Welcome

The **Combat for Game Creator** module is now available for purchase on the Unity Asset store. Visit the asset's product page for details.

The module introduces combat features to the Game Creator ecosystem. It provides optional integrations for both the Shooter and Melee modules. It has also been designed to work with the Behavior module and the Accessibility Module's Mobile UI controls (by Pivec Labs).

> **Unity 2019.4 LTS (or higher) is required beginning with release v0.8.0 (previous versions required 2019.3 or newer).**

Combat Demos: Fire Chicken Games YouTube Channel

Combat (and other) Tutorials: RVR YouTube Channel

Discord: Game Creator Server's #Combat Channel

Getting Started

FIRE CHICKEN GAMES

COMBAT

The **Combat** module adds enhanced, game-ready, combat features:

- Targeting
  - Proximity-based Targeting (with programmatic API, actions, and triggers).
  - Point-and-Click Mouse Targeting.
  - AI Targeting (requires the Behavior module).
  - Mobile UI Button Compatibility for Accessibility Module (by Pivec Labs).
  - Customizable Target Indicator.
  - Targeting by Visibility.
- Spawn System (with optional weighted random selection).
- Weapon Stashes (weapon carrying/switching mechanic).
- Melee and Shooter Integration (requires Melee and/or Shooter modules).
- Homing Projectile.
- Support for Destructible Targets.

Dependencies

Combat is an extension for **Game Creator**. It is required - **Combat** will not work without it.

They can be purchased from the Unity Asset Store:

- Game Creator

The **Melee** and **Shooter** modules are optional dependencies. Get them here:

- Melee
- Shooter

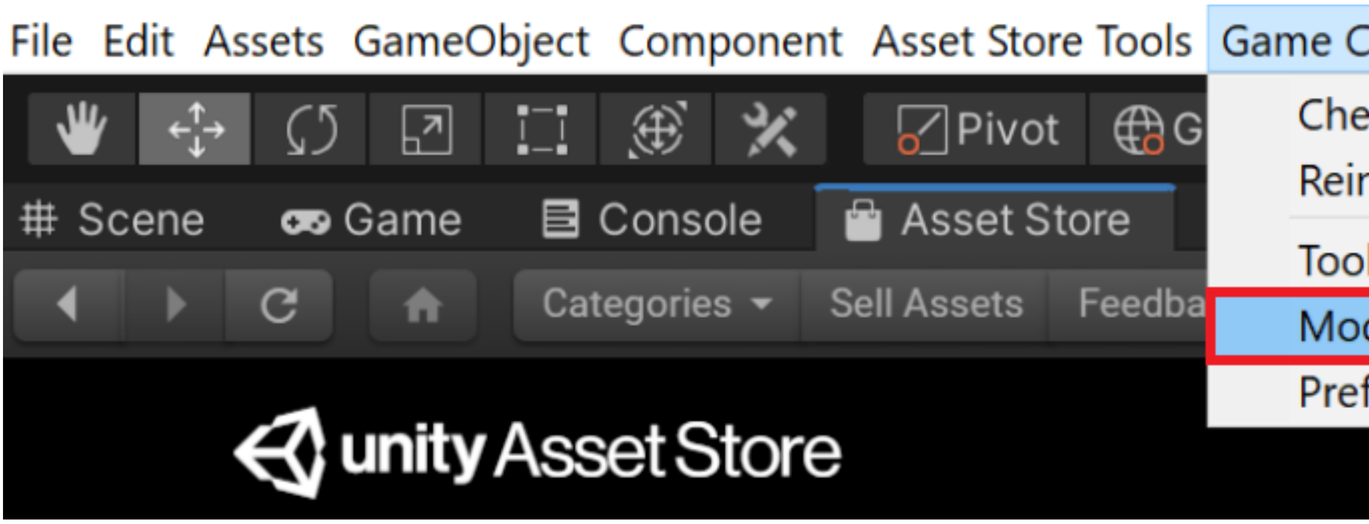
What's Included

- Full source code.
- An examples module that contains scenes that demonstrate the features listed above.

Module Installation

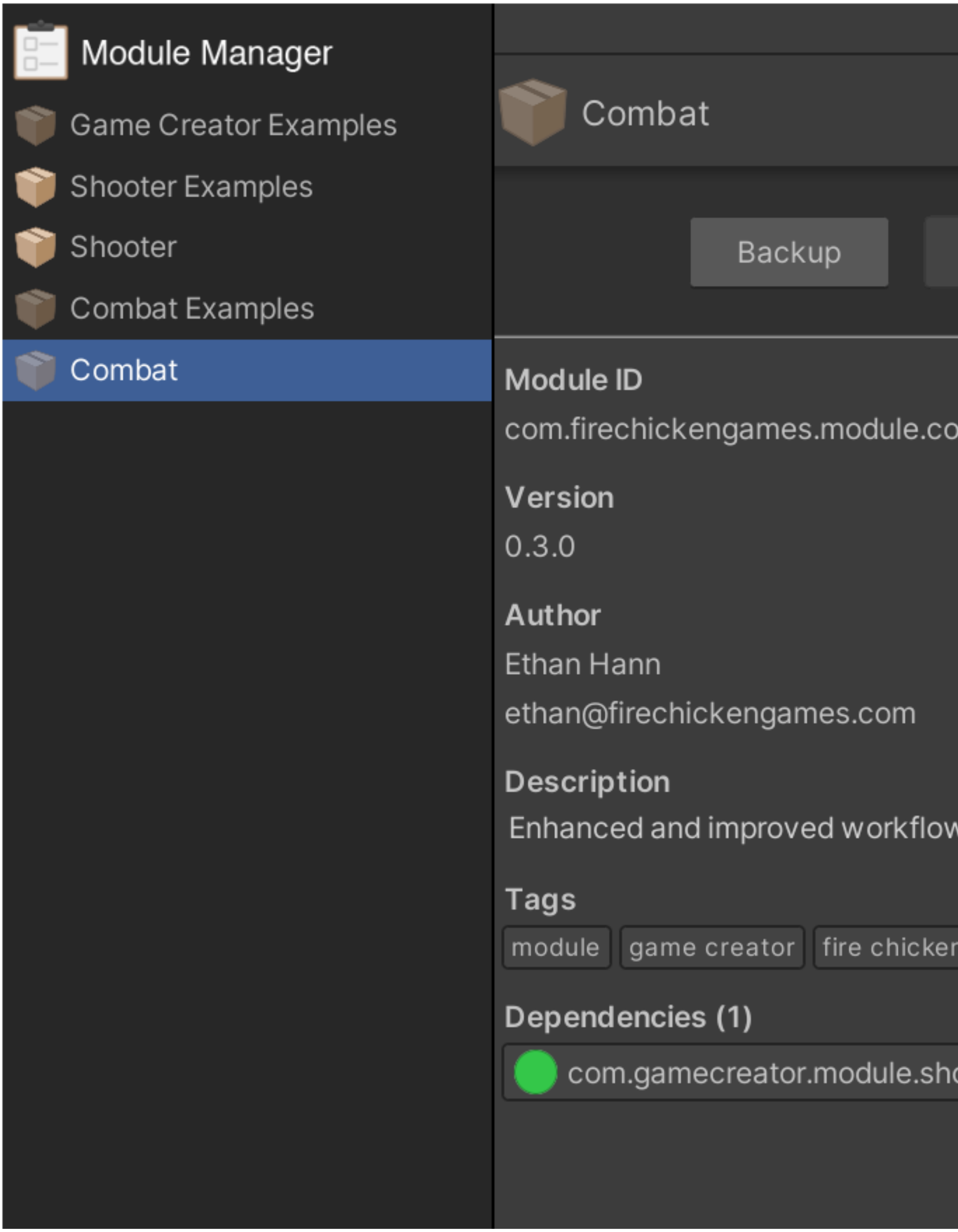
After purchasing and downloading the **Combat** module, it must be enabled with the Game Creator Module Manager.

Step 1: Open the Module Manager



Step 2: Enable the Combat Module

Module Manager



Module ID

com.firechickengames.module.co

Version

0.3.0

Author

Ethan Hann

ethan@firechickengames.com

Description

Enhanced and improved workflow

Tags

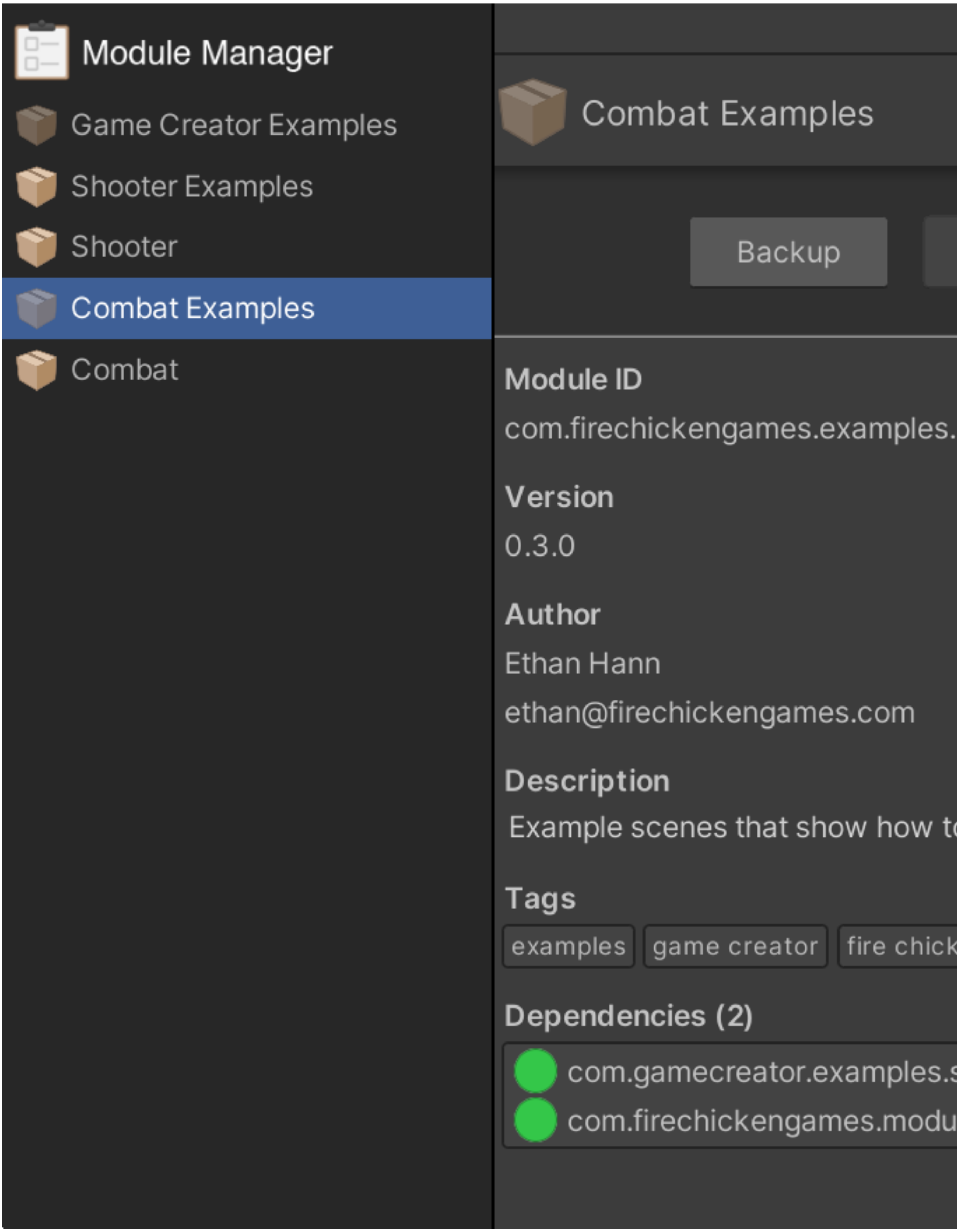
module | game creator | fire chicken

Dependencies (1)

● com.gamecreator.module.sho

Step 3 (optional): Install the Combat Examples Module

Module Manager



Module ID

com.firechickengames.examples.

Version

0.3.0

Author

Ethan Hann

ethan@firechickengames.com

Description

Example scenes that show how t

Tags

examples game creator fire chick

Dependencies (2)

● com.gamecreator.examples.s

● com.firechickengames.modu

Melee and Shooter Module Integrations

The **Combat** module provides a lightweight integration to allow seamless Shooter/Melee targeting and weapon swit

To enable the integration, simply enable the included **Combat (Melee)** and/or **Combat (Shooter)** integration module

Note that also included are the **Combat Examples (Melee)** and **Combat Examples (Shooter)** modules that demonst

Homing Projectile

As the name suggests, a **Homing Projectile** seeks its target even if the weapon firing the projectile is not pointed dir

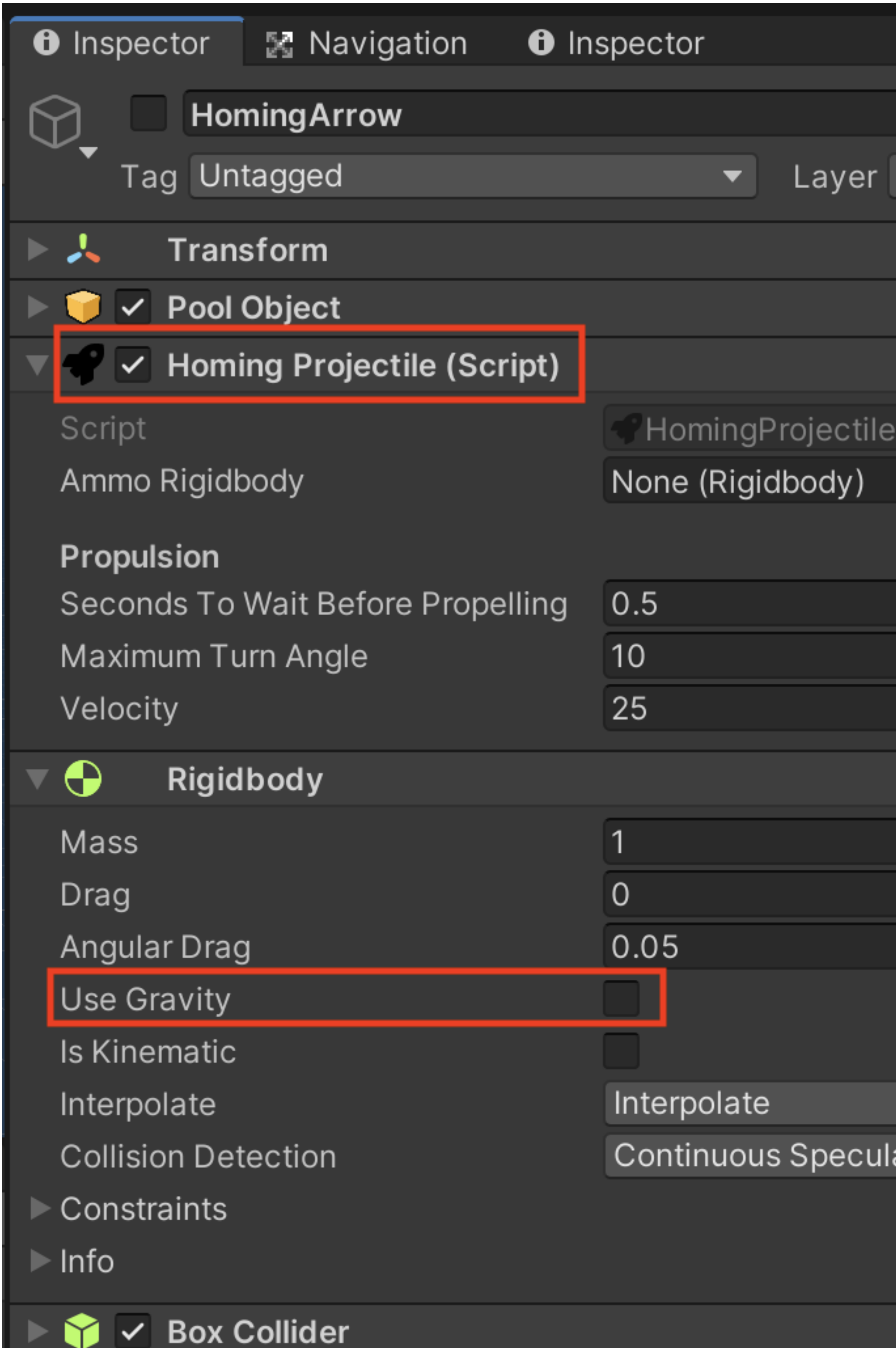
Setup is trivial. Simply attach the Combat module’s **Homing Projectile** component to any projectile. The component object contains a Rigidbody component.

The **Propulsion** settings control the movement behavior of the projectile:

- **Seconds To Wait Before Propelling**: Delays the propulsion of the projectile by a number of seconds. A value of
- **Maximum Turn Angle**: The maximum angle, in degrees, that the projectile will turn while homing in on its target.
- **Velocity**: How fast the projectile moves toward its target - this should likely match or exceed the max velocity of t setting will effectively override the projectile ammo’s min/max velocity.

Rigidybody Gravity

Turning off gravity on the Rigidbody is optional, but might be desired depending on the specific projectile.



□ HomingArrow

Tag Untagged ▼ Layer

▶ Transform

▶ ☑ Pool Object

▼ ☑ Homing Projectile (Script)

Script	HomingProjectile
Ammo Rigidbody	None (Rigidbody)

Propulsion

Seconds To Wait Before Propelling	0.5
Maximum Turn Angle	10
Velocity	25

▼ Rigidbody

Mass	1
Drag	0
Angular Drag	0.05
Use Gravity	□
Is Kinematic	□
Interpolate	Interpolate
Collision Detection	Continuous Specula

▶ Constraints

▶ Info

▶ ☑ Box Collider

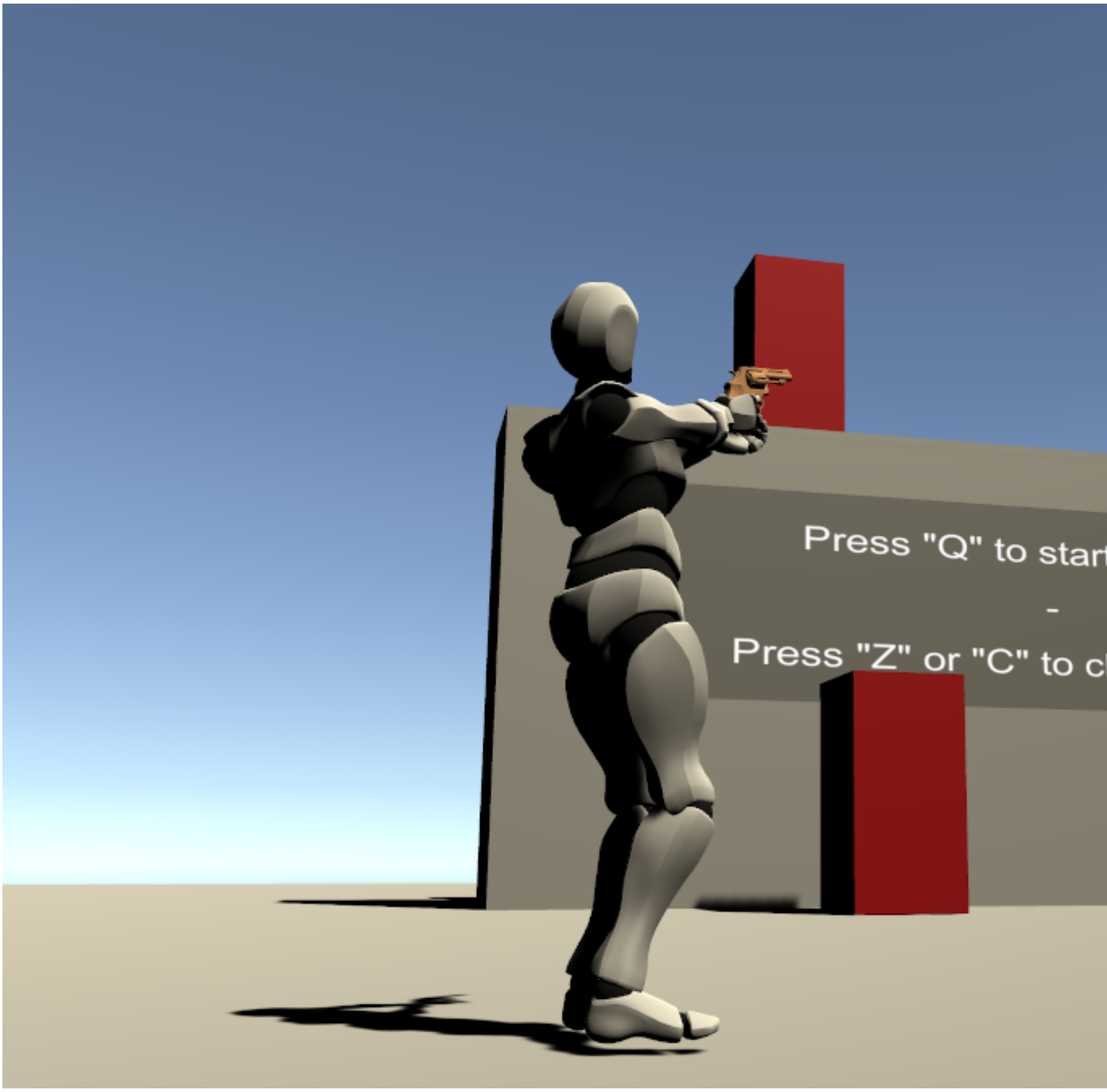
Overview

The **Combat** module provides a proximity-based targeting system. It allows a player to target characters (or other ga

Targeter and Targetable.

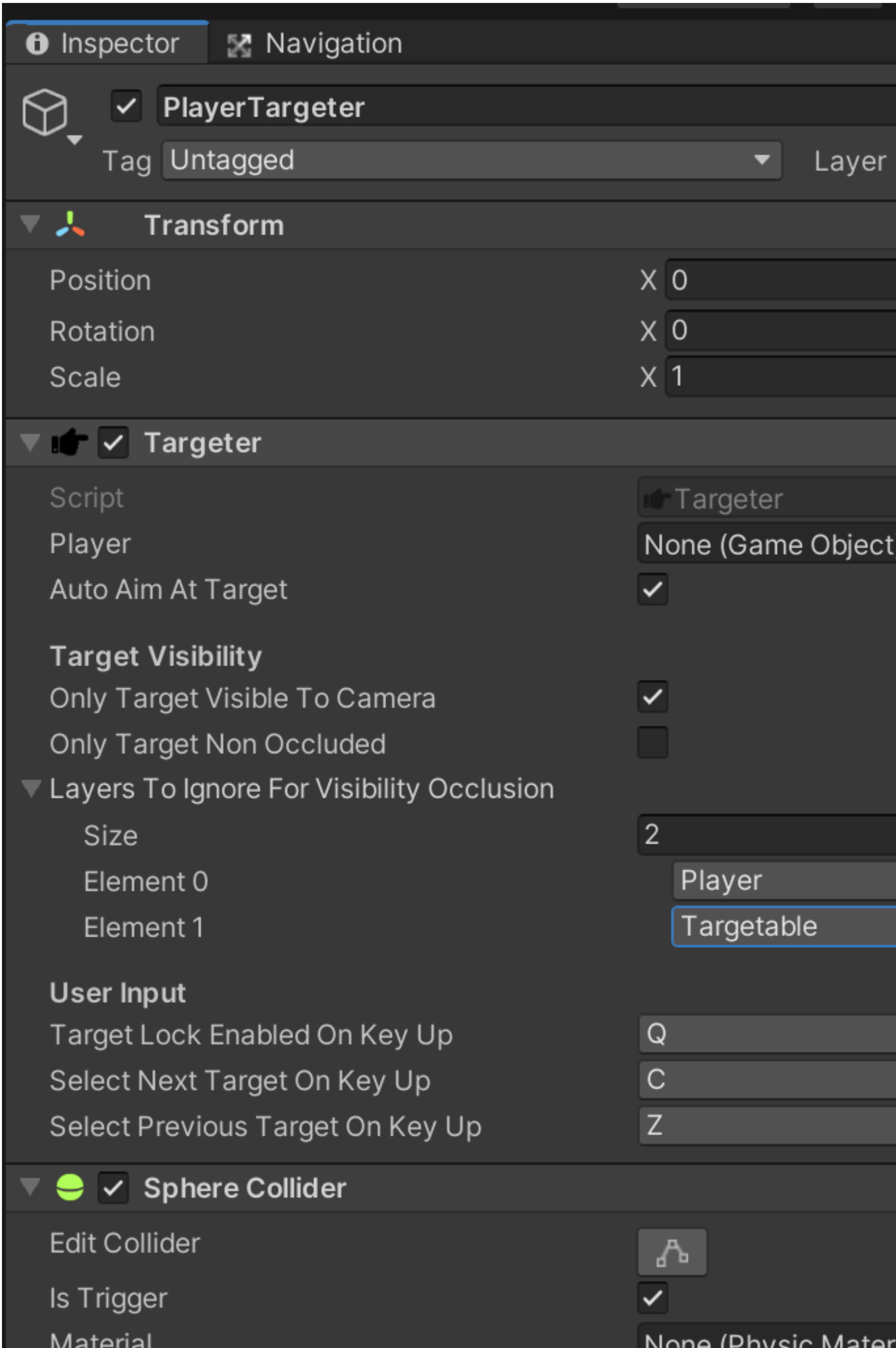
When a player with a **Targeter** component approaches objects with the **Targetable** component attached, the player c

used to switch targets. Please note that these keys can be remapped.



Targeter

Included in the **Combat Examples** module is a prefab called **PlayerTargeter** that demonstrates how to use the **Targe** depicted in the inspector screenshot below) is required. Note that the range of the **Targeter** is dictated by the **Radius** Game Creator Player object.



☑ PlayerTargeter

Tag Untagged

Layer

Transform

Position X 0

Rotation X 0

Scale X 1

☑ **Targeter**

Script Targeter

Player None (Game Object

Auto Aim At Target ☑

Target Visibility

Only Target Visible To Camera ☑

Only Target Non Occluded □

Layers To Ignore For Visibility Occlusion

Size 2

Element 0 Player

Element 1 Targetable

User Input

Target Lock Enabled On Key Up Q

Select Next Target On Key Up C

Select Previous Target On Key Up Z

☑ **Sphere Collider**

Edit Collider

Is Trigger ☑

Material None (Physic Mater

Auto Aim At Target

Automatically aiming at a target can be disabled with the **Auto Aim At Target** option - the character will still be locked

for some games.

Target Visibility

New in 0.4.0

The component's target visibility options allow for target selection to be limited by what the camera can see.

Only Target Visible To Camera

If enabled, only targets possibly visible to the camera (i.e. in its view frustum) are targetable.

Only Target Non-Occluded

If enabled, targets hidden behind objects are not targetable.

Note that this option is turned off by default because the player and targetables need to be on dedicated layers which

Layers To Ignore For Visibility Occlusion

The layers to ignore when determining target visibility when Only Target Non-Occluded is enabled.

Typically, there should be two layers:

- A "Targetable" layer that contains all targetable objects.
- A "Player" layer that contains the player.

User Input

The **User Input** section of the **Targeter** component allows the keys that control target locking and switching to be cu

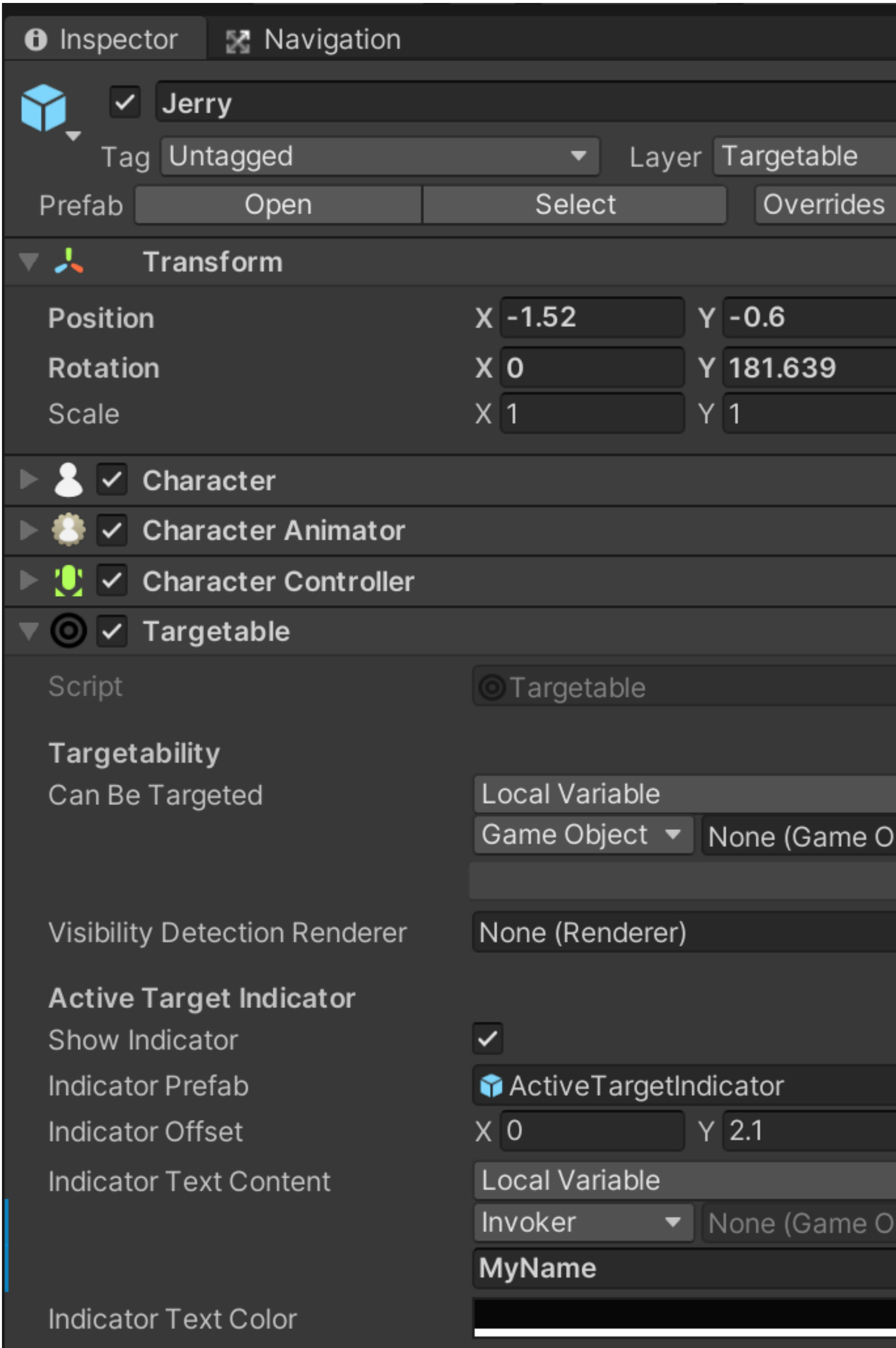
Targetable

Making game objects targetable using the **Combat** module’s **Targetable** component is trivial for Game Creator Char

Basic Setup

Characters

To make a character targetable, simply add the **Targetable** component to it.



Inspector Navigation

☑ Jerry

Tag Untagged Layer Targetable

Prefab Open Select Overrides

Transform

Position X -1.52 Y -0.6

Rotation X 0 Y 181.639

Scale X 1 Y 1

☑ Character

☑ Character Animator

☑ Character Controller

☑ Targetable

Script Targetable

Targetability

Can Be Targeted Local Variable

Game Object None (Game O

Visibility Detection Renderer None (Renderer)

Active Target Indicator

Show Indicator ☑

Indicator Prefab ActiveTargetIndicator

Indicator Offset X 0 Y 2.1

Indicator Text Content Local Variable

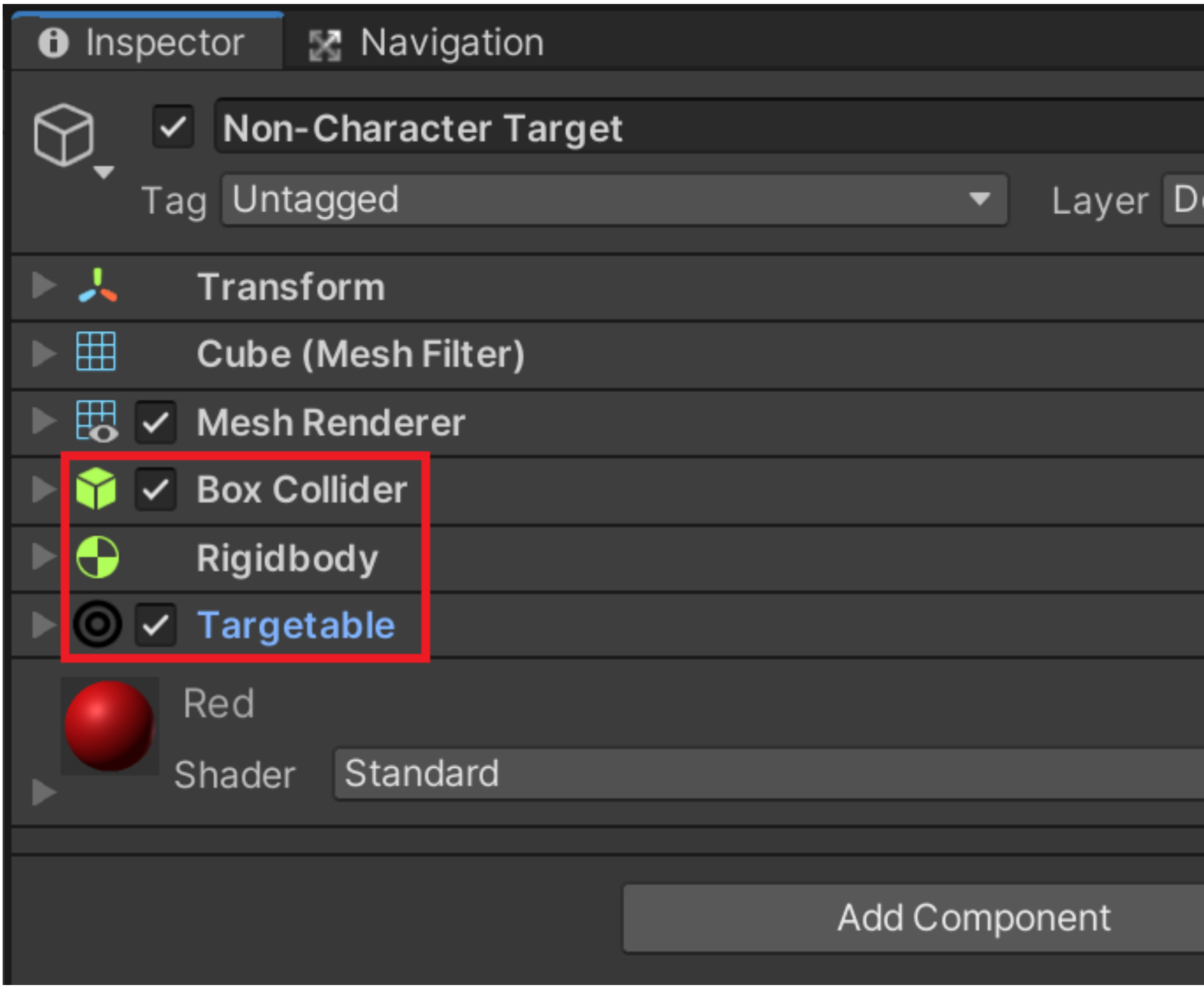
Invoker None (Game O

MyName

Indicator Text Color

Non-Character Game Objects

Any game object can be targetable if it has the **Targetable**, **Rigidbody**, and **Collider** components.



☑ Non-Character Target

Tag Untagged

Layer D

Transform

Cube (Mesh Filter)

☑ Mesh Renderer

☑ Box Collider

Rigidbody

☑ Targetable

Red

Shader Standard

Making a Target Untargetable

The **Targetable** component contains a boolean Game Creator Variable property, called **Can Be Targeted**, that can ma

For example, once the target has been defeated, it is likely desirable for the player to automatically stop targeting it ar

This is accomplished by:

1. Adding a boolean **Local Variable** to a character (e.g. "IsAlive").
2. Assigning the boolean variable to the **Can Be Targeted** property.
3. In the character's **On Receive Shot Actions**, set the boolean value to false - this will deselect the target and make

The **Combat Examples** module's **Example4-KillableCharacters** demo scene contains a pre-configured **KillableCha**

Visibility Detection Renderer

This property is used to determine if a **Targetable's** mesh is visible to a **Targeter** component. It does not normally need t the **Targetable** component's parent object does not have a mesh renderer in its hierarchy, or a mesh renderer other than

If visibility features are not used, this property can be ignored completely.

"On Target Become Untargetable" Trigger

New in 0.4.0

There is also a trigger called "On Target Become Untargetable" that allows for actions to be executed when the target **DestructibleTarget** prefab. It showcases how to use this trigger to implement destructible targets.

Advanced Options

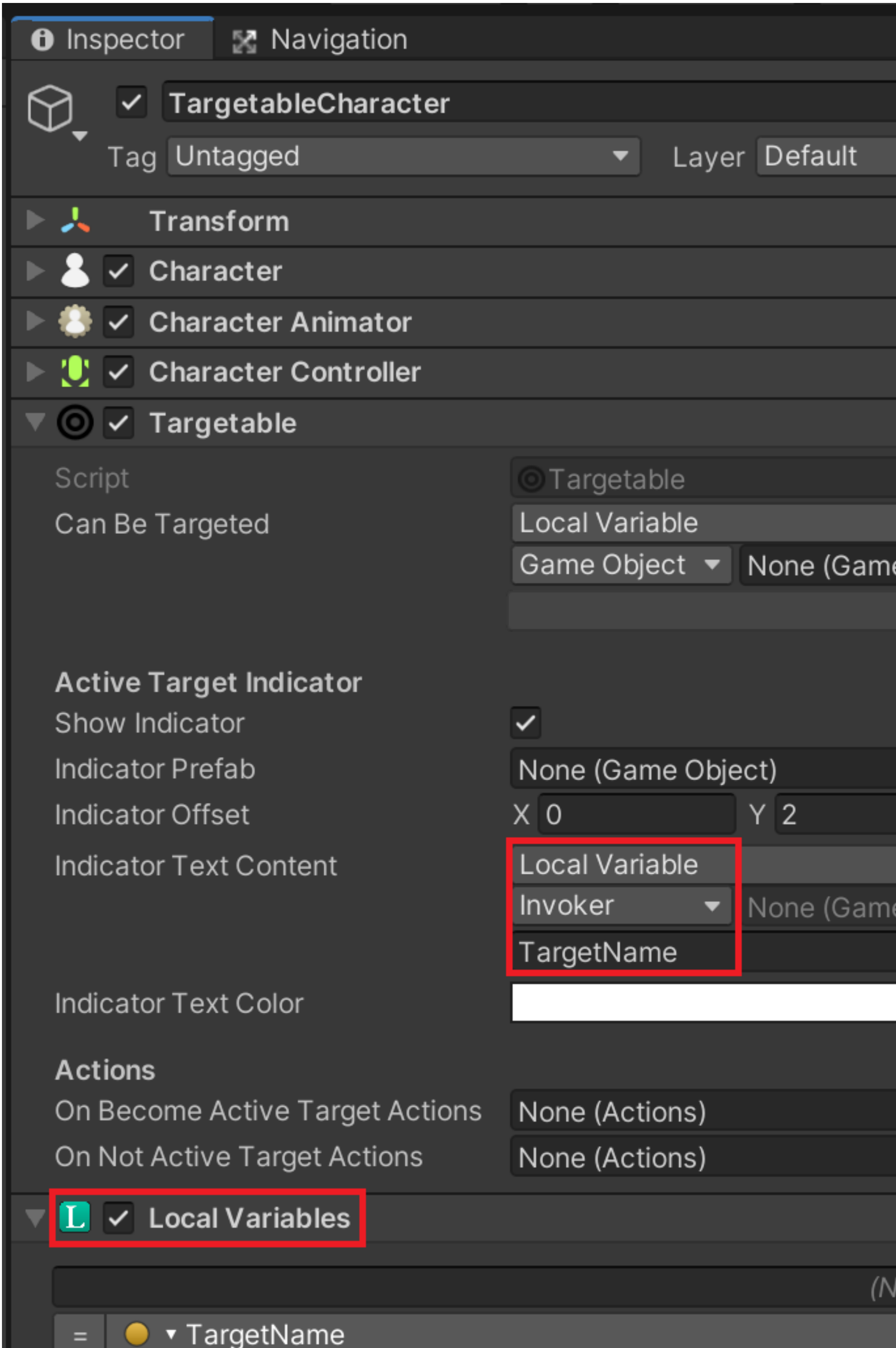
Active Target Indicator

The **Targetable** component provides an "indicator" feature that highlights the currently targeted game object. The cor

Jerry

Text

An indicator can have custom text, defined via a Game Creator **Global/Local Variable**. Practically speaking, it almos
Local Variable would then be configured on the instance of the prefab when used in a scene.



☑ TargetableCharacter

Tag Untagged ▾ Layer Default

Transform

☑ Character

☑ Character Animator

☑ Character Controller

☑ Targetable

Script Targetable

Can Be Targeted Local Variable

Game Object ▾ None (Game

Active Target Indicator

Show Indicator ☑

Indicator Prefab None (Game Object)

Indicator Offset X 0 Y 2

Indicator Text Content Local Variable

Invoker ▾ None (Game

TargetName

Indicator Text Color

Actions

On Become Active Target Actions None (Actions)

On Not Active Target Actions None (Actions)

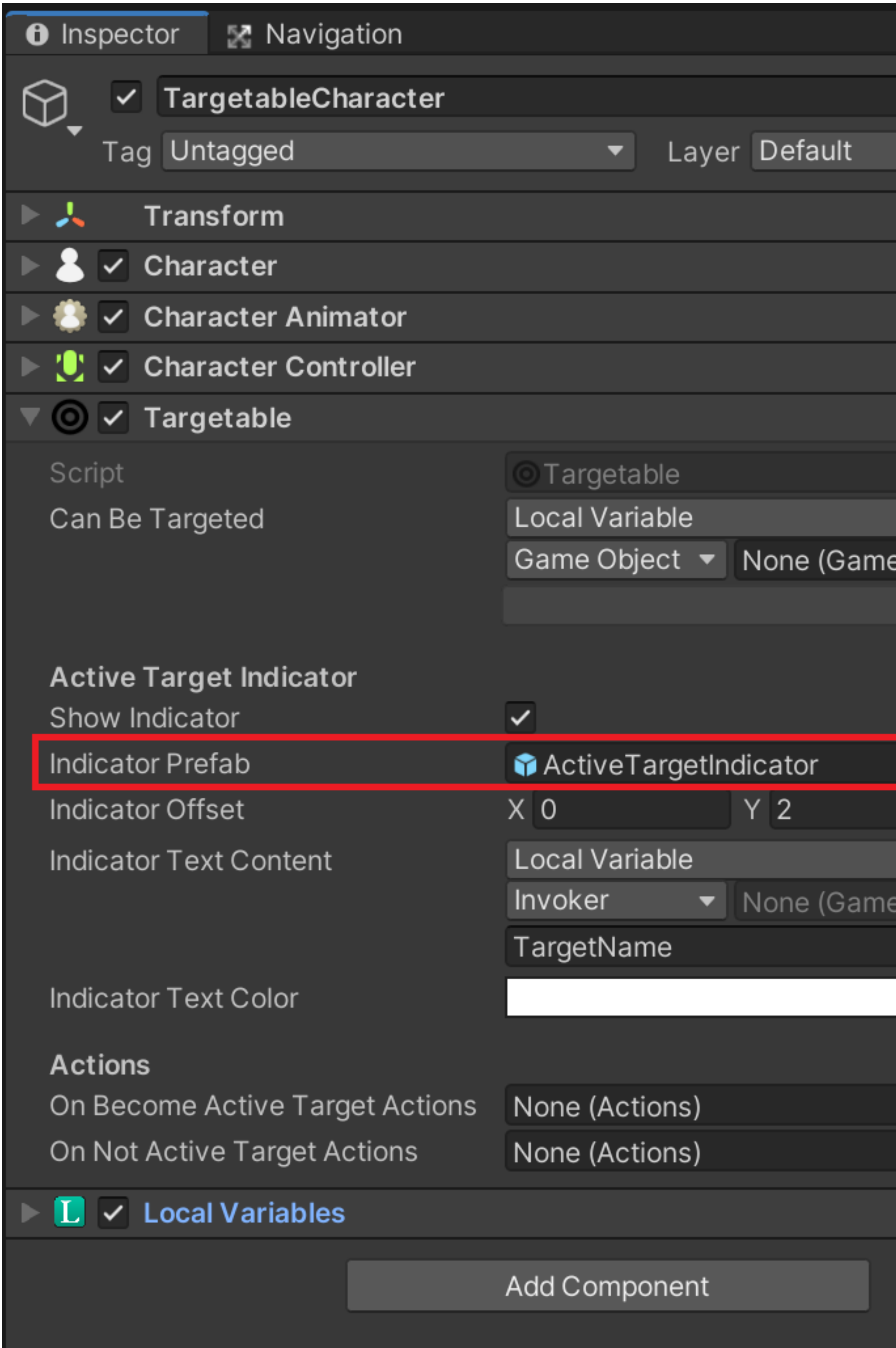
☑ Local Variables

(N

= ▾ TargetName

Prefab

If not set, the Game Creator **Floating Message** prefab is automatically set as the target indicator prefab at runtime. Th



☑ TargetableCharacter

Tag Untagged

Layer Default

Transform

☑ Character

☑ Character Animator

☑ Character Controller

☑ Targetable

Script Targetable

Can Be Targeted Local Variable
Game Object None (Game

Active Target Indicator

Show Indicator ☑

Indicator Prefab ActiveTargetIndicator

Indicator Offset X 0 Y 2

Indicator Text Content Local Variable
Invoker None (Game
TargetName

Indicator Text Color

Actions

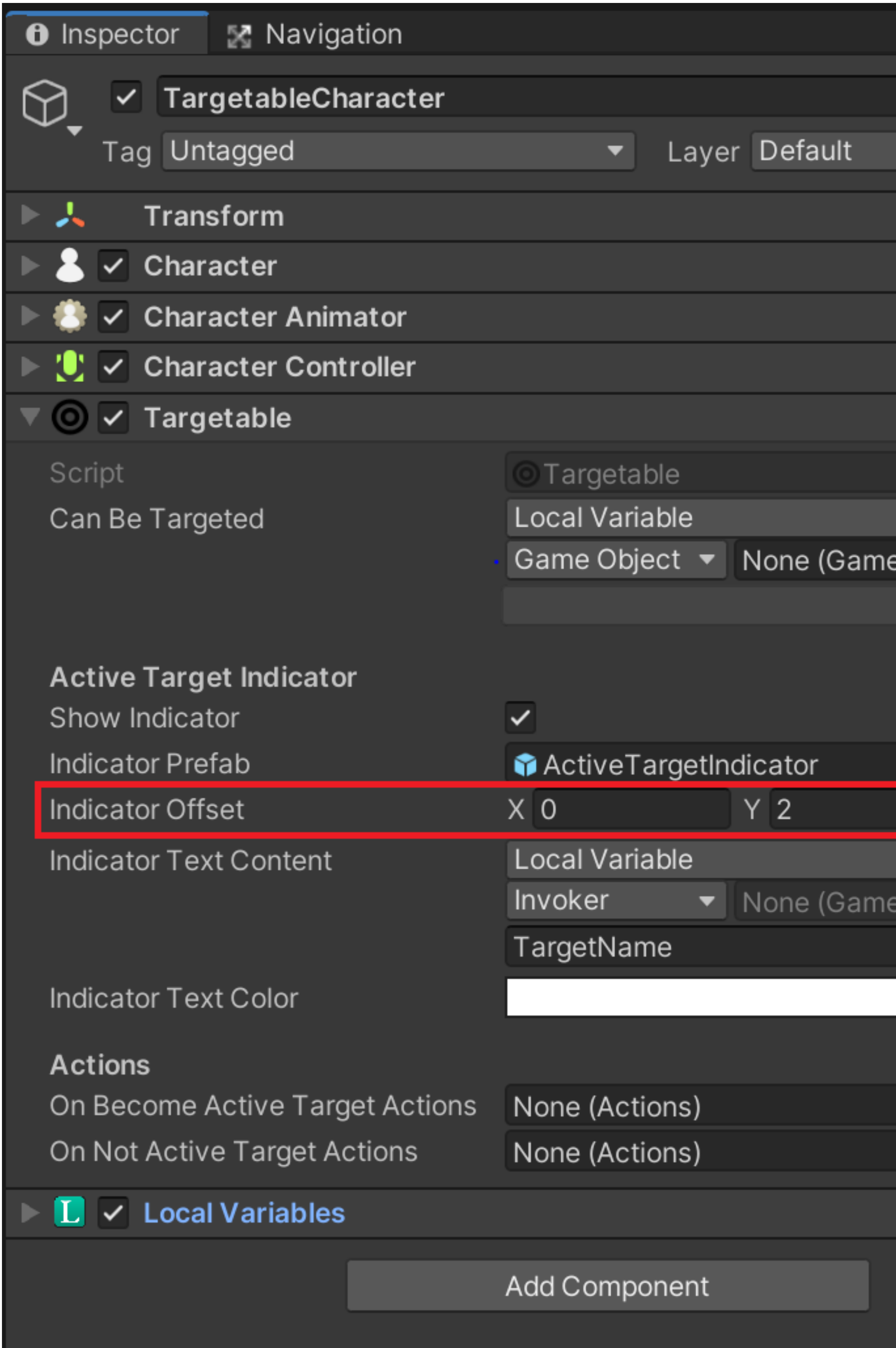
On Become Active Target Actions None (Actions)

On Not Active Target Actions None (Actions)

☑ Local Variables

Positioning

The target indicator is positioned relative to the parent game object. By default, the **Indicator Offset** vector will positi



Inspector | Navigation

☑ TargetableCharacter

Tag: Untagged | Layer: Default

- Transform
- ☑ Character
- ☑ Character Animator
- ☑ Character Controller
- ☑ Targetable

Script: Targetable

Can Be Targeted: Local Variable

Game Object | None (Game

Active Target Indicator

Show Indicator: ☑

Indicator Prefab: ActiveTargetIndicator

Indicator Offset: X 0 | Y 2

Indicator Text Content: Local Variable

Invoker | None (Game

TargetName

Indicator Text Color

Actions

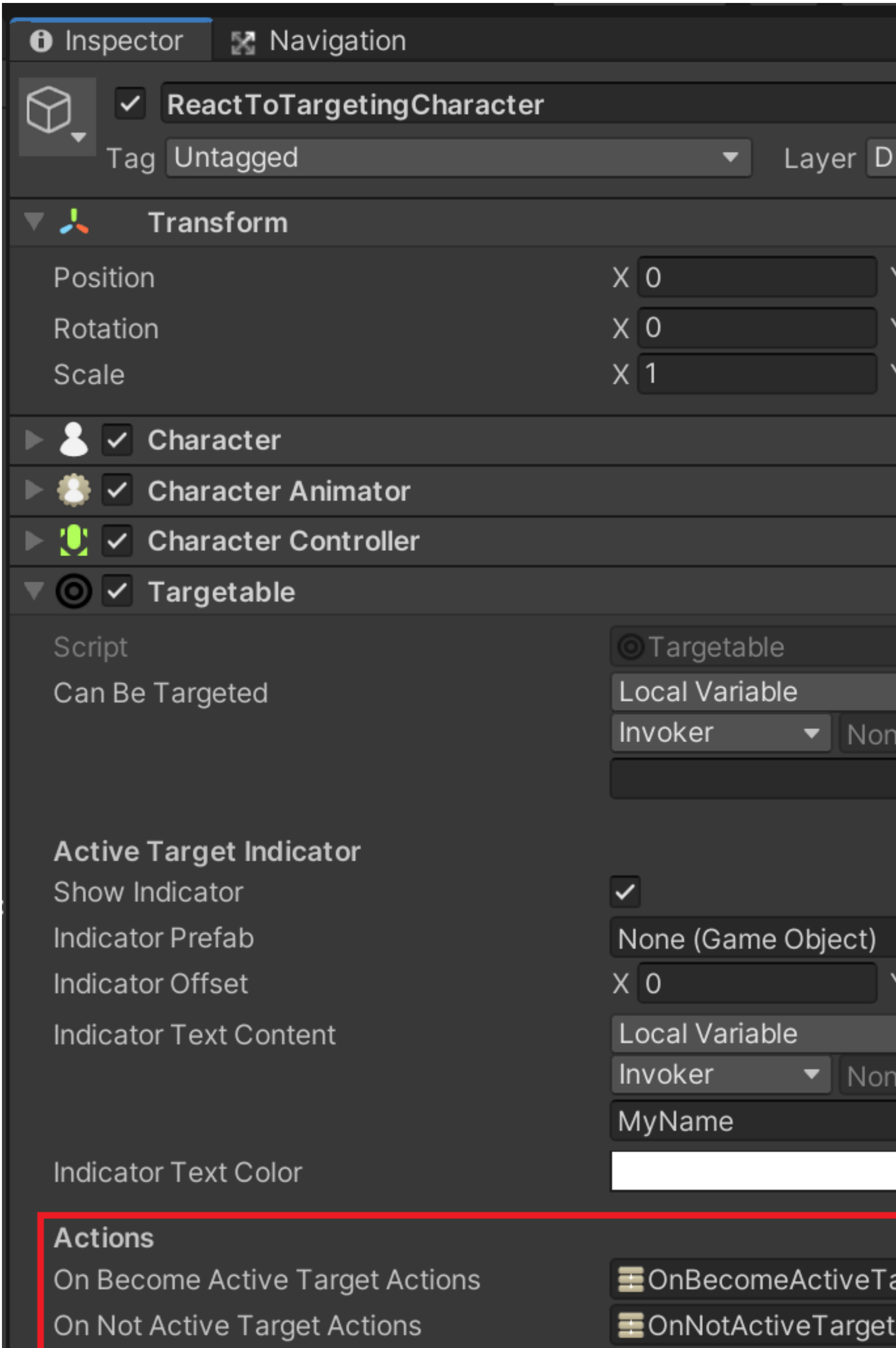
On Become Active Target Actions: None (Actions)

On Not Active Target Actions: None (Actions)

- ☑ Local Variables

Targeting Actions

A **Targetable** game object can optionally execute actions when it becomes the active target, and another set of actio



☑ ReactToTargetingCharacter

Tag Untagged

Layer D

Transform

Position X 0

Rotation X 0

Scale X 1

☑ Character

☑ Character Animator

☑ Character Controller

☑ Targetable

Script Targetable

Can Be Targeted Local Variable

Invoker Non

Active Target Indicator

Show Indicator ☑

Indicator Prefab None (Game Object)

Indicator Offset X 0

Indicator Text Content Local Variable

Invoker Non

MyName

Indicator Text Color

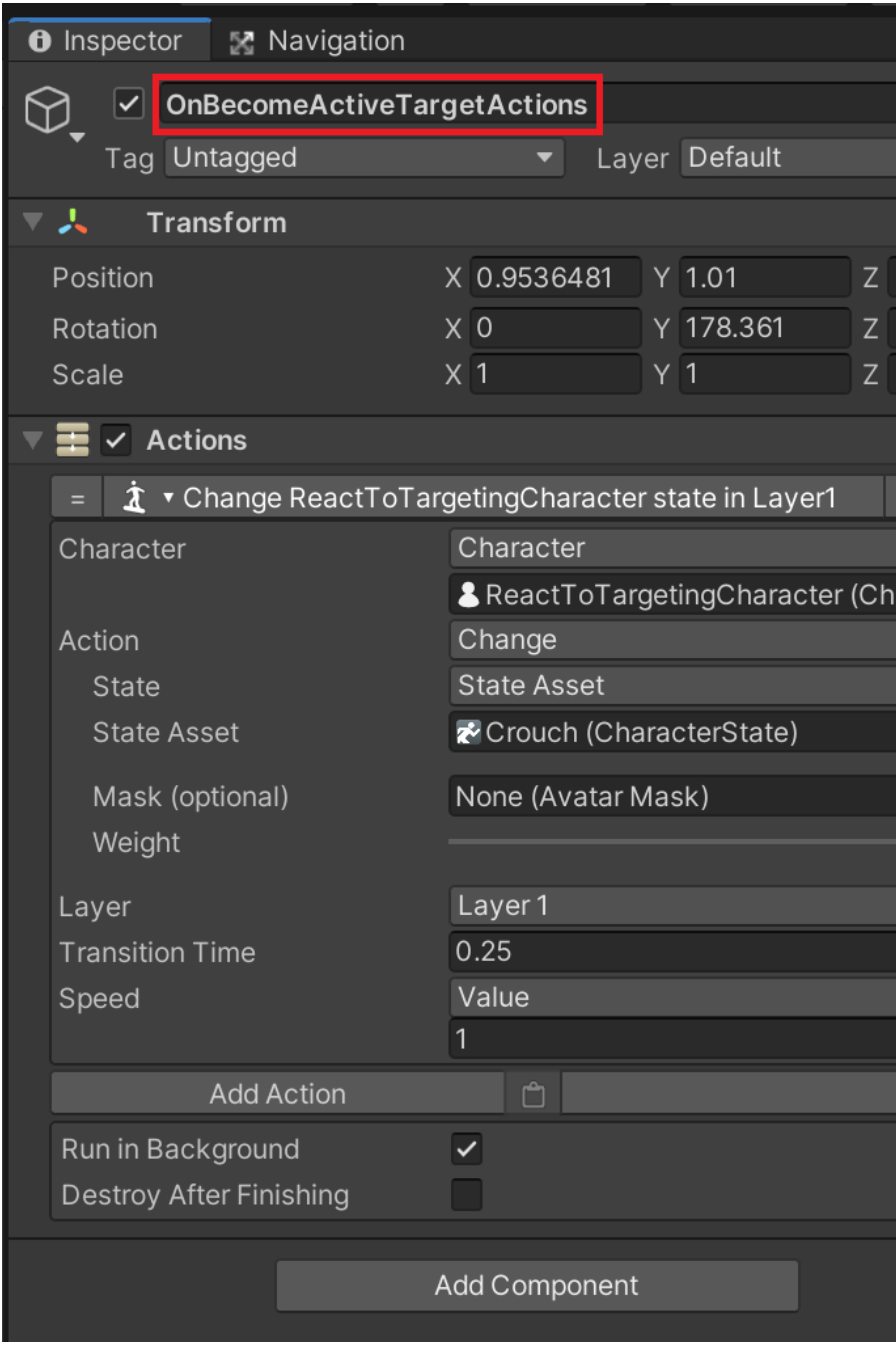
Actions

On Become Active Target Actions OnBecomeActiveTa

On Not Active Target Actions OnNotActiveTarget

"On Become Active Target"

When the target becomes the active target, these actions can (for example) make the target crouch. A more practical
flee).



☑ OnBecomeActiveTargetActions

Tag: Untagged
Layer: Default

Transform

	X	Y	Z
Position	0.9536481	1.01	
Rotation	0	178.361	
Scale	1	1	

☑ **Actions**

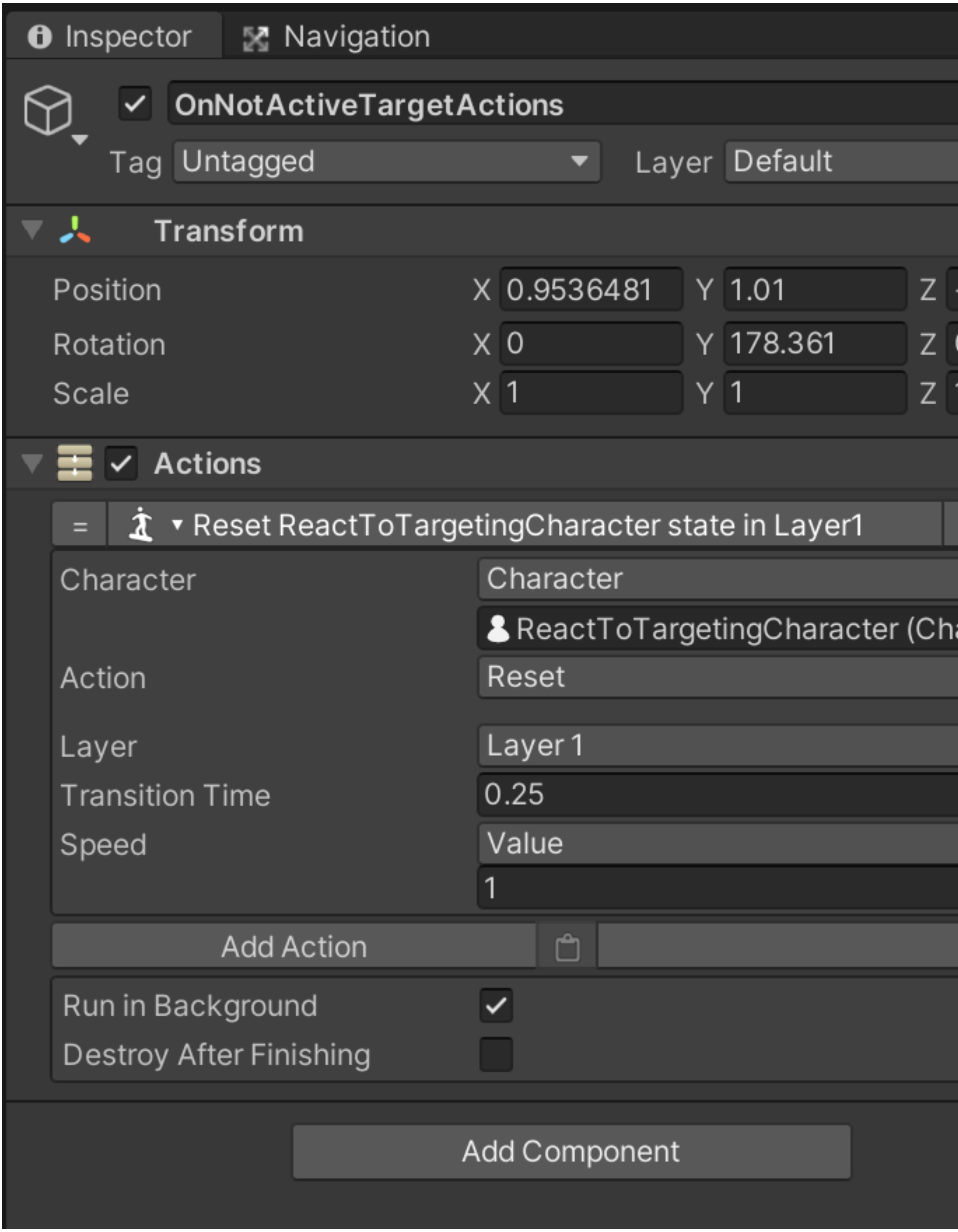
Change ReactToTargetingCharacter state in Layer1

Character	Character ReactToTargetingCharacter (Ch
Action	Change
State	State Asset
State Asset	Crouch (CharacterState)
Mask (optional)	None (Avatar Mask)
Weight	
Layer	Layer 1
Transition Time	0.25
Speed	Value 1

Run in Background ☑
Destroy After Finishing □

"On Not Active Target"

Related to the previous section, when the target is changed or targeting is disabled, this action will reset the target cha



☑ OnNotActiveTargetActions

Tag Untagged

Layer Default

Transform

Position	X	0.9536481	Y	1.01	Z
Rotation	X	0	Y	178.361	Z
Scale	X	1	Y	1	Z

☑ Actions

= Reset ReactToTargetingCharacter state in Layer1

Character: Character
ReactToTargetingCharacter (Ch

Action: Reset

Layer: Layer 1

Transition Time: 0.25

Speed: Value
1

Run in Background ☑

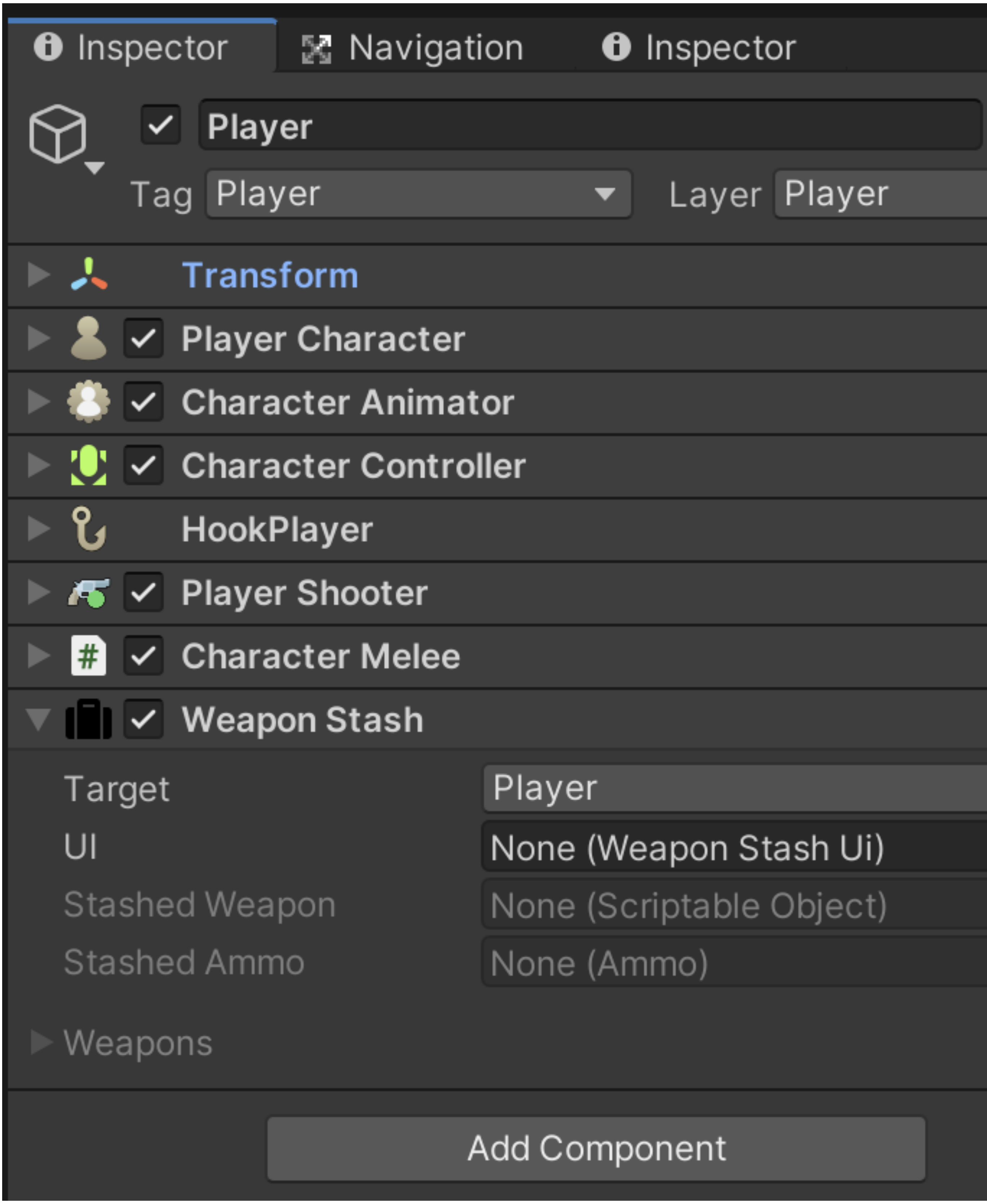
Destroy After Finishing □

Weapon Stashes

Weapon Stashes is a lightweight inventory system to assign weapons to a Player Character and allow the player to s

Weapon Stash Component

A **Weapon Stash** must be added to a game object (usually the player):



☑ Player

Tag Player ▾ Layer Player

▶ Transform

▶ ☑ Player Character

▶ ☑ Character Animator

▶ ☑ Character Controller

▶ HookPlayer

▶ ☑ Player Shooter

▶ ☑ Character Melee

▼ ☑ Weapon Stash

Target	Player
UI	None (Weapon Stash Ui)
Stashed Weapon	None (Scriptable Object)
Stashed Ammo	None (Ammo)

▶ Weapons

Adding a Weapon

There are two actions for adding weapons: Give Shooter Weapon and Give Melee Weapon. When configuring these

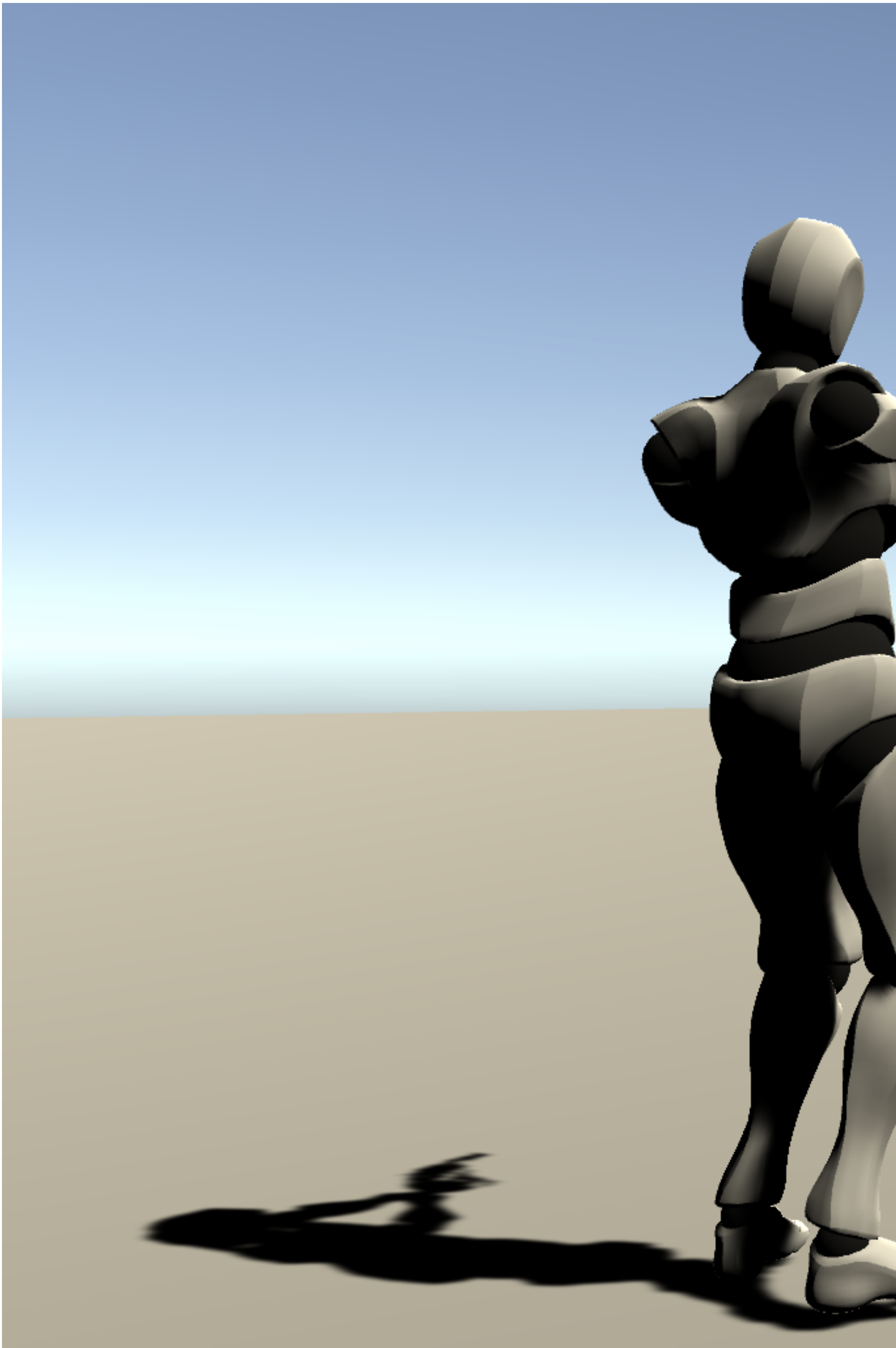
Changing Weapons

The current weapon can be switched to the next (or previous) weapon in the stash with the **Cycle to Next Weapon** ac

Weapon Stash UI Component

The Weapon Stash UI component allows a stash's current weapon and ammo (if any) to be displayed on screen. Thi

The **Combat Examples** module includes a **WeaponStashUI** prefab that demonstrates how to use the component:

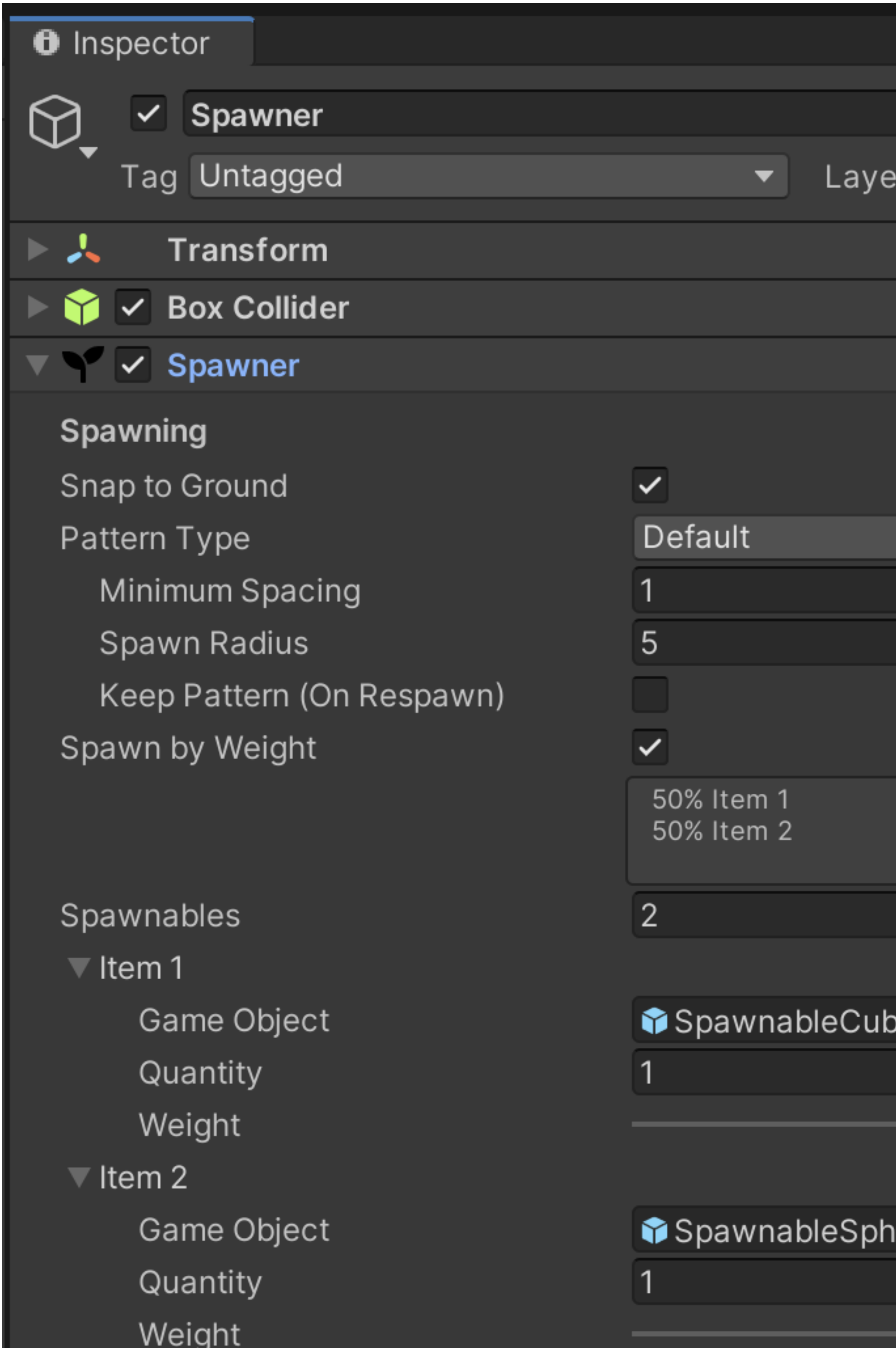


Spawner

A **Spawner** manages the creation of game objects (often enemies, loot, and visual effects like flames). The **Combat**
multiple pools of spawnable objects.

Spawner Component

A **Spawner** component can be added to any game object. It has built in support for spawning, despawning, and resp
collider. Manual control of the component via Game Creator actions is also supported.



Inspector

☑ Spawner

Tag Untagged Laye

Transform

☑ Box Collider

☑ Spawner

Spawning

Snap to Ground ☑

Pattern Type Default

Minimum Spacing 1

Spawn Radius 5

Keep Pattern (On Respawn) □

Spawn by Weight ☑

50% Item 1
50% Item 2

Spawnables 2

Item 1

Game Object SpawnableCub

Quantity 1

Weight

Item 2

Game Object SpawnableSph

Quantity 1

Weight

Spawning

Snap to Ground

When enabled, the Y position of the spawned object is dynamically set to the ground under the object - effectively "sn
spawned under or above uneven terrain, then hovering out of site or falling (if the spawned object has a rigid body c

Pattern Type

Spawning can be done in a variety of patterns, relative to the position of the **Spawner**:

- **Default**
  - Randomly places spawned objects in a developer designated radius around the **Spawner**.
  - If the number of objects is sufficiently large and the spawn radius is too small/minimum spacing is too large
- **Spiral**
  - Places spawned objects in a classic spiral pattern, with the center of the spiral originating at the **Spawner**.
  - This pattern is useful for small clusters of objects, like a small group of ~5 enemies.
- **Random with Overlap**
  - Randomly places spawned objects within a developer defined 3 dimensional area around the **Spawner**.
  - The objects may (or may not) overlap, depending on object density in the spawn area size.
  - This pattern is useful for spawning visual effects (e.g. flames, dust, etc.).
- **At Spawner Origin**
  - Places all objects exactly at the **Spawner's** origin.
  - This pattern is useful for spawning a single complicated prefab (rather than spawning parts of the prefab as

Spawn by Weight

If enabled, a spawnable is randomly spawned using a simple weight-based selection formula.

For example, if spawnable 1 has a weight of `25` and spawnable 2 has a weight of `100`, spawnable 1 will be spawne

```
Spawnable 1: 25 / (25 + 100) = 20%
Spawnable 2: 100 / (25 + 100) = 80%
```

Spawnables

The list of **spawnables** defines which (and how many) objects are spawned by the **Spawner** component. If the **Spaw**
disregarded.

Respawning

Spawnable objects can be automatically respawned either immediately after the player re-enters the spawn zone or a

Enabled

If enabled, spawned objects will be automatically respawned when the player re-enters the spawn zone.

Cooldown

If enabled, a respawn will only happen after a specified number of seconds.

Despawning

Despawn by Distance

If enabled, spawned objects will be automatically despawned when the player leaves the spawn zone and is a set dis

Auto-calculate Distance

If enabled, the despawning distance will be automatically computed. The auto-calculated spawn distance is the radiu **BoxCollider**.

Despawn Distance

The distance the player has to travel from the **Spawner's** origin before the objects are automatically despawned. This

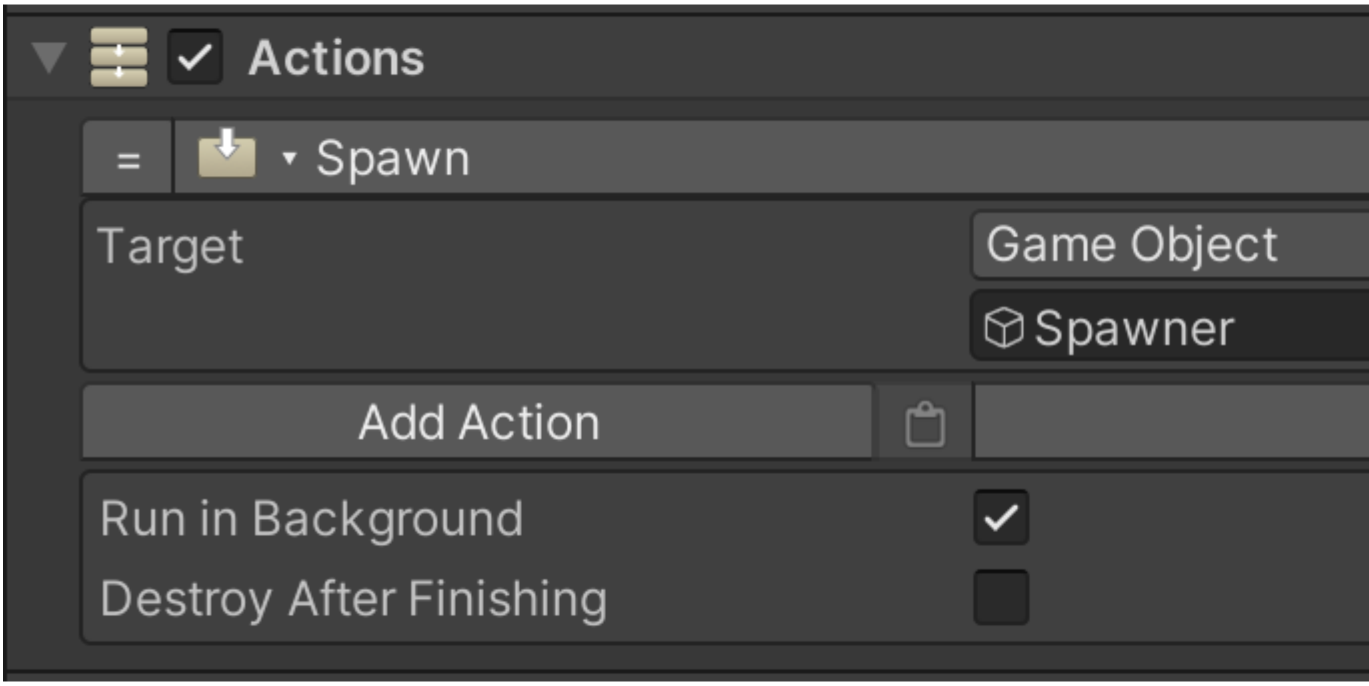
Game Creator Actions

A **Spawner** can be manually controlled with the **Spawn** and **Despawn** actions, included with the **Combat** module. Any standard Game Creator trigger can be used to invoke these actions.

Spawn Action

Select an object that has a **Spawner** component as the actions target.

> **Disable the Despawn by Distance option when manually controlling the Spawner with triggers/actions, or the objec



Target: Game Object

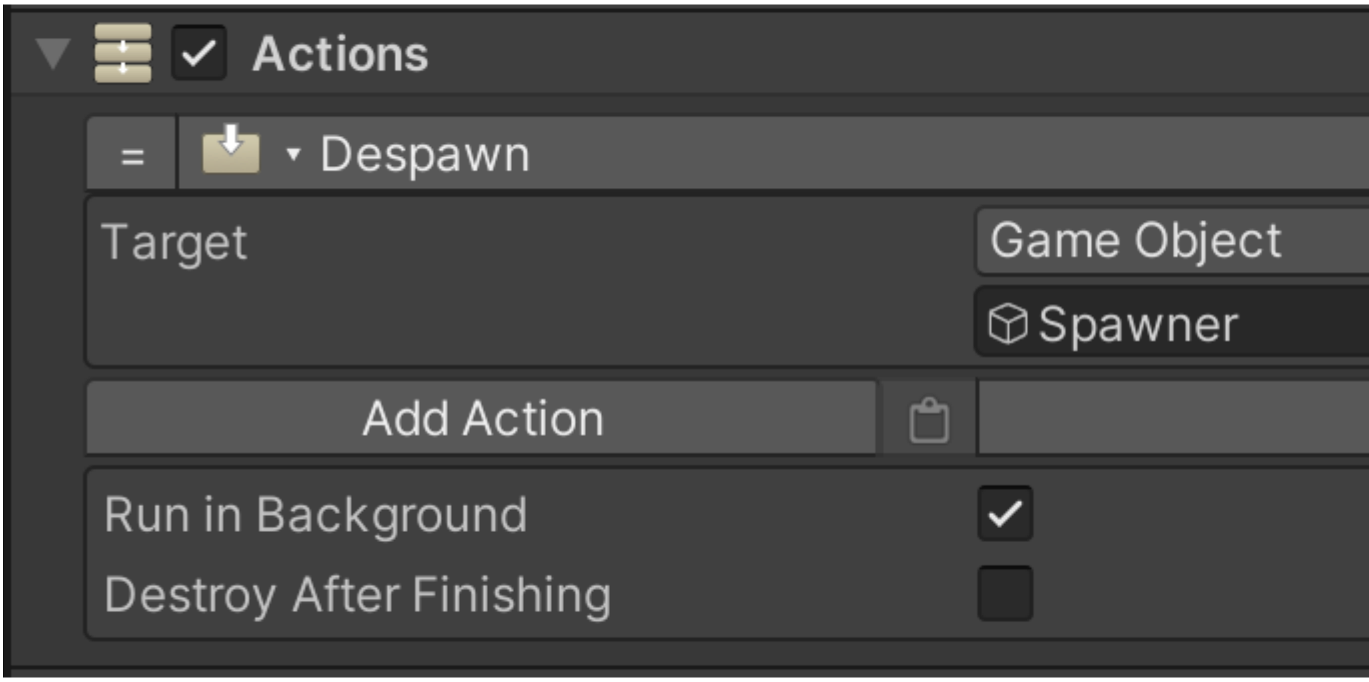
Spawner

Run in Background ☑

Destroy After Finishing □

Despawn Action

Select an object that has a **Spawner** component as the actions target.



Target: Game Object

Spawner

Run in Background ☑

Destroy After Finishing □

Roadmap

v0.8.0

General

- Shooter module is now an optional dependency. Support is now enabled through the **Combat (Shooter)** module.

Targeting Features

- Additional targeting actions to support the Behavior module.
- Improve Mouse Targeting -- Click current target to deselect it. -- Click no target to deselect the current target. -- Mouse targeting now enabled through the Targeter component.

Weapon Stash Feature

- New action to remove weapon from stash.

v0.7.0

Targeting Features

- New public API for Targeter component: -- HasTarget -- GetCurrentTarget -- IsCurrentTarget -- SetCurrentTarget -- SetTargetingEnabled -- ToggleTargetingEnabled -- CycleToNextTarget
- New actions to enable, disable, and toggle targeting.
- New action to switch to next target.
- New "On Target Changed" trigger.
- Accessibility Module (by Pivec Labs) mobile touchstick compatibility.
- Mouse targeting w/ optional hover indicator (i.e. Targetable can be selected with mouse).
- Targeter's built-in user input controls can now be disabled.
- Targeter can now be configured to not auto-acquire first/next target.

Spawning Features

- Spawn by Weight (i.e. random spawn chance).

Bug Fixes

- Targetable's internal event now correctly cleaned up.

- When created from GameObject menu, a Spawner's collider now is set as a trigger by default.

v0.6.0

- Spawner component.

v0.5.0

- Melee targeting integration.
- Weapon Stashes (weapon carrying/switching feature).

v0.4.1

- Removed .blend files that caused an issue when Blender was not installed.

v0.4.0

- Targeting by visibility.
- Support for destructible targets.

v0.3.0

- Proximity-based weapon targeting.
- Target indicator.
- Homing projectile.

Possible Future Features

- Aim-assist.
- Headshots/body part targeting.
- Simplified death action for targetable characters.
- In editor weapon positioning.
- Proximity mines.
- Dual-wielding.

- When a target is defeated, switch to its nearest neighboring target (instead of closest to the player).
- Left/right target cycling (instead of nearest/farthest from player).
- AI targeting.