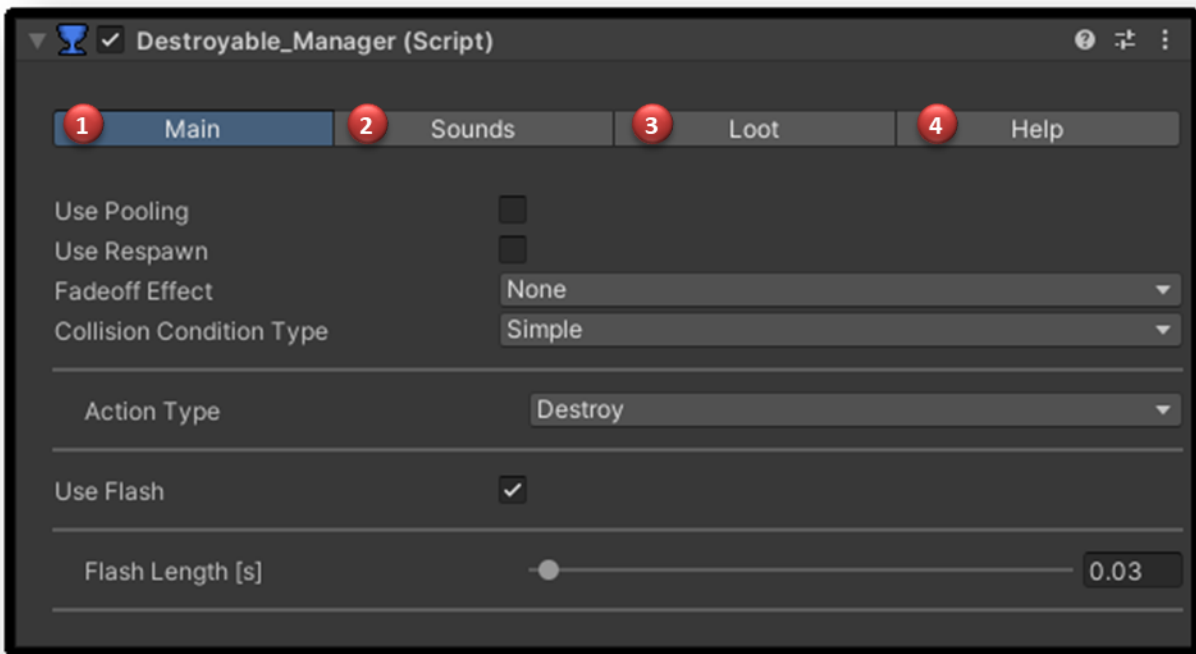


Destroyable_Manager

Description

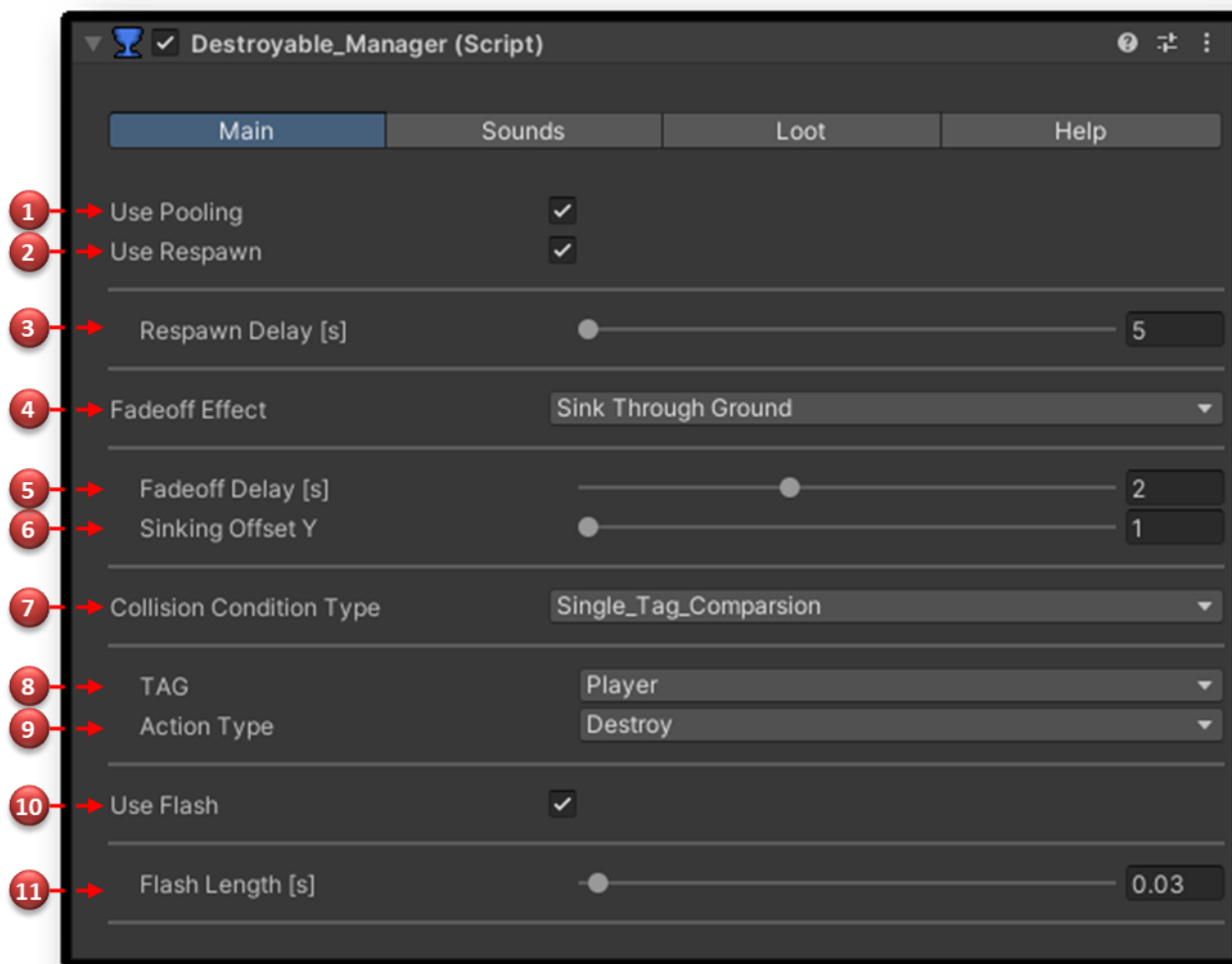
A GameObject with the "Destroyable_Manager" component is **required** for breakable items to function correctly. If the user has not created a "Destroyable_Manager" in the scene, it will be automatically added with default settings upon the first collision with a breakable item. The "Destroyable_Manager" with default settings is located in the Resources folder and should not be modified.



The "Destroyable_Manager" custom inspector window has been divided into four tabs: Main, Sounds, Loot, and Help.

1	Main Tab	Handles settings and features for breakable items.
2	Sounds Tab	Manages sounds associated with breakable items.
3	Loot Tab	Organizes the global loot table used with breakable items.
4	Help Tab	Provides quick access to learning resources and developer support.

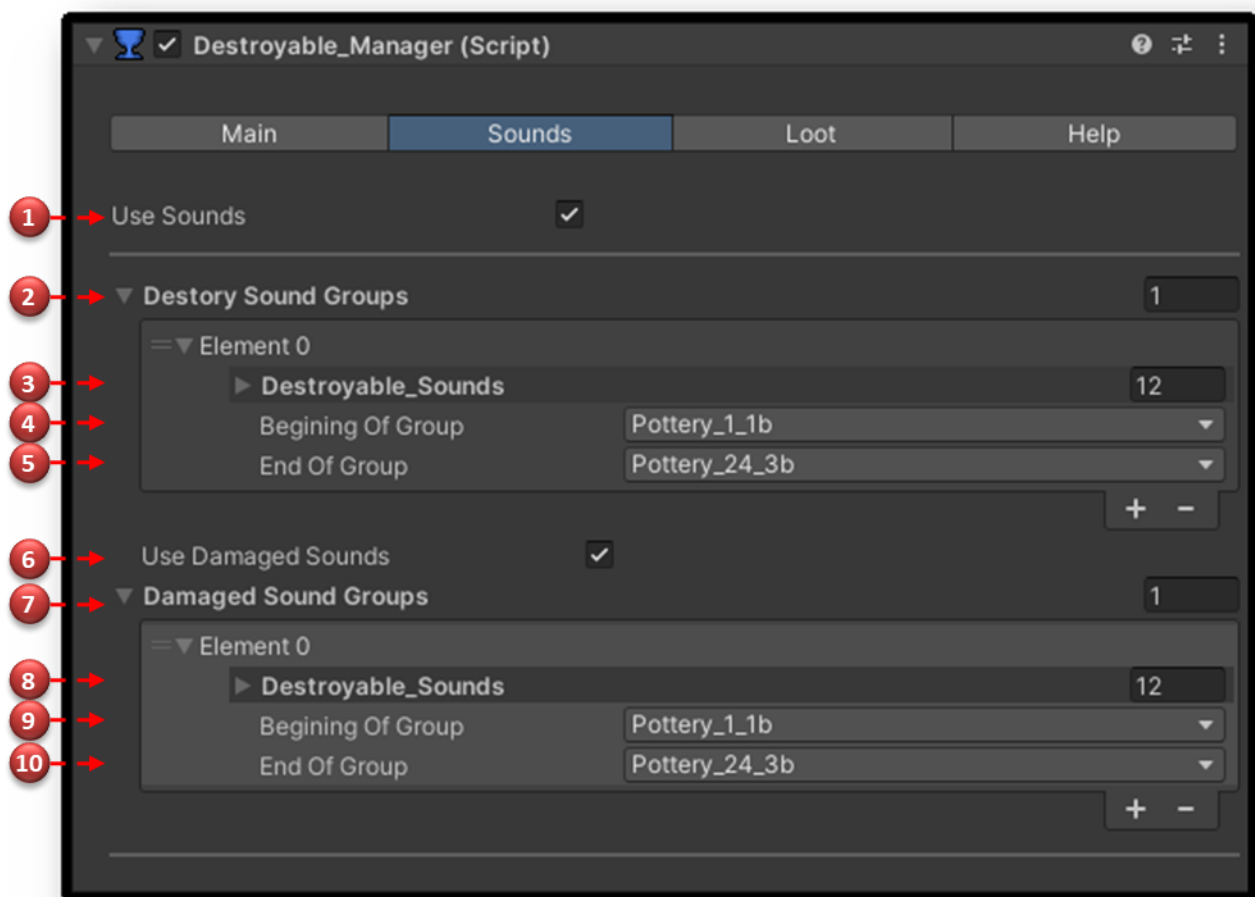
Main Tab Description



1	Use Pooling	When enabled, destroyed items will be recovered and reused. In order to work Fadeoff Effect need to be set to 'Disappear', or 'SinkThroughGround'
2	Use Respawn	When enabled, destroyed items will disappear for some declared by user amount of time. After which will appear again ready to be destroyed.
3	Respawn Delay [s]	Allows setting the amount of time in seconds after which each item will be respawned.
4	FadeOff Effect	Allows setting up the 'FadeOffEffect' value to either 'None', 'Disappear', or 'SinkThroughGround'. The FadeOffEffect will be executed after the destroyed item falls to pieces.
5	FadeOff Delay [s]	Allows setting the amount of time in seconds after which each FadeOffEffect will be executed.
6	Sinking Offset Y	Allows setting the distance of sinking which will occur during 'FadeOffEffect' execution.
7	Collision Condition Type	Allows setting up the 'CollisionConditionType' value to either 'None', 'Simple', 'SingleTagComparison', or

		'MultipleTagComparison'. The 'CollisionConditionType' will be executed each time either when OnCollisionEnter occurs or when the method TryDestroy_AccordingToAction() is called on Destroyable_WholeItem.
8	TAG	A single TAG required during OnCollisionEnter. If it matches, the chosen ActionType will be executed.
9	Action Type	Allows setting the 'ActionType' value to either 'Destroy', 'DamageConstant', 'DamageRandom', or 'ChanceRandom'. 'ActionType' will be executed after a successful 'CollisionConditionType' check.
10	Use Flash	When enabled, destroyed items will flash for short amount of time.
11	Flash Length [s]	Allows setting the duration, in seconds, of the flash effect.

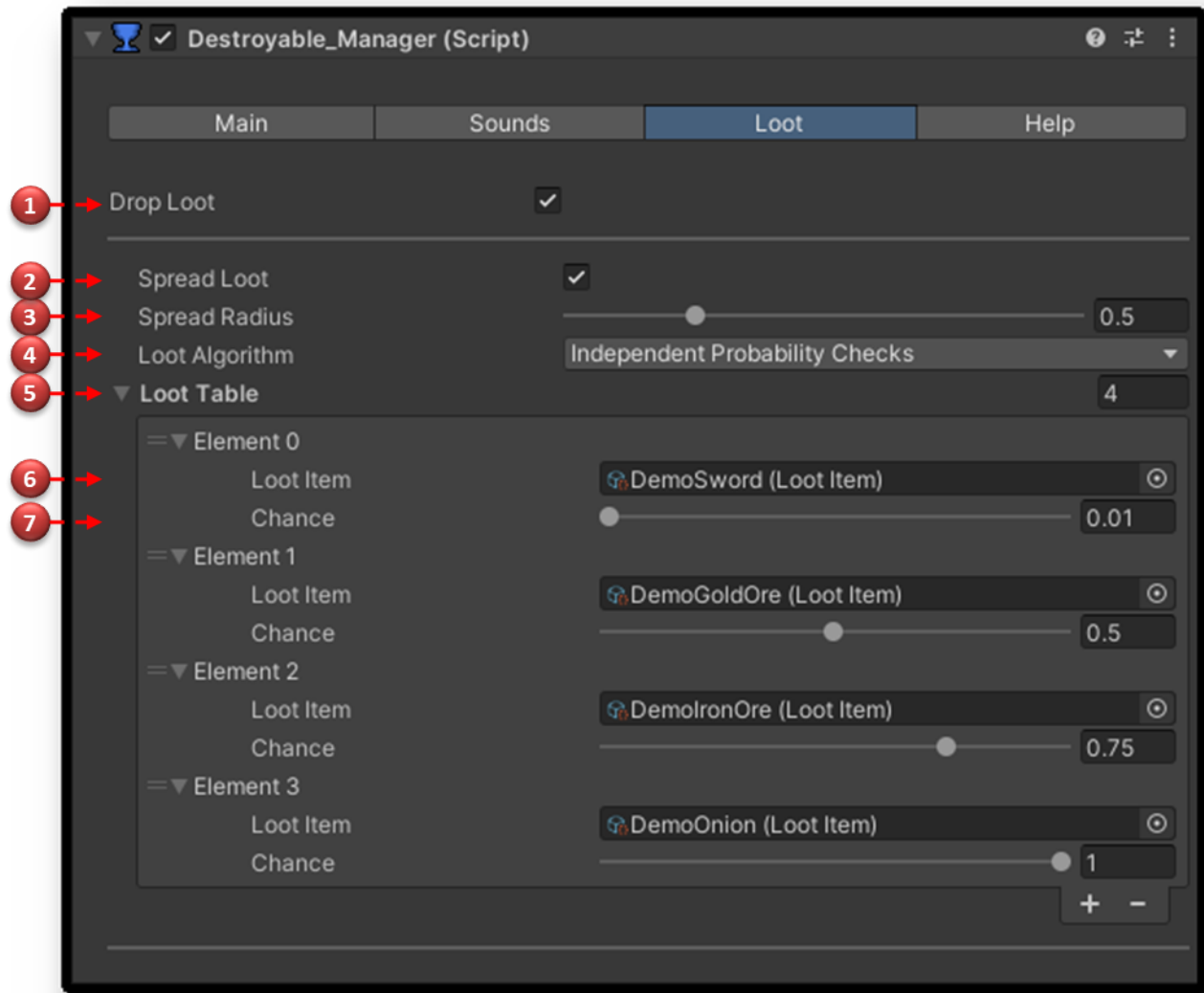
Sounds Tab Description



1	Use Sounds	When enabled, destroyed items will play sound according to declared Sound_Groups.
2	Destroy Sound Groups	Allows adding Destroy Sound Groups for each type of Breakable Items. All items in Sound Group will share

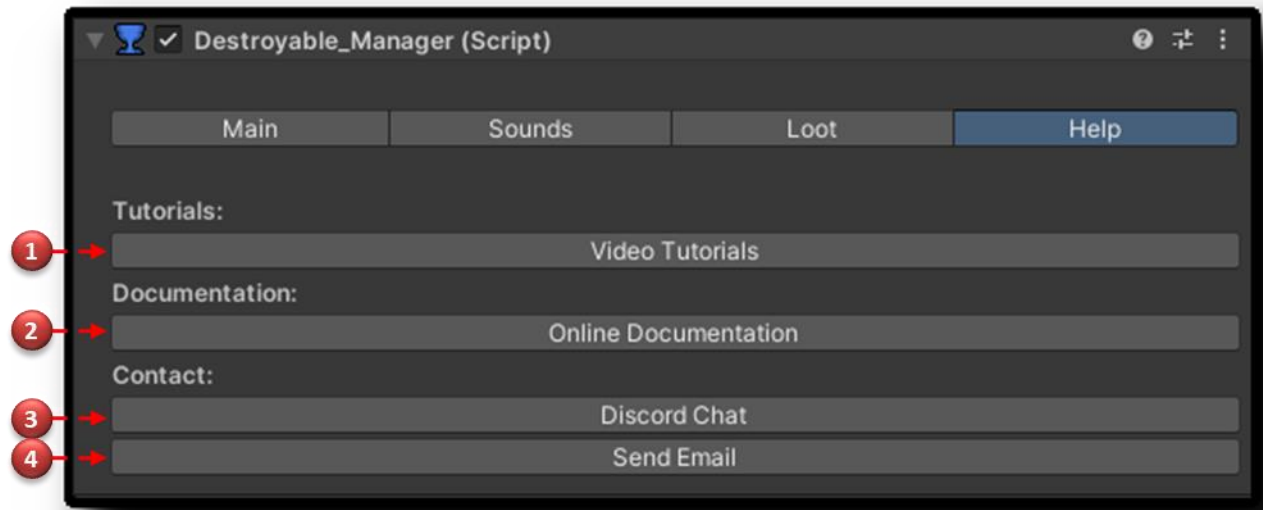
		same sounds.
3	DestoyableSounds Array	Allows assign sound file assign to Sound Group. If array had more than 1 file, during each destruction random clip will be used.
4	Beginning Of Group	Allows choosing first item of Sound Group.
5	End Of Group	Allows choosing last item of Sound Group.
6	Use Damaged Sounds	When enabled, damaged items will play sound according to declared Sound_Groups.
7	Damaged Sound Groups	Allows adding Damaged Sound Groups for each type of Breakable Items. All items in Sound Group will share same sounds.
8	DestoyableSounds Array	Allows assign sound file assign to Sound Group. If array had more than 1 file, during each damage random clip will be used.
9	Beginning Of Group	Allows choosing first item of Sound Group.
10	End Of Group	Allows choosing last item of Sound Group.

Loot Tab Description



1	Drop Loot	When enabled, damaged items will spawn loot GameObject according to User defined 'Loot Table'.
2	Spread Loot	When enabled, loot prefabs will be spawned around the destroyed item within the declared radius.
3	Spread Radius	Define the radius for spreading the loot.
4	Loot Algorithm	Choose the Loot Algorithm. Independent Probability Checks or Roulette Wheel Selection .
5	Loot Table	Define the Loot Table by choosing prepared 'LootItems' ScriptableObjects and setting the 'Chance' for dropping them.
6	Loot Item	Assign Loot Item ScriptableObjects as item which will be dropped.
7	Chance	Define Chance of dropping item.

Help Tab Description



1	Video Tutorials	When clicked, hyperlink to YouTube playlist will be activated.
2	Online Documentation	When clicked, hyperlink to Github manual file will be activated.
3	Discord Chat	When clicked, hyperlink to CatBorg discord channel will be activated.
4	Send Email	When clicked, create email window will pop up.

(Enum) Collision Condition Type

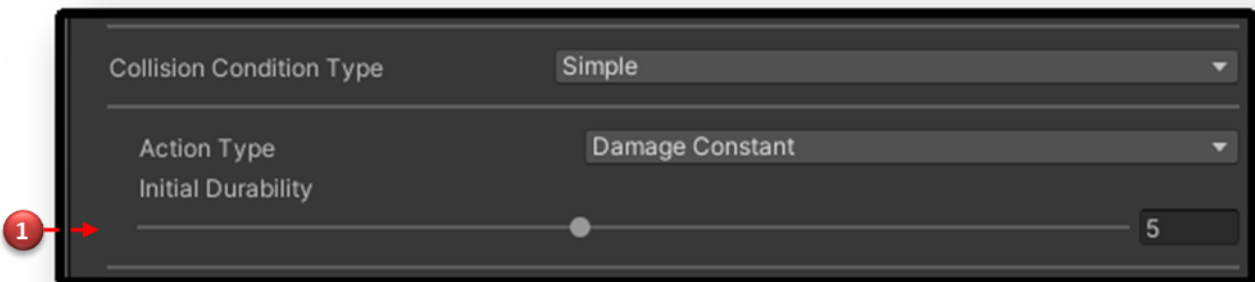
The Collision Condition Type enum defines the behavior of an item when a collision occurs, specifically during the OnCollisionEnter event. It provides several options to control how collisions are handled based on tag comparisons:

- **None:** No action will be taken when a collision occurs. This option effectively disables collision handling for the item.
- **Simple:** No tag check will be conducted. Any collision will trigger the chosen ActionType.
- **Single_Tag_Comparison:** A single tag is required during OnCollisionEnter. If the tag of the colliding object matches the specified tag, the chosen ActionType will be executed.
- **Multiple_Tag_Comparison:** Multiple tags are required during OnCollisionEnter. If the tag of the colliding object matches any of the specified tags, the chosen ActionType will be executed.

(Enum) Action Type

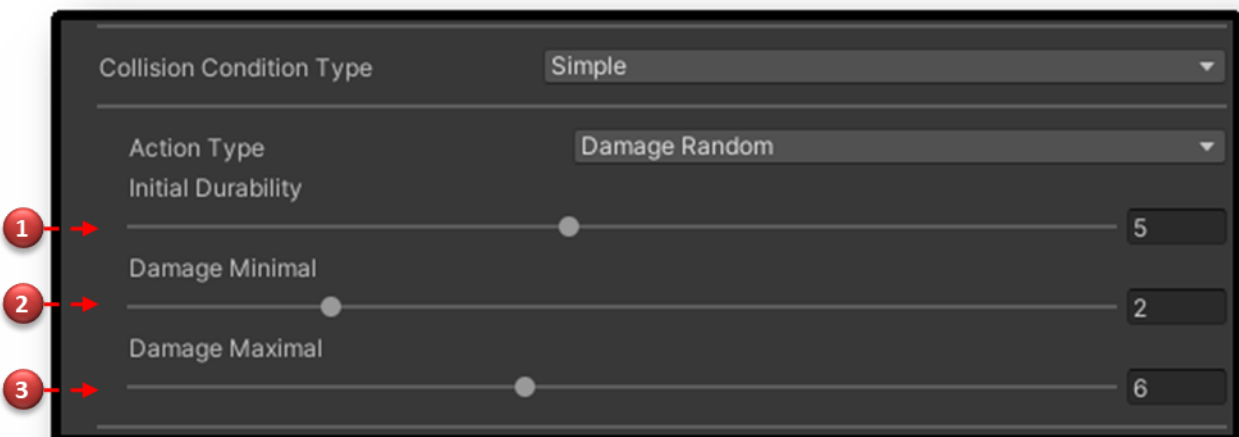
The ActionType enum defines various actions that can be executed based on specific conditions in a game environment. Each ActionType offers distinct functionality to enhance gameplay dynamics.

- **Destroy:** This action immediately removes an object from the scene upon meeting specified conditions. It's useful for scenarios where objects need to disappear or be eliminated upon collision.
- **DamageConstant:** This action inflicts a consistent amount of damage to affected objects or entities. It provides predictable outcomes for game mechanics involving health reduction or durability reduction.



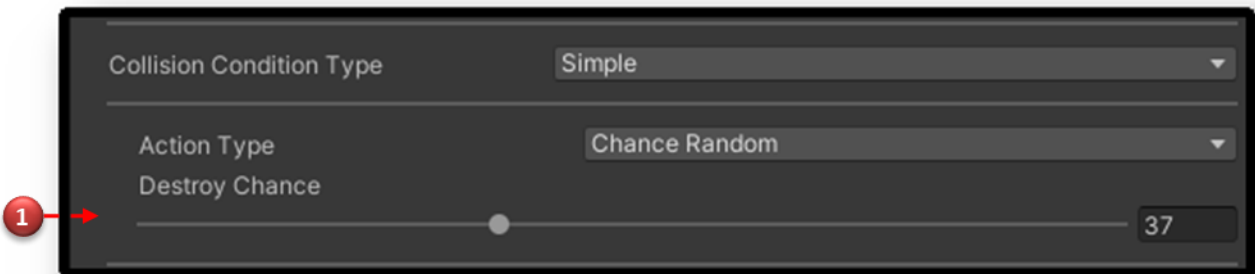
1	Initial Durability	Allows setting the initial value of 'Item Durability'. This value functions similarly to health points for game characters; when it reaches 0, the item will break into parts.
---	--------------------	--

- **DamageRandom:** This action introduces variability by inflicting random amounts of damage within a specified range. It adds unpredictability to gameplay, making each encounter or collision unique in its consequences.



1	Initial Durability	Allows setting the initial value of 'Item Durability'. This value functions similarly to health points for game characters; when it reaches 0, the item will break into parts.
2	Damage Minimal	Specifies the minimum amount of damage subtracted from 'Item Durability' with each hit or trigger
3	Damage Maximal	Specifies the maximum amount of damage subtracted from 'Item Durability' with each hit or trigger

- **ChanceRandom:** This action introduces a probabilistic element where outcomes are determined based on chance. It allows for scenarios where the success or failure of an action is influenced by probability, adding a layer of risk and reward to gameplay decisions.



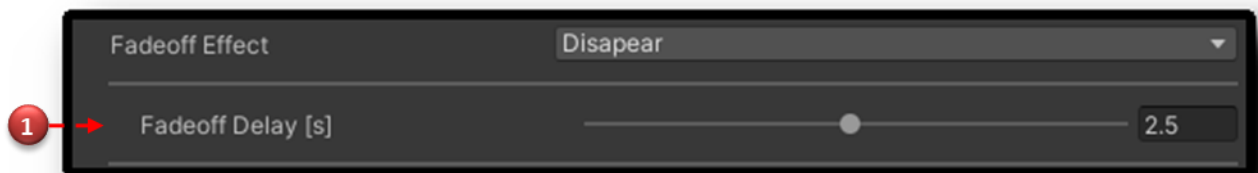
1	Destroy Chance	Allows setting the 'Destroy Chance' value, which ranges from 1 to 100. A value of 100 ensures the item will definitely fall apart when triggered.
---	----------------	---

These ActionType options are designed to be flexible and adaptable, allowing game developers to create diverse and engaging interactions based on collision conditions specified using the Collision Condition Type enum.

(Enum) FadeOff Effect

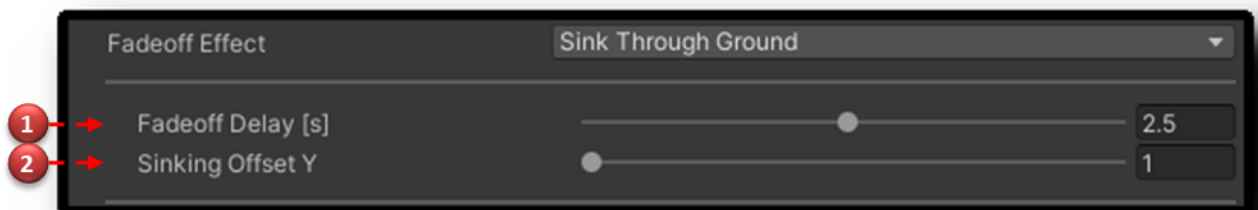
The FadeOffEffect enum provides options for visual effects that occur after an item falls to pieces in a game environment. Each FadeOffEffect option enhances the visual feedback and realism of the destruction process.

- **None:** No additional visual effect will occur after the item is destroyed. This option is suitable for scenarios where the destroyed object remains on the ground in pieces, contributing to environmental detail.
- **Disappear:** The destroyed item is instantly removed from the world after a user-defined amount of time. This option offers a quick and clean removal of the destroyed object, streamlining gameplay and visual continuity.



1	Fadeoff Delay [s]	Allows setting the amount of time in seconds after which each FadeOffEffect will be executed.
---	-------------------	---

- **SinkThroughGround:** The destroyed item sinks through the ground or platform after falling to pieces. This effect adds a dynamic element to the destruction sequence, providing a visual conclusion to the interaction and enhancing realism.



1	Fadeoff Delay [s]	Allows setting the amount of time in seconds after which each FadeOffEffect will be executed.
2	Sinking Offset Y	Allows setting the distance of sinking which will occur during 'FadeOffEffect' execution.

These FadeOffEffect options enable game developers to tailor the post-destruction visual experience, ensuring that the game environment reacts realistically and engagingly to interactions and events within the game world. Each option serves a distinct purpose in enhancing gameplay immersion and visual storytelling.

(Scriptable Objects) Loot Item

Description

Contains a reference to an Item GameObject Prefab, which defines the visual and functional representation of the loot item in the game. This Scriptable Object can be used to add an unlimited amount of user data, such as the item's name, amount, description, rarity, and any special properties or behaviors it may have when spawned.

Creation

To add a new Loot Item Scriptable Object in Unity, follow these steps:

1. Navigate to the **Project Window**: In the Unity Editor, go to the Project Window where your assets are displayed.
2. Create a New Loot Item: Right-click in the Project window, then select **Create** from the context menu. Navigate to **ScriptableObjects** and then select **LootItem**. This will create a new Loot Item Scriptable Object.
3. **Name Your Loot Item**: You will be prompted to name your new Loot Item Scriptable Object. Enter a descriptive name, such as "Excalibur".
4. Configure the Loot Item: Select the newly created Loot Item in the Project window to view its properties in the Inspector. Currently here, you can only assign GameObject prefab reference if willing to configure various attributes, such as the item's name, amount, description, rarity, check out and edit LootItem.cs.

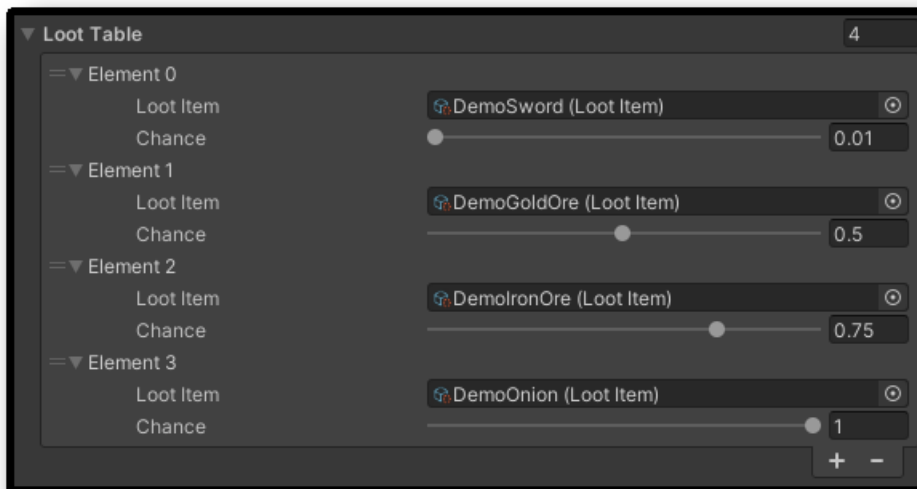
(Loot Algorithm) Independent Probability Checks

Description

This algorithm evaluates each element in the Loot Table individually. For each element, the algorithm performs a random check using the following condition:

$\text{Random.Range}(0,100) < \text{Chance}$

If the condition is met, the loot item is added to the spawned loot list.



Example

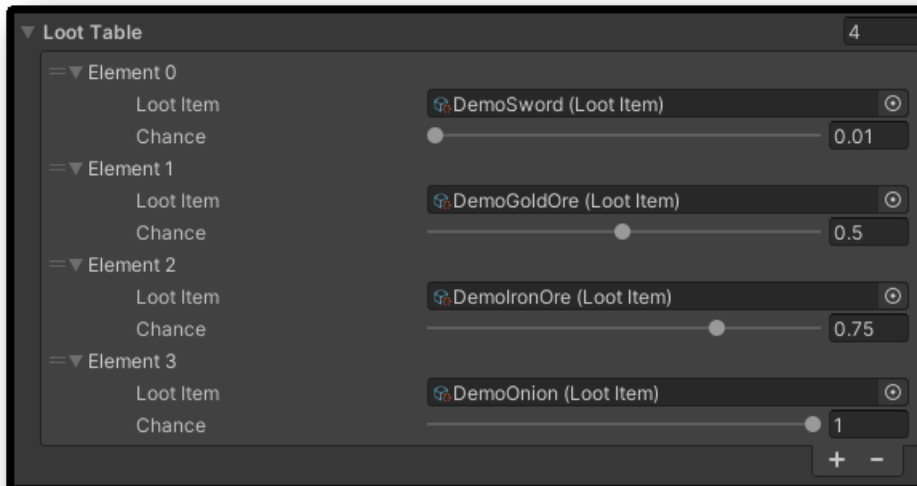
Let's conduct **theoretical** loot table cast according to Independent Probability Checks.

1. **Loot Item "Demosword"**: $\text{Random.Range}(0, 1)$ resulted in 0.5. Since 0.5 is higher than the Chance (0.01), the item **will not be added** to the spawned loot list.
2. **Loot Item "DemoGoldOre"**: $\text{Random.Range}(0, 1)$ resulted in 0.27. Since 0.27 is smaller than the Chance (0.5), the item **will be added** to the spawned loot list.
3. **Loot Item "DemoIronOre"**: $\text{Random.Range}(0, 1)$ resulted in 0.89. Since 0.89 is higher than the Chance (0.75), the item **will not be added** to the spawned loot list.
4. **Loot Item "DemoOnion"**: $\text{Random.Range}(0, 1)$ resulted in 0.07. Since 0.07 is smaller than the Chance (1), the item **will be added** to the spawned loot list.

As a result of above **theoretical** cast, 2 items will be spawned.

(Loot Algorithm) Roulette Wheel Selection, Description

This algorithm assigns each element in the Loot Table a segment on a roulette wheel based on its Chance value. A single random number is generated to determine which segment is selected. The item corresponding to the selected segment is added to the spawned loot list. The process ensures that items with higher chances occupy larger segments, increasing their likelihood of being selected.



Example

Let's conduct **theoretical** loot table cast according to Independent Probability Checks.

1. The algorithm calculates the total weight (sum of all Chance values): $0.01 + 0.5 + 0.75 + 1 = 2.26$.
2. A random number between 0 and 2.26 is generated, e.g., 1.3.
3. The algorithm iterates through the Loot Table, summing the Chance values until the sum exceeds the random number:
 - o **Loot Item "Demosword"** segment: 0 to 0.01
 - o **Loot Item "DemoGoldOre"** segment: 0.01 to 0.51
 - o **Loot Item "DemoIronOre"** segment: 0.51 to 1.26
 - o **Loot Item "DemoOnion"** segment: 1.26 to 2.26
4. Since the random number (1.3) falls within the **"DemoOnion"** segment (1.26 to 2.26), **"DemoOnion"** is selected and added to the spawned loot list.

As a result of above **theoretical** cast, **always only one** item will be spawned.

Destroyable_WholeItem

TODO