bullet Hits event is chosen.



This event relies on the weapon entity, so the script should be mounted on the weapon. Add an explosion effect on hit

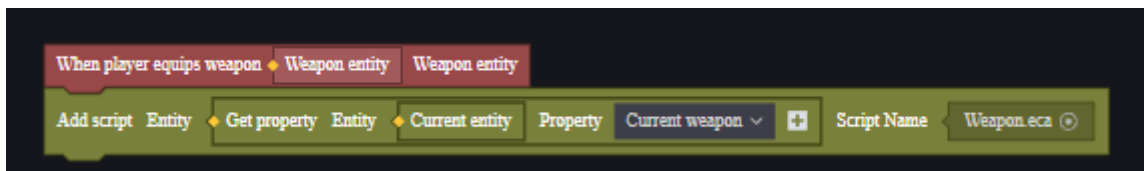


For every player, this effect is visible to add
AOE damage

Only AI zombies and players will be harmed here.



This way the weapon script is created and mounted on the corresponding weapon.



Here is the player script, any player equips a weapon and that weapon gains
an AOE ability. Add some AI zombies to the scenario and test it with multiple
ends.

