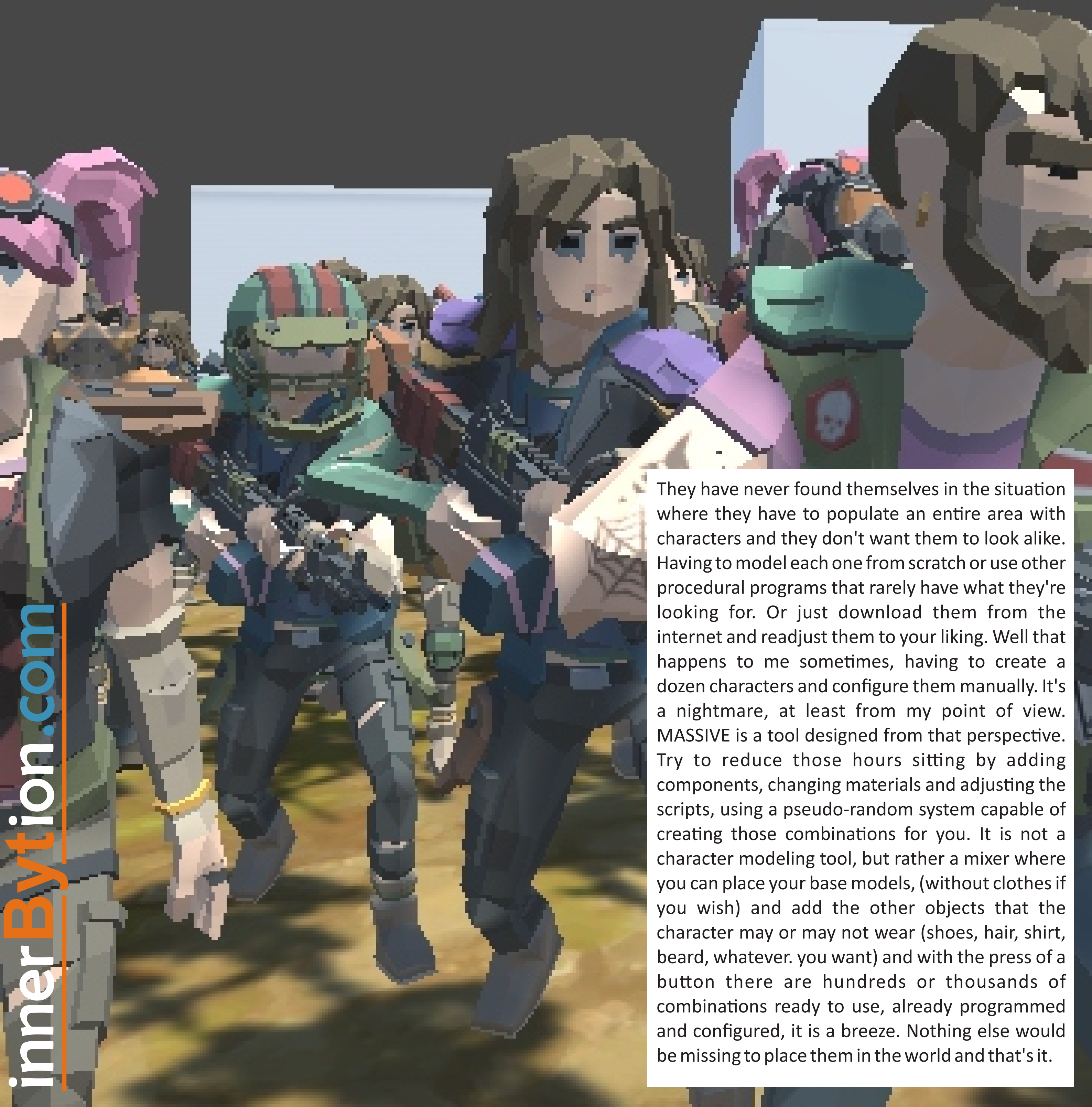
MASSIVE

CHARACTER CONFIGURER



They have never found themselves in the situation where they have to populate an entire area with characters and they don't want them to look alike. Having to model each one from scratch or use other procedural programs that rarely have what they're looking for. Or just download them from the internet and readjust them to your liking. Well that happens to me sometimes, having to create a dozen characters and configure them manually. It's a nightmare, at least from my point of view. MASSIVE is a tool designed from that perspective. Try to reduce those hours sitting by adding components, changing materials and adjusting the scripts, using a pseudo-random system capable of creating those combinations for you. It is not a character modeling tool, but rather a mixer where you can place your base models, (without clothes if you wish) and add the other objects that the character may or may not wear (shoes, hair, shirt, beard, whatever. you want) and with the press of a button there are hundreds or thousands of combinations ready to use, already programmed and configured, it is a breeze. Nothing else would be missing to place them in the world and that's it.

innerBytion.com

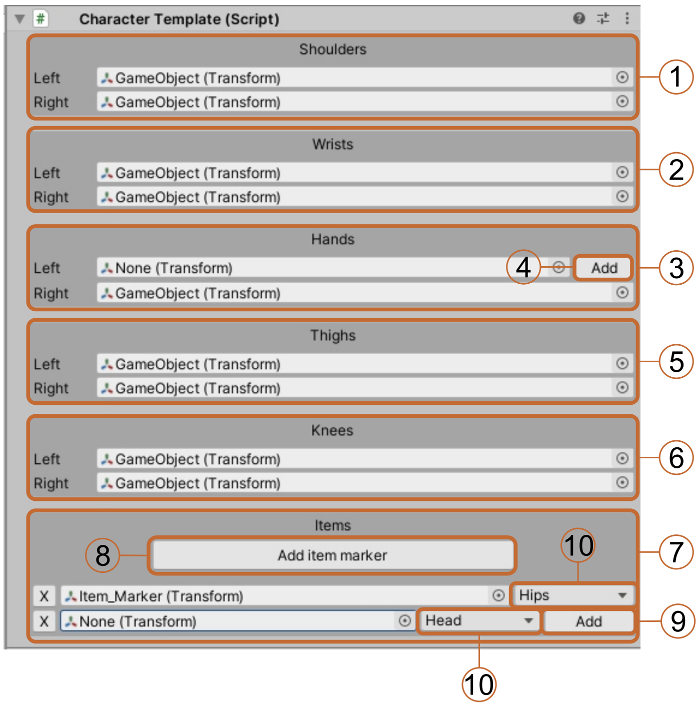
SETTING UP
THE CHARACTER TEMPLATE

The Character Template is the humanoid model by which the system will be governed to decide the positions of the objects that are going to be placed in the final model. Each template will contain an Animator component. It is recommended to use the same template avatar for all models that require Animator component.

1. Protection for shoulders in case the characters are wearing armor. These points allow you to place independent parts of the reinforcement.
2. Protection and objects for the dolls.
3. Equipment point for weapons or objects that the characters can carry in their hands.
4. Add marker: Add a marker in the avatar position corresponding to that marker.
5. Marker for thigh protection.
6. Protection for the knees.

   Items:
7. Panel to add markers for items. The items will be objects that will be placed on the character. They will only have an aesthetic purpose to increase the variety of the characters.
8. Add new space for item marker.
9. If there is no defined marker, one will be added.
10. Parent bone that will contain the marker

After adding all the markers must be configured so that they have a suitable position and rotation for the objects that are going to be instantiated in them. This can be done by placing objects related to these markers within the scene, then rotating and moving the marker and not the object itself.


Character Template (Script)

Shoulders
Left GameObject (Transform)
Right GameObject (Transform)

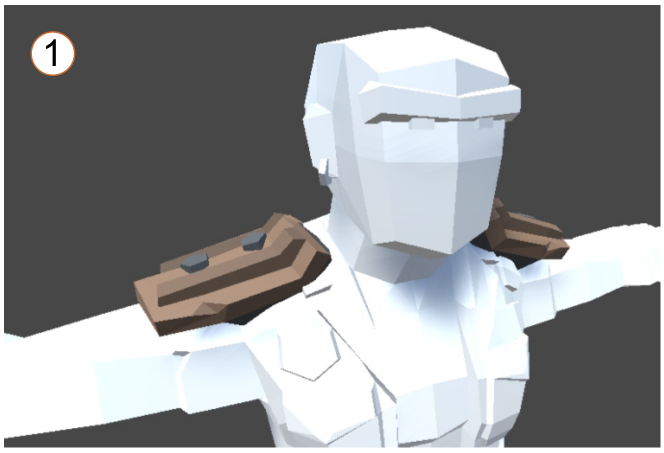
Wrists
Left GameObject (Transform)
Right GameObject (Transform)

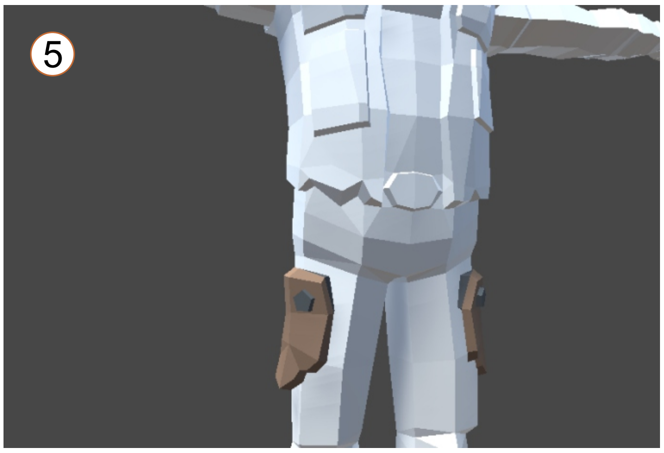
Hands
Left None (Transform) Add
Right GameObject (Transform)

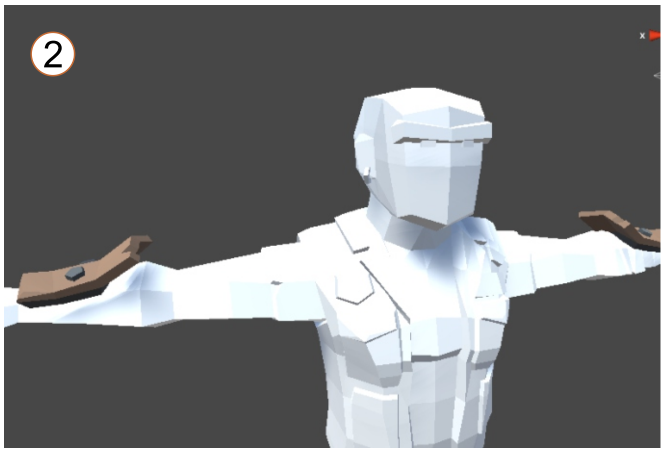
Thighs
Left GameObject (Transform)
Right GameObject (Transform)

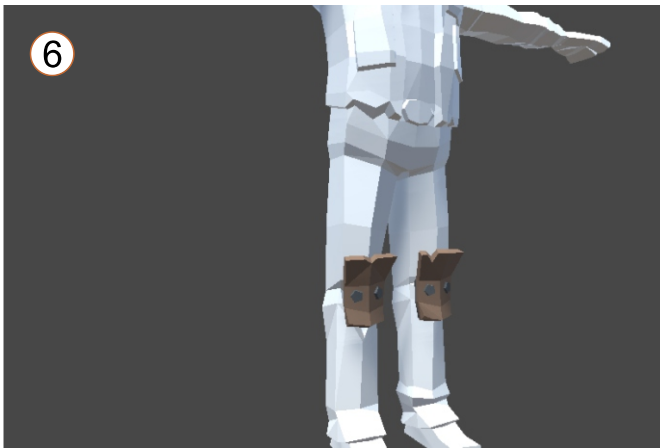
Knees
Left GameObject (Transform)
Right GameObject (Transform)

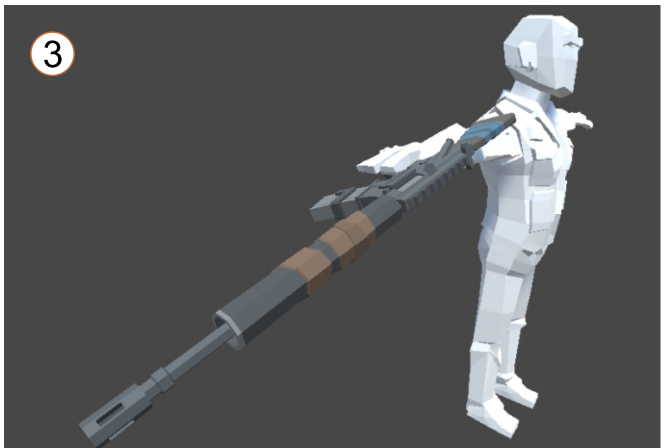
Items
Add item marker
X Item_Marker (Transform) Hips
X None (Transform) Head Add









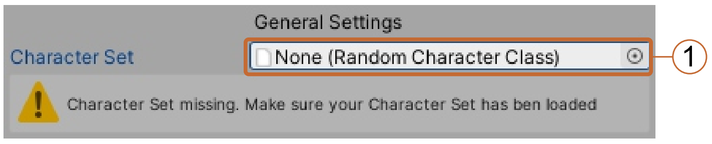


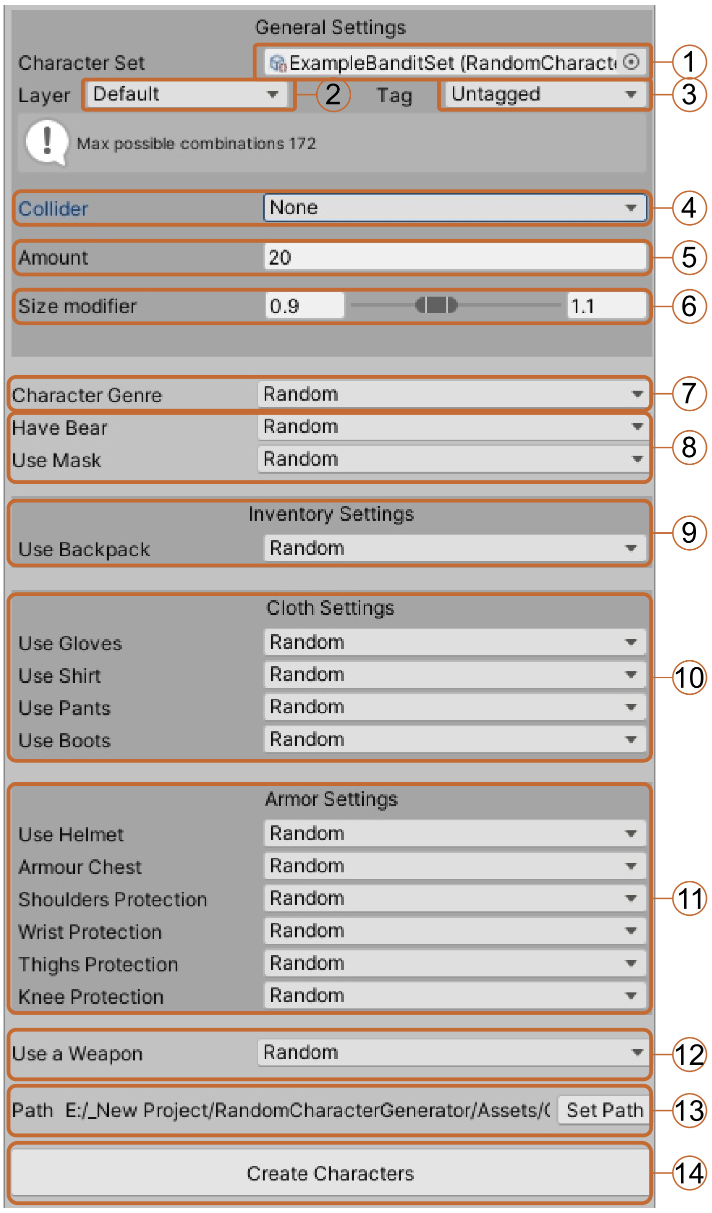
CREATING A NEW CHARACTER SET

The **Character Set** window allows the creation of character sets as prefabs stored in a folder within the project. The window is accessed through MASSIVE> Create Characters in the menu bar that will display a window as follows:

Before continuing to work in this window, a Character Class must be assigned to it. This class can be created in the project tab by right clicking MASSIVE> Character Class. This will create a character class that can be loaded into any of the character generators. This class is the one that contains all the objects and models that can be placed in the characters to be generated.

After the new class of characters created is loaded, all the options for customizing the characters will be displayed in the window.


General Settings
Character Set: None (Random Character Class) — 1
Character Set missing. Make sure your Character Set has ben loaded

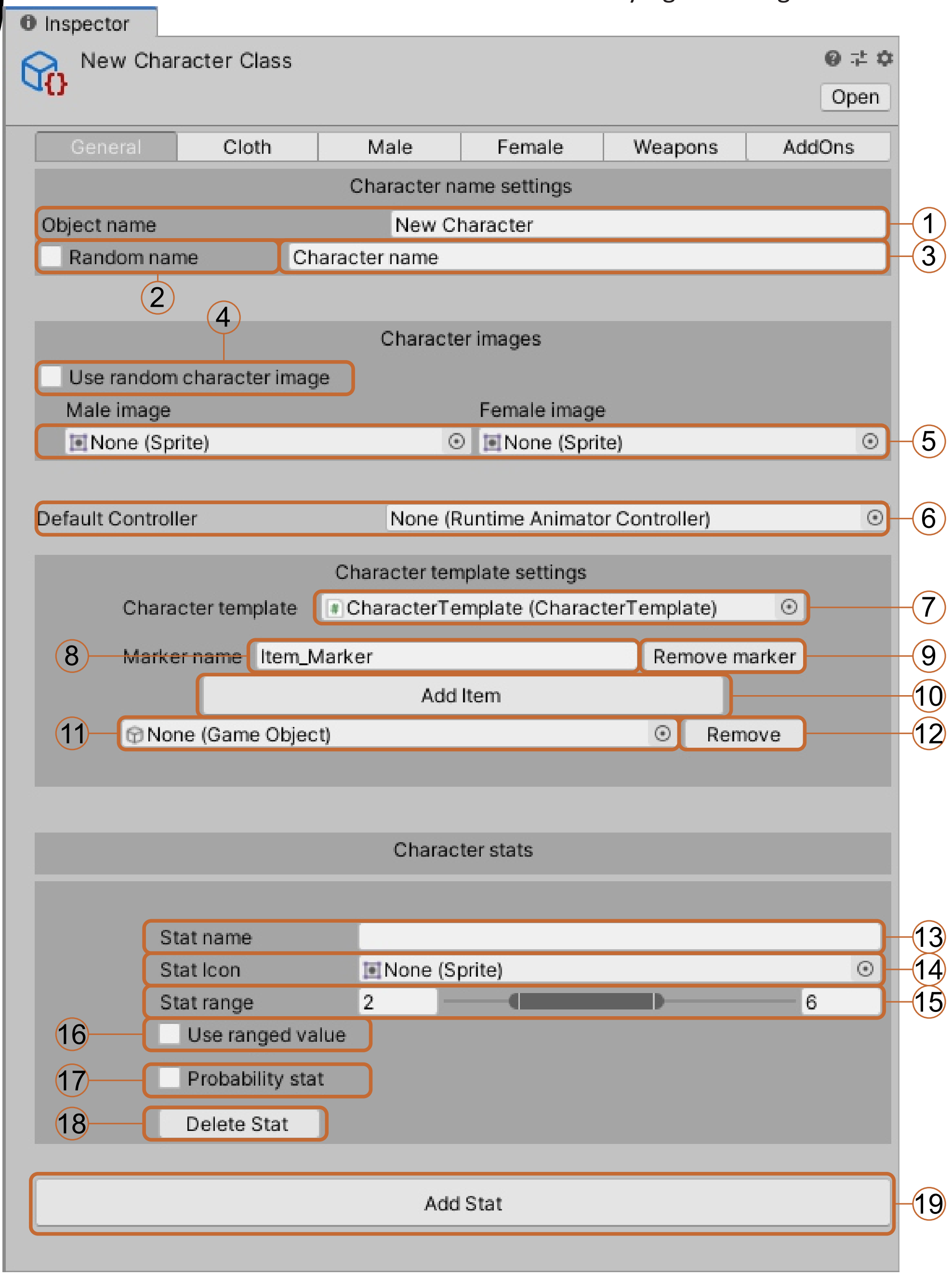

General Settings
Character Set: ExampleBanditSet (RandomCharact — 1
Layer: Default — 2
Tag: Untagged — 3
Max possible combinations 172
Collider: None — 4
Amount: 20 — 5
Size modifier: 0.9 — 1.1 — 6
Character Genre: Random — 7
Have Bear: Random — 8
Use Mask: Random — 8
Inventory Settings
Use Backpack: Random — 9
Cloth Settings
Use Gloves: Random
Use Shirt: Random
Use Pants: Random
Use Boots: Random — 10
Armor Settings
Use Helmet: Random
Armour Chest: Random
Shoulders Protection: Random
Wrist Protection: Random
Thighs Protection: Random
Knee Protection: Random — 11
Use a Weapon: Random — 12
Path E:/_New Project/RandomCharacterGenerator/Assets/(Set Path — 13
Create Characters — 14

1. The Character Class where the objects that will be placed on the resulting character are stored.
2. The name of the layer in which the character will be placed.
3. The label that will be placed on the character
4. What type of Collider will the character use. There are three options:
   None: No Collider will be added to the generated character.
   Box Collider: A BoxCollider will be added to the generated character
   Capsule Collider: A CapsuleCollider will be added to the character.
5. The number of characters that will be generated once the generate button is pressed.
6. The scale modifier for the generated character. It will be a random value within the selected range. This will affect the size of the character.
7. The gender of the character. It has three variants:
   Female: All characters generated will be female.
   Male: All the characters generated will be male.
   Random: The gender of the character will be randomly decided.
8. Generic parameters of the characters. Each of these parameters has three values:
   Yes: This parameter will always be used in all generations.
   No: The parameter will never be used.
   Random: It is decided randomly if the parameter is used or not.
9. Parameter that decides if the characters have inventory or not. It is related to the instantiation of backpacks and bags in the characters. So if a character has inventory, he will have a cargo object such as a backpack or a bag.
10. Clothing placement parameters for the characters.
11. Parameters for the placement of the armor of the characters.
12. Parameters for the placement of weapons on the characters.
13. The path in which all the character prefabs resulting from the generation will be stored.
14. Character creation button. Generate and save all characters in the specified path.

CHARACTER CLASS

GENERAL SETTINGS

It is where the objects and bodies of characters that will be mixed are stored. It can be created from the Project window by right clicking Create> MASSIVE> Character Class.



Inspector

New Character Class

Open

General | Cloth | Male | Female | Weapons | AddOns

Character name settings

Object name: New Character (1)

Random name (2) | Character name (3)

Character images

Use random character image (4)

Male image: None (Sprite) | Female image: None (Sprite) (5)

Default Controller: None (Runtime Animator Controller) (6)

Character template settings

Character template: CharacterTemplate (CharacterTemplate) (7)

Marker name (8): Item_Marker | Remove marker (9)

Add Item (10)

None (Game Object) (11) | Remove (12)

Character stats

Stat name (13)

Stat Icon: None (Sprite) (14)

Stat range: 2 — 6 (15)

Use ranged value (16)

Probability stat (17)

Delete Stat (18)

Add Stat (19)

Objects and general information for any characters within the class.

1. **Object name**: Name that will be assigned to the GameObject created.
2. **Random name**: Use a random name taken from a predefined list.
3. **Character name**: Predefined name of the character.
4. **Use random image**: Use a random image for the character taken from a list.
5. **Male / Female image**: Predefined image for both sexes.
6. **Default Controller**: The default Animator Runtime Controller that the character will have for animations.

CHARACTER TEMPLATE OPTIONS

The Character Template to use for the character. When this field is filled, the item spaces that the template has are shown, allowing you to add specific items for each position.

7. **Character template**: The character template that will be used for the character.
8. **Marker name**: The name of the marker. It can be edited.
9. **Remove marker**: Remove the marker. The marker will be removed from the template.
10. **Add item**: A new item space is added for the marker.
11. The empty field where the new item will be added.
12. **Remove**: Remove the item from the list of items belonging to this marker.

CHARACTER STATISTICS

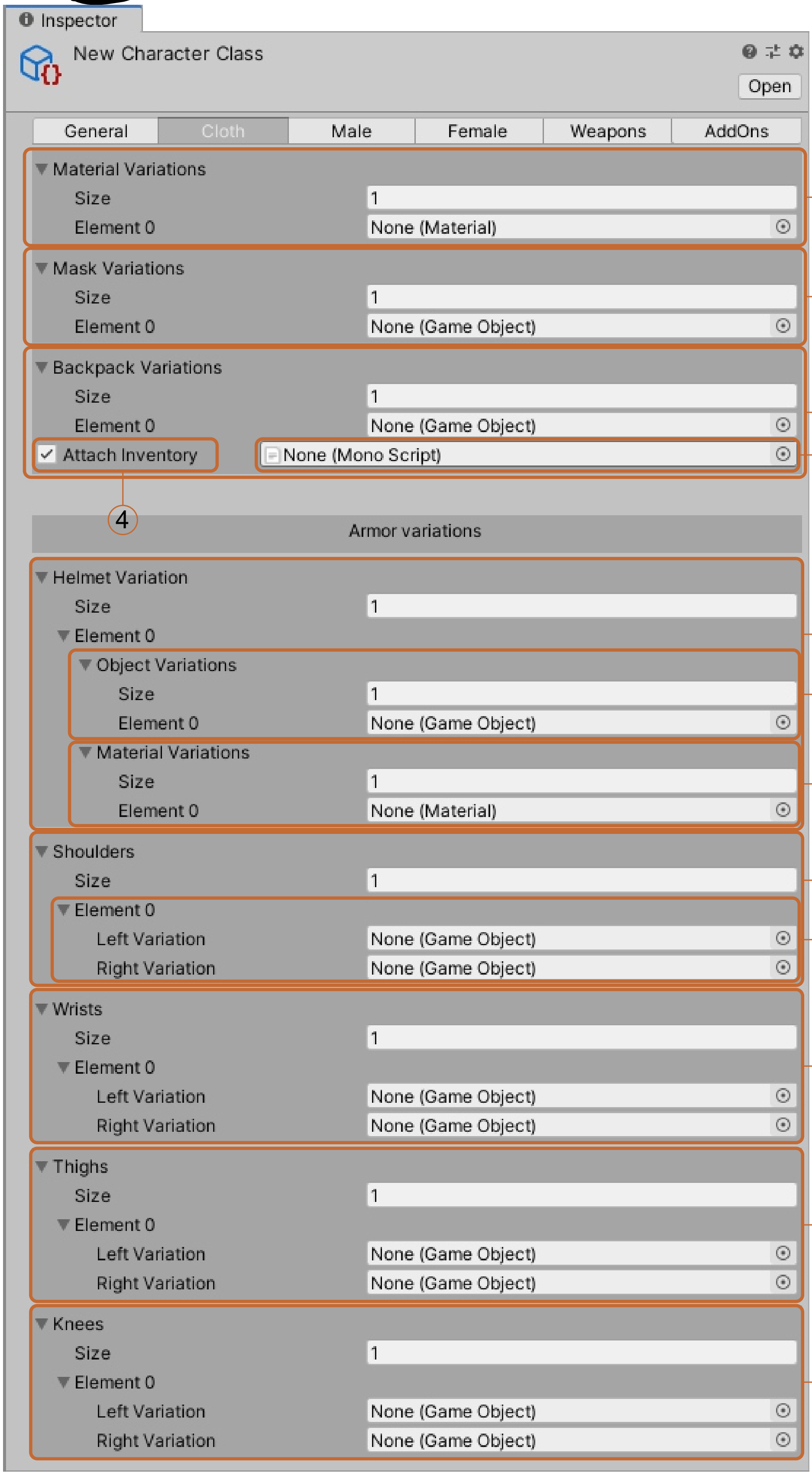
Character statistics. MASSIVE allows you to create statistics for the characters to be created. These statistics after the character is created will be transmitted to a component within the character itself called RandomCharacterBehaviour.

13. **Stat name**: Name of the statistic.
14. **Stat icon**: Statistic icon.
15. **Stat range**: Statistic range. To add more variation to the characters the stat values of the characters are randomly chosen within a range predetermined by you.
16. **Use ranged value**: In cases where this attribute is required to return varied values, the range of the statistic can be stored instead of the defined random value.
17. **Probability stat**: For cases in which the statistic requires a probability to return a value. This probability is defined by a range from which a static value will be taken that will represent the probability of this statistic for that character.
18. **Delete Stat**: Delete the statistic from the list.
19. **Add Stat**: Add a new blank statistic to the list.

CHARACTER CLASS

CLOTH

Tab in which the lists for the attachable reinforcement parts are configured. Each part corresponds to a specific body position whose positions and rotations are determined by the character template.



Inspector

New Character Class

Open

General | Cloth | Male | Female | Weapons | AddOns

Armor variations

```
▼ Material Variations
    Size                    1
    Element 0               None (Material)
▼ Mask Variations
    Size                    1
    Element 0               None (Game Object)
▼ Backpack Variations
    Size                    1
    Element 0               None (Game Object)
  ✓ Attach Inventory        None (Mono Script)

                Armor variations

▼ Helmet Variation
    Size                    1
  ▼ Element 0
    ▼ Object Variations
        Size                1
        Element 0           None (Game Object)
    ▼ Material Variations
        Size                1
        Element 0           None (Material)
▼ Shoulders
    Size                    1
  ▼ Element 0
      Left Variation        None (Game Object)
      Right Variation       None (Game Object)
▼ Wrists
    Size                    1
  ▼ Element 0
      Left Variation        None (Game Object)
      Right Variation       None (Game Object)
▼ Thighs
    Size                    1
  ▼ Element 0
      Left Variation        None (Game Object)
      Right Variation       None (Game Object)
▼ Knees
    Size                    1
  ▼ Element 0
      Left Variation        None (Game Object)
      Right Variation       None (Game Object)
```

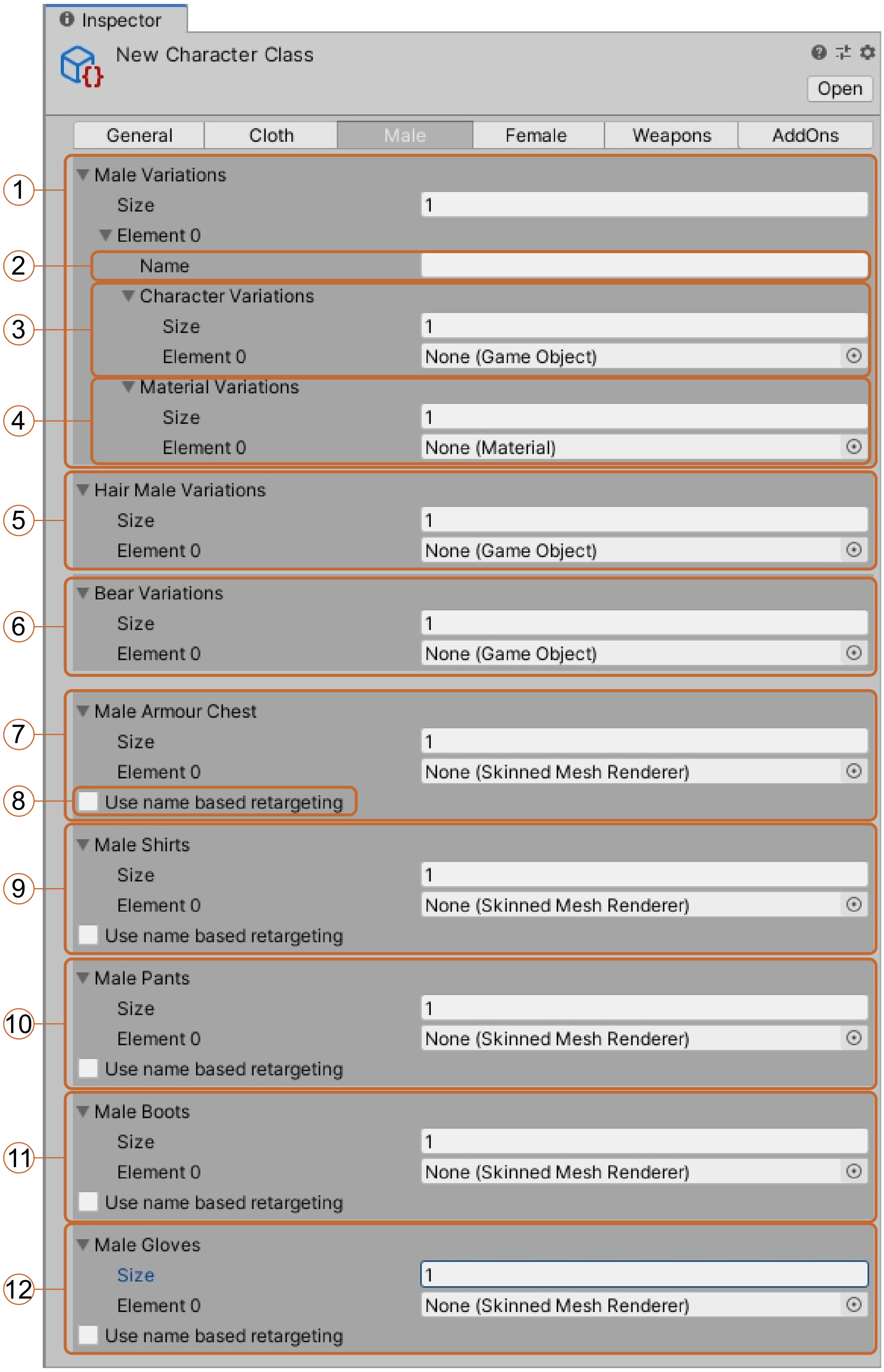
1. **Material variations**: To add more variation, MASSIVE proposes variations in the materials so that, while maintaining the same model, the textures and materials of the same vary.
2. **More variations**: The characters may or may not use masks. In this field all the objects that can perform this function are listed.
3. **Backpack variations**: Characters can carry bags or backpacks which can contain pre-configured inventories.
4. **Attach inventory**: Add inventory. Determines whether or not an inventory script is added to the character.
5. Shown when Attach inventory tab is checked. Allows you to add an inventory script that will be added to the character once created. When a script is loaded MASSIVE will request where to host a prefab that will contain the script. This will allow you to edit said script in this same window without having to change the screen and thus the character will be added with these settings.
6. **Helmet variations**: Collections of helmets that the characters can use. Some objects are divided into collections to allow objects with different texture mappings to be used.
7. **Object variations**: All the objects that can be contained in this collection.
8. **Material variations**: All material variations compatible with the texture mappings of the objects in the collection.
9. **Shoulders**: Variations of the objects that can be placed on the shoulders of the characters. It is another type of collection in which the variants for right and left must be included.
10. Object of the double collection. Requests two fields corresponding to the right and left elements.
11. **Wrist**: Protections for the wrists.
12. **Thighs**: Protections for the upper part of the character's legs.
13. **Knees**: Protections for the knees of the characters.

CHARACTER CLASS

MALE / FEMALE

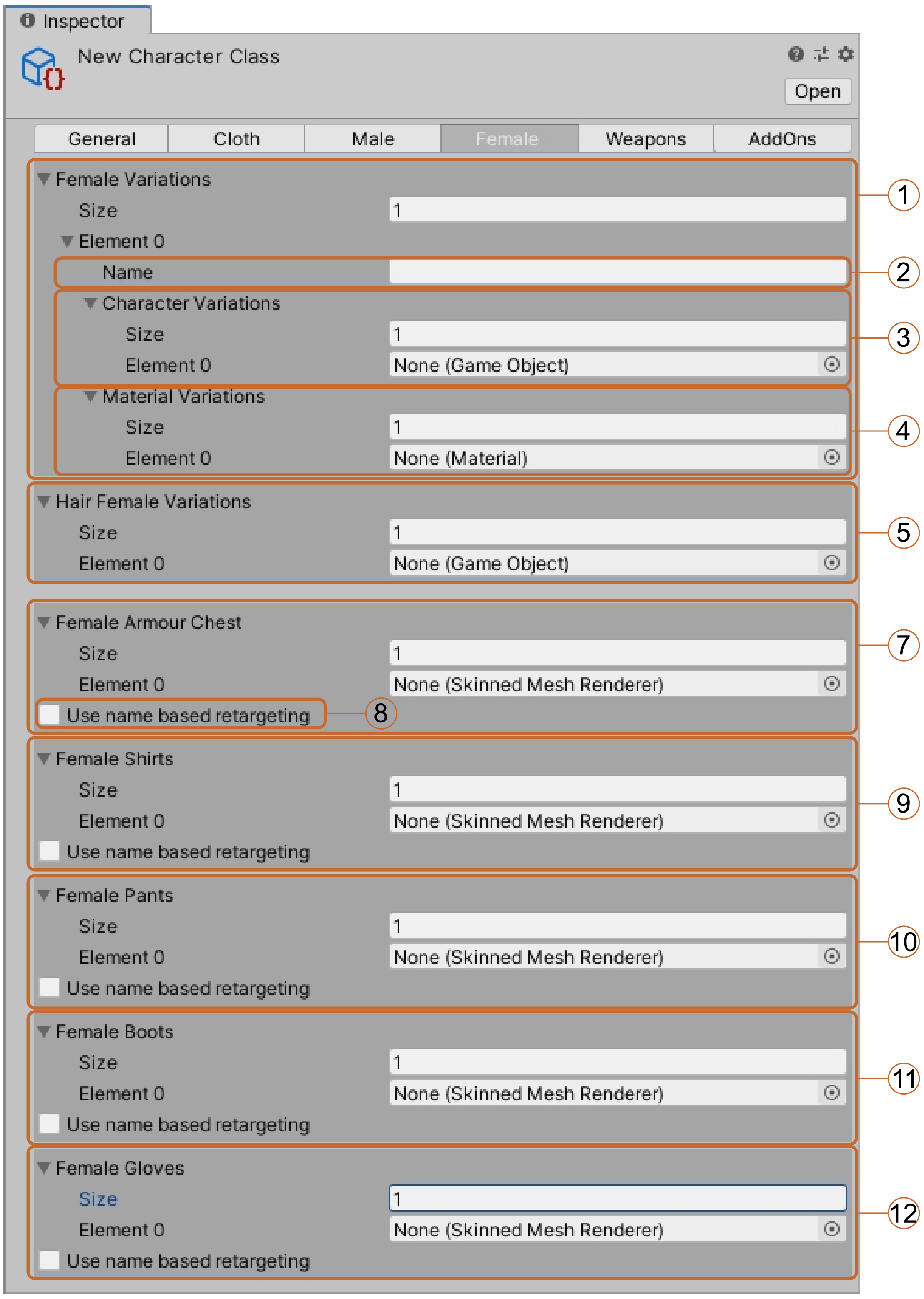
In these two tabs the objects that can vary according to the sex of the character to be generated will be placed. This category contains elements of different categories but that can only belong to a specific sex.

1. **Male variations**: Variations for the bodies of the characters. They are classified by collections to be able to use different models with different texture mappings. Only models and materials that match the same texture mapping can be in each collection.
2. **Name**: The name of the collection.
3. **Character variations**: The variations of characters in the class.
4. **Material variations**: Material variations for this collection. These materials must match the texture mappings of the models.
5. **Hair Male / Female variations**: Hair variations of the characters.
6. **Beard variations**: In the case of men, the variations that these can have in their beards.



Male Variations
Size: 1
Element 0
Name:
Character Variations
Size: 1
Element 0: None (Game Object)
Material Variations
Size: 1
Element 0: None (Material)
Hair Male Variations
Size: 1
Element 0: None (Game Object)
Bear Variations
Size: 1
Element 0: None (Game Object)
Male Armour Chest
Size: 1
Element 0: None (Skinned Mesh Renderer)
☐ Use name based retargeting
Male Shirts
Size: 1
Element 0: None (Skinned Mesh Renderer)
☐ Use name based retargeting
Male Pants
Size: 1
Element 0: None (Skinned Mesh Renderer)
☐ Use name based retargeting
Male Boots
Size: 1
Element 0: None (Skinned Mesh Renderer)
☐ Use name based retargeting
Male Gloves
Size: 1
Element 0: None (Skinned Mesh Renderer)
☐ Use name based retargeting

7. **Male / Female armor chest**: The variations of the chest armor (if any) that men can have. For the case of this object a skinned mesh is requested, it must have its Animator component with its Avatar correctly configured to avoid mistakes
8. **Use name retargeting**: This option disables the relocation of the Skinned Mesh based on the Avatar contained in the Animator component of the Mesh. Using then the repositioning based on the name of the bones of each Maya. For this function to work correctly both skeletons, the one of the character or characters and the one of the Mesh must have the same names in the hierarchy. The use of Avatar repositioning is recommended.
9. **Male / Female Shirts**: Shirts for men. In case the base models do not have clothes, clothes can be assigned for each part of the body. These will be placed in the same way that the armor is placed.
10. **Male / Female Pants**: Pants for men. In case the base models do not have clothes, clothes can be assigned for each part of the body. These will be placed in the same way that the armor is placed.
11. **Male / Female Boots**: Boots or shoes for men. In case the base models do not have clothes, clothes can be assigned for each part of the body. These will be placed in the same way that the armor is placed.
12. **Male / Female Gloves**: Gloves for men. In case the base models do not have clothes, clothes can be assigned for each part of the body. These will be placed in the same way that the armor is placed.

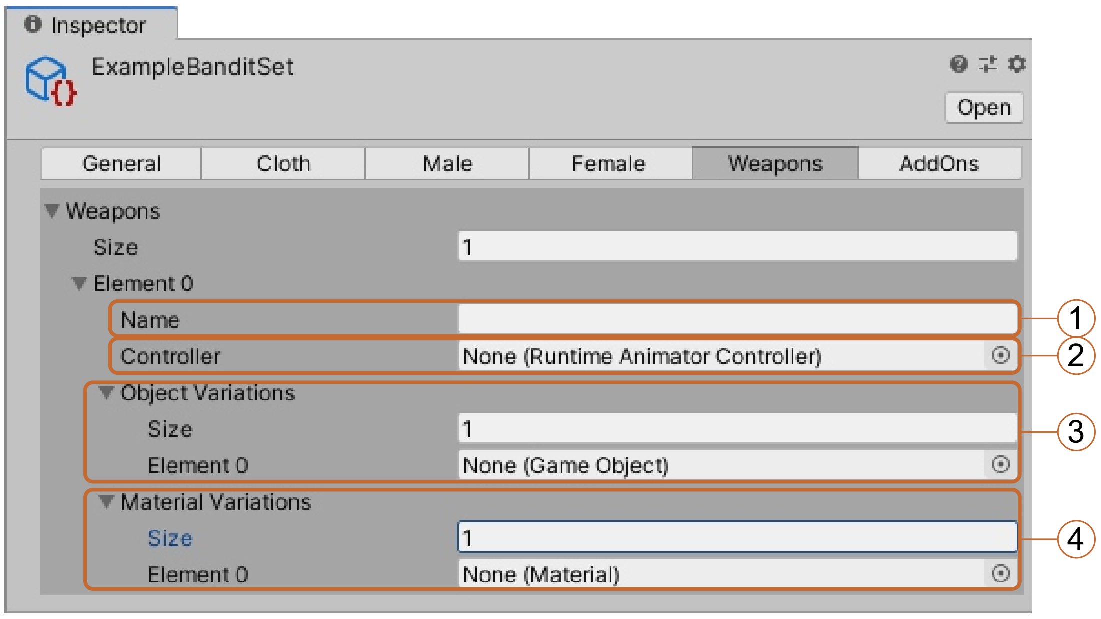


Female Variations
Size 1
Element 0
Name
Character Variations
Size 1
Element 0 None (Game Object)
Material Variations
Size 1
Element 0 None (Material)
Hair Female Variations
Size 1
Element 0 None (Game Object)
Female Armour Chest
Size 1
Element 0 None (Skinned Mesh Renderer)
Use name based retargeting
Female Shirts
Size 1
Element 0 None (Skinned Mesh Renderer)
Use name based retargeting
Female Pants
Size 1
Element 0 None (Skinned Mesh Renderer)
Use name based retargeting
Female Boots
Size 1
Element 0 None (Skinned Mesh Renderer)
Use name based retargeting
Female Gloves
Size 1
Element 0 None (Skinned Mesh Renderer)
Use name based retargeting

CHARACTER CLASS

WEAPONS

Weapons on characters are important when making an attack. In MASSIVE these can also be varied for each character.



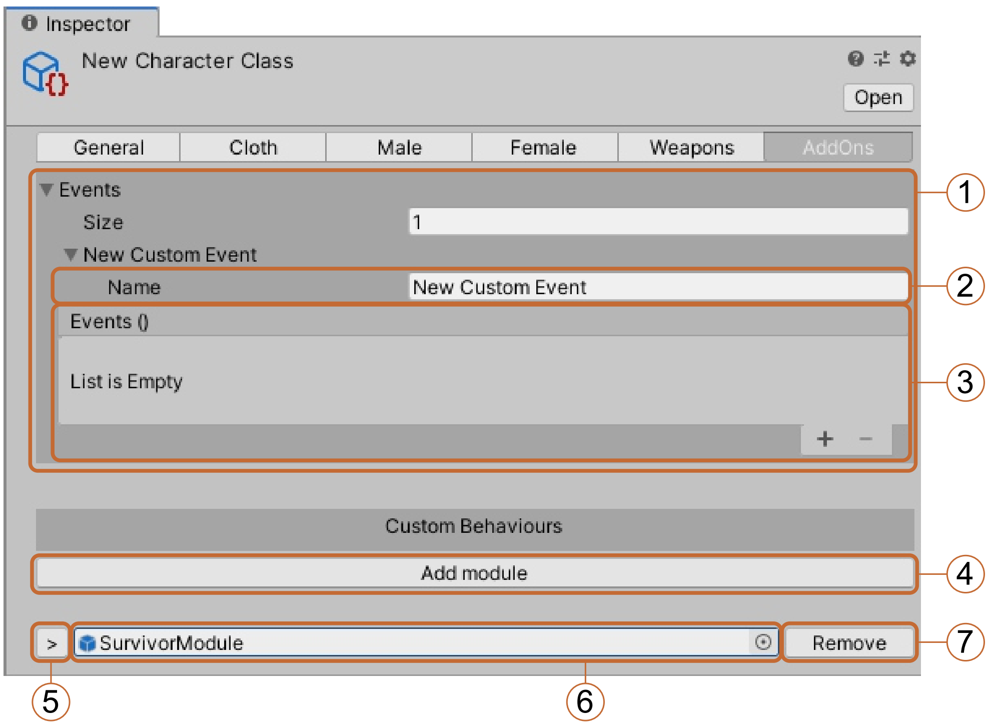
Weapons
Size 1
Element 0
Name
Controller None (Runtime Animator Controller)
Object Variations
Size 1
Element 0 None (Game Object)
Material Variations
Size 1
Element 0 None (Material)

1. **Name**: The name of the collection.
2. **Weapon Controller**: The controller that will be added to the character if he has a weapon from this collection.
3. **Object variations**: The weapon variations that this collection can contain
4. **Material variations**: Material variations that are compatible with the texture mappings in this collection.

CHARACTER CLASS

ADD ONS - CUSTOM EVENTS

The ADD ONS tab is used to add Events, Components and Behaviors to the character. These are added as components or configured in the RandomCharacterBehaviour



Events
Size 1
New Custom Event
Name New Custom Event
Events ()
List is Empty
SurvivorModule

1. **Events**: List of events that can be added to the character.
2. **Name**: Name of the event.
3. Events for this field.

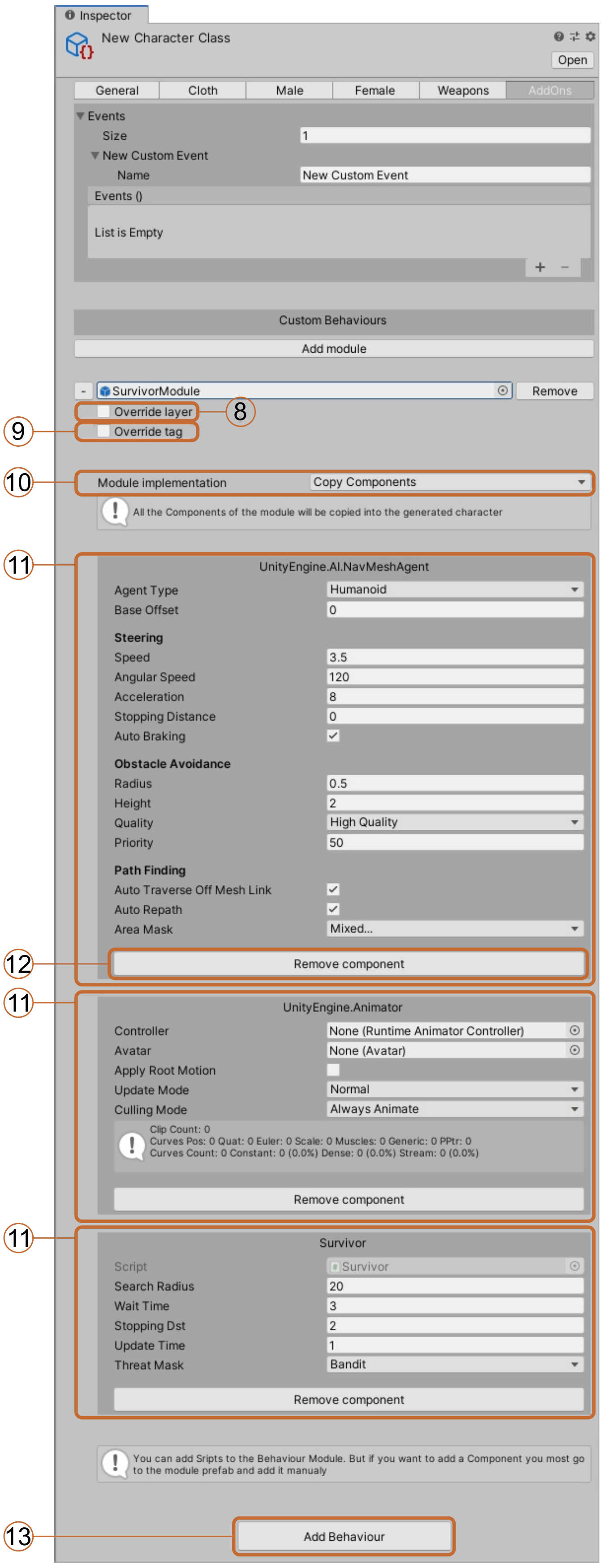
CHARACTER CLASS

ADD ONS - CUSTOM BEHAVIOUR MODULES

Components and scripts predefined by the user can be added to the generated characters by means of behavior modules (Custom Behavior Module). These modules will be prefabs where all the necessary components for the operation of the character will be contained. With this option it is possible to configure the generated characters en masse. Each Character Class can contain different behavioral modules that will be selected randomly when generating the character.

4. **Add module**: Add new behavior module.
5. **Fold / Unfold**: Shows or hides all the components within the module allowing their editing in real time without having to change the object.
6. Behavior module loaded.
7. **Remove**: Remove the behavior module from the Character Class.
8. **Override layer**: Allows you to select a layer that will overwrite the default layer selected in the General Settings.
9. **Override tag**: Allows you to select a tag that will overwrite the default tag selected in the General Settings.
10. **Module implementation**: In each module you will be able to define how it interacts with the generated character for greater compatibility with other plugins and to achieve the most automated result possible. There are three preconfigured ways in which modules can interact with characters.

    - **Parent container**: The module will become the parent of the character in the hierarchy. When this option is selected and the module contains more child objects, you will be able to select if you want the character to be a child of one of those objects or only of the module, leaving this space null. In addition, if the module contains an Animator Component, the option to use this as Animator by default will be shown, allowing all the Animator configurations of the character to be transferred to those of the module.
    - **Copy components**: Copy all components with their current module settings to the character.
    - **Copy hierarchy**: Copies not only the components of the module with their current configurations, but also transfers all the child objects with their components and current configurations to the character's hierarchy.
11. View of the components contained in the module.
12. **Remove component**: Removes the displayed component from the module.
13. **Add Behavior**: Add behavior. It allows adding scripts to the module. Only scripts can be added if a component needs to be added, in that case it will be necessary to go to the module's prefab and add it manually. It is recommended not to add Collider components as these are configured on the instantiators and could lead to component duplication. If it is necessary to add one, special care should be taken not to activate the option to add a component in the instance where the Character Class is used.



Inspector
New Character Class
Open
General | Cloth | Male | Female | Weapons | AddOns

Events
Size: 1
New Custom Event
Name: New Custom Event
Events ()
List is Empty

Custom Behaviours
Add module
SurvivorModule — Remove
Override layer
Override tag
Module implementation: Copy Components
All the Components of the module will be copied into the generated character

UnityEngine.AI.NavMeshAgent
Agent Type: Humanoid
Base Offset: 0
Steering
Speed: 3.5
Angular Speed: 120
Acceleration: 8
Stopping Distance: 0
Auto Braking: ✓
Obstacle Avoidance
Radius: 0.5
Height: 2
Quality: High Quality
Priority: 50
Path Finding
Auto Traverse Off Mesh Link: ✓
Auto Repath: ✓
Area Mask: Mixed...
Remove component

UnityEngine.Animator
Controller: None (Runtime Animator Controller)
Avatar: None (Avatar)
Apply Root Motion
Update Mode: Normal
Culling Mode: Always Animate
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%) Dense: 0 (0.0%) Stream: 0 (0.0%)
Remove component

Survivor
Script: Survivor
Search Radius: 20
Wait Time: 3
Stopping Dst: 2
Update Time: 1
Threat Mask: Bandit
Remove component

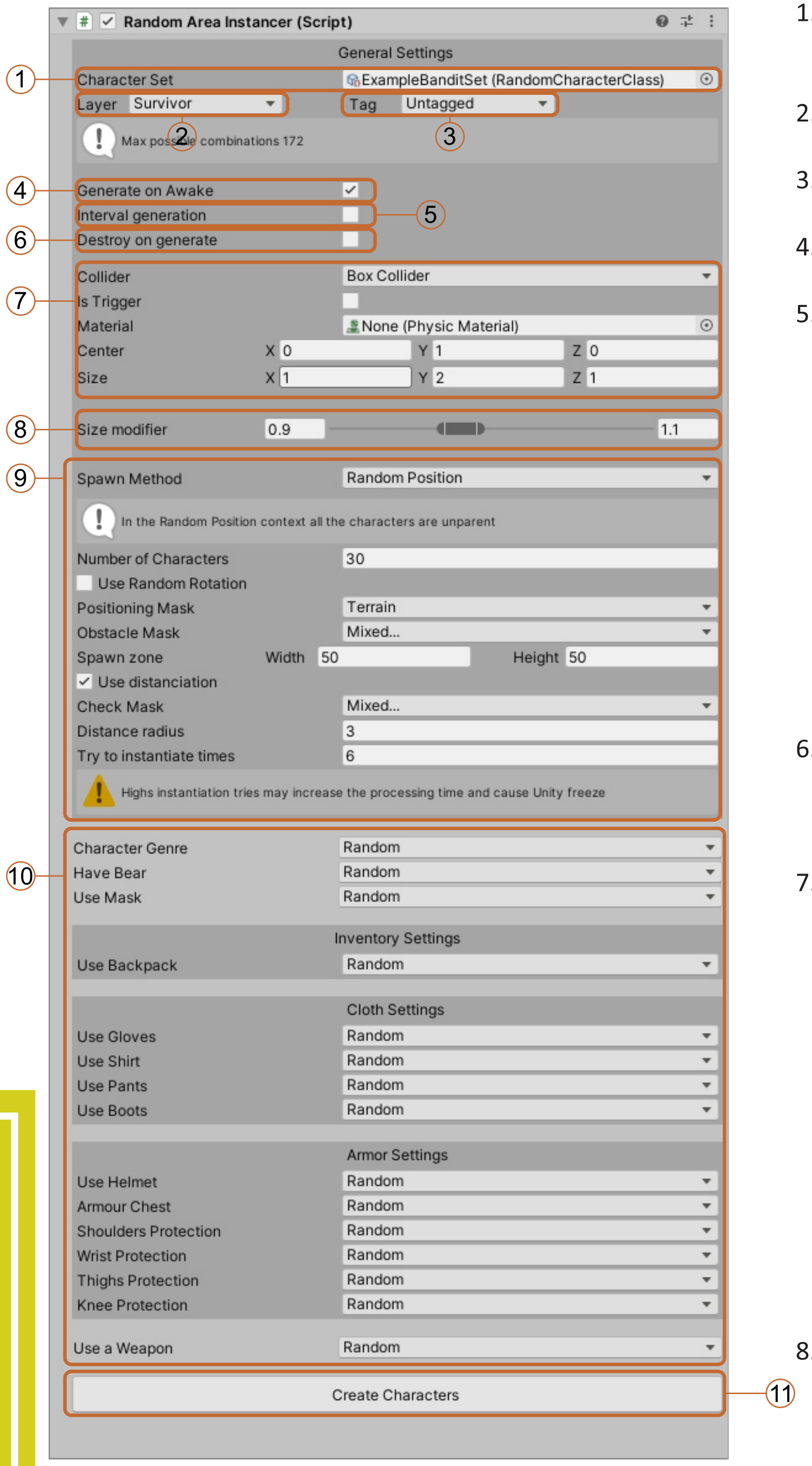
You can add Sripts to the Behaviour Module. But if you want to add a Component you most go to the module prefab and add it manualy

Add Behaviour

RANDOM AREA INSTANCER

RANDOM AREA INSTANCER

Area Instancer. Allows instances of a certain number of characters in a given area or in fixed assigned positions. A RandomCharacterInstancer can be created by right clicking on the Hierarchy tab and then 3D Object> MASSIVE> Single Instancer or GameObject> 3D Object> Massive> Area Instancer menu bar.



Random Area Instancer (Script)

General Settings

Character Set: ExampleBanditSet (RandomCharacterClass)
Layer: Survivor
Tag: Untagged
Max possible combinations 172

Generate on Awake: ✓
Interval generation:
Destroy on generate:

Collider: Box Collider
Is Trigger:
Material: None (Physic Material)
Center: X 0 Y 1 Z 0
Size: X 1 Y 2 Z 1

Size modifier: 0.9 – 1.1

Spawn Method: Random Position
In the Random Position context all the characters are unparent
Number of Characters: 30
Use Random Rotation:
Positioning Mask: Terrain
Obstacle Mask: Mixed...
Spawn zone: Width 50 Height 50
Use distanciation: ✓
Check Mask: Mixed...
Distance radius: 3
Try to instantiate times: 6
Highs instantiation tries may increase the processing time and cause Unity freeze

Character Genre: Random
Have Bear: Random
Use Mask: Random

Inventory Settings
Use Backpack: Random

Cloth Settings
Use Gloves: Random
Use Shirt: Random
Use Pants: Random
Use Boots: Random

Armor Settings
Use Helmet: Random
Armour Chest: Random
Shoulders Protection: Random
Wrist Protection: Random
Thighs Protection: Random
Knee Protection: Random

Use a Weapon: Random

Create Characters

1. **Character set**: Character class. Container from which values and objects are taken for the generation of the character.
2. **Layer**: The layer that will be assigned to the character once generated.
3. **Tag**: The tag that will be assigned to the character once generated.
4. **Generate on Awake**: Generate the characters when the scene starts.
5. **Interval generation**: Interval generation.
    a. **Start delay**: Initial delay. Time it takes for the instantiator to start instantiating.
    b. **Generation interval**: Generation interval. Waiting time between instantiating one character and the next.
    c. **Number of characters**: Number of characters that this instnciador can instantiate before being destroyed. If this value is left at 0, the instantiator will instantiate characters infinitely
6. **Destroy on generate**: After the generation is done, the instantiator will be destroyed on the scene. It does not work for interval generation.
7. **Collider**: Type of Collider component that will be added to the generated character.
    a. **None**: No Collider will be added to the character.
    b. **Capsule Collider**: A Capsule Collider component will be added to the generated character.
    c. **Box Collider**: A Box Collider component will be added to the generated character.
    After selecting one of these options, the configuration options will be displayed depending on the selected Collider.
8. **Size modifier**: Scale modifier for the generated characters. Rank by which the scale of the generated character will be determined.
9. **Spawn Method**: Instantiation method that will be used to place the characters in the scene.
    a. **Fixed position**: The characters will be instantiated in a fixed position determined by a list of positions (Transform).
    b. **Random position**: Characters will be placed randomly within a certain area.
10. Character customization options. They are the options by which the algorithm will design the characters.
11. **Create characters**: Generate the characters in the scene. If you don't want to use the Generate on Awake or Interval Generation option you can instantiate the characters in the scene using this button.

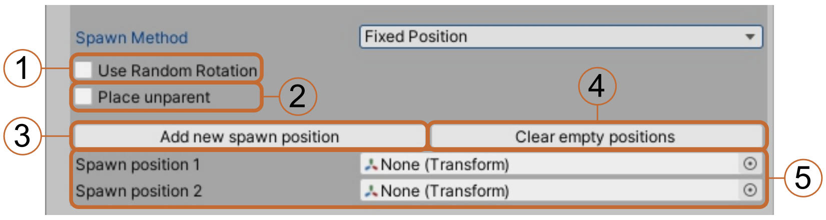
RANDOM AREA INSTANCER

SPAWN METHODS

FIXED POSITIONS

It allows the instantiation of characters in fixed positions in the scene.

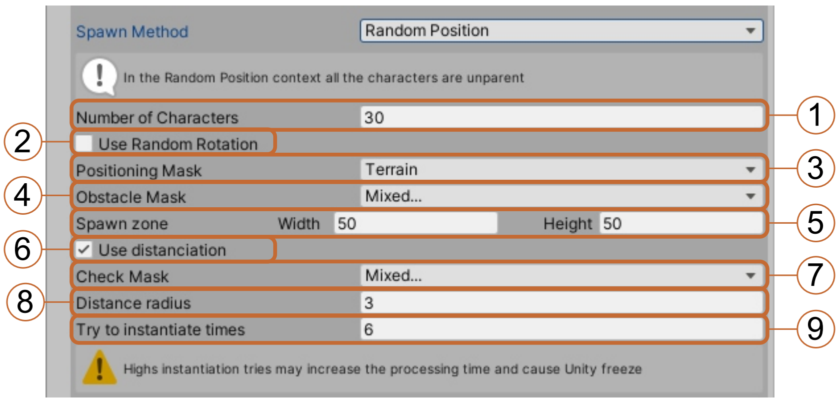
1. **Use random rotation**: Normally in this option the character will take the rotation of the instantiation point. When this option is enabled that rotation will be overwritten by a random rotation.
2. **Place unparent**: The character will be instantiated outside the hierarchy of the instantiation point. If disabled, the character will be instantiated as a child object of the position.
3. **Add new spawn position**: A new space is added for a new instantiation point. This field must be filled with a GameObject from the scene. Its position and rotation will be taken from this object.
4. **Clear empty positions**: Clears the list of empty positions instantiation positions.
5. List of instantiation positions.


Spawn Method: Fixed Position
Use Random Rotation
Place unparent
Spawn position 1: None (Transform)
Spawn position 2: None (Transform)

RANDOM POSITIONS

Instance the generated characters in random positions within an area in the scene. A RandomAreaInstancer can be created by right-clicking on the Hierarchy tab and then 3D Object> MASSIVE> Area Instancer or on the GameObject> 3D Object> Massive> menu bar. Area Instancer.

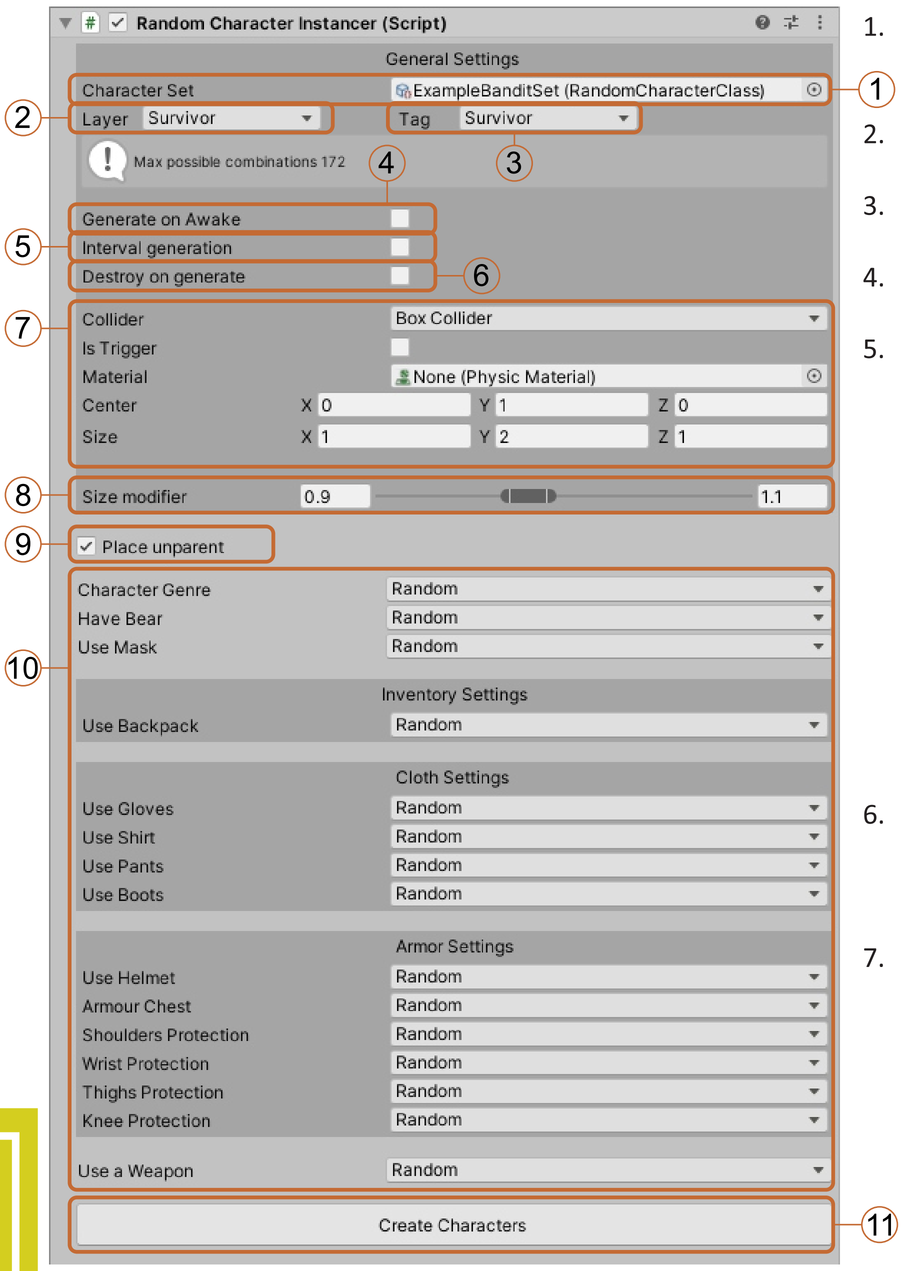
1. **Number of characters**: The number of characters that will be instantiated in a single hit in the scene as long as the instantiation option per interval is not activated.
2. **Use random rotation**: All characters in the scene will be instantiated in the normal rotation of the global scene system. Activating this option varies the rotation in a random value.
3. **Position mask**: Terrain on which the character can be placed.
4. **Obstacle mask**: Objects and terrain that represent an obstacle for the placement of the generated character in the scene.
5. **Spawn zone**: Zone in which the characters will be instantiated.
6. **Use distancing**: Use distancing. The use of distancing means that the system will check that the generated characters do not overlap in a certain radius. For this function to take effect, the collider must be added to the character and the character layer must be active in the distancing mask.
7. **Check mask**: Checking mask or distancing mask. Masks of objects on which the overlap will be checked.
8. **Distance radius**: Distance that must exist between the generated character and the objects that must be distanced from in order to instantiate the character in that position.
9. **Try to instantiate times**: Every time the system checks a position and it does not meet the selected requirements, the position will be discarded and the system will try to move to the next character without having placed the current one on the scene. This option warns the system by indicating the number of attempts to make to find a valid position for each character. This option should be a reasonable number, too high values can cause long check cycles and cause a temporary stop of Unity.


Spawn Method: Random Position
In the Random Position context all the characters are unparent
Number of Characters: 30
Use Random Rotation
Positioning Mask: Terrain
Obstacle Mask: Mixed...
Spawn zone: Width 50, Height 50
✓ Use distanciation
Check Mask: Mixed...
Distance radius: 3
Try to instantiate times: 6
Highs instantiation tries may increase the processing time and cause Unity freeze

RANDOM SINGLE INSTANCER

RANDOM CHARACTER INSTANCER

The simplest form of instantiation. It is based on the instantiation of a character in the same position. A RandomCharacterInstancer can be created by right clicking on the Hierarchy tab and then 3D Object> MASSIVE> Single Instancer or GameObject> 3D Object> Massive> Single Instancer menu bar.



Random Character Instancer (Script)

General Settings

Character Set	ExampleBanditSet (RandomCharacterClass)
Layer	Survivor
Tag	Survivor

Max possible combinations 172

Generate on Awake	☐
Interval generation	☐
Destroy on generate	☐
Collider	Box Collider
Is Trigger	☐
Material	None (Physic Material)
Center	X 0 Y 1 Z 0
Size	X 1 Y 2 Z 1
Size modifier	0.9 – 1.1
Place unparent	✓
Character Genre	Random
Have Bear	Random
Use Mask	Random

Inventory Settings

Use Backpack	Random

Cloth Settings

Use Gloves	Random
Use Shirt	Random
Use Pants	Random
Use Boots	Random

Armor Settings

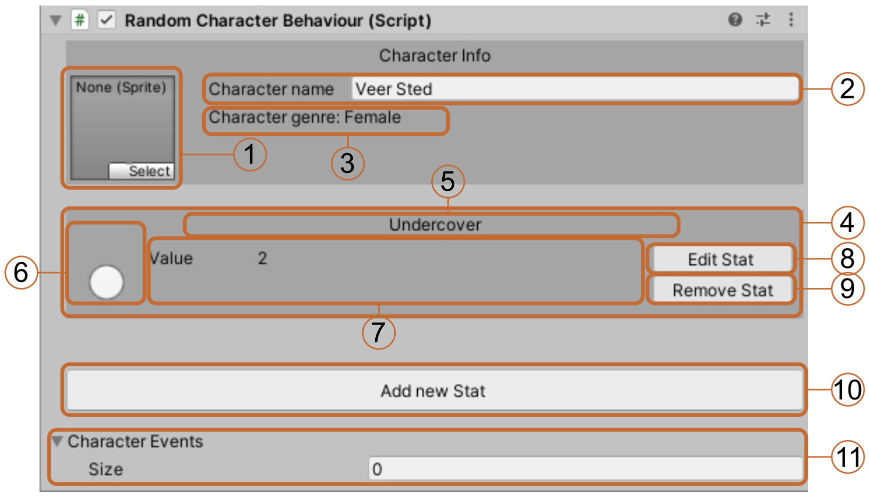
Use Helmet	Random
Armour Chest	Random
Shoulders Protection	Random
Wrist Protection	Random
Thighs Protection	Random
Knee Protection	Random
Use a Weapon	Random

Create Characters

1. **Character set**: Character class. Container from which values and objects are taken for the generation of the character.
2. **Layer**: The layer that will be assigned to the character once generated.
3. **Tag**: The tag that will be assigned to the character once generated.
4. **Generate on Awake**: Generate the characters when the scene starts.
5. **Interval generation**: Interval generation.
   a. **Start delay**: Initial delay. Time it takes for the instantiator to start instantiating.
   b. **Generation interval**: Generation interval. Waiting time between instantiating one character and the next.
   c. **Number of characters**: Number of characters that this instnciador can instantiate before being destroyed. If this value is left at 0, the instantiator will instantiate characters infinitely
6. **Destroy on generate**: After the generation is done, the instantiator will be destroyed on the scene. It does not work for interval generation.
7. **Collider**: Type of Collider component that will be added to the generated character.
   a. **None**: No Collider will be added to the character.
   b. **Capsule Collider**: A Capsule Collider component will be added to the generated character.
   c. **Box Collider**: A Box Collider component will be added to the generated character.
   After selecting one of these options, the configuration options will be displayed depending on the selected Collider.
8. **Size modifier**: Scale modifier for the generated characters. Rank by which the scale of the generated character will be determined.
9. **Place unparent**: The character will be instantiated out of the instantiation point hierarchy. If this is unabled the object will be instantiated as a child object of the current position.
10. Character customization options. They are the options by which the algorithm will design the characters.
11. **Create characters**: Generate the characters in the scene. If you don't want to use the Generate on Awake or Interval Generation option you can instantiate the characters in the scene using this button.

CHARACTER BEHAVIOUR

RANDOM CHARACTER BEHAVIOUR


Random Character Behaviour (Script)
Character Info
None (Sprite)
Select
Character name: Veer Sted
Character genre: Female
Undercover
Value 2
Edit Stat
Remove Stat
Add new Stat
Character Events
Size 0

In the editor the component will be shown with the following elements.

1. Image of the character.
2. Name of the character. It can be edited for this character if desired.
3. Gender of the character. It is not editable.
4. Fields with the character's statistics.
5. Name of the statistic.
6. Image of statistics.
7. Statistical values.
8. **Edit Stat**: Edit the statistic. Allows you to edit the statistics for this characters in case you don't like the generated values.
9. **Remove stat**: Remove this stat for the character.
10. **Add new stat**: Display a new field to add a new statistic to the selected character. This field has two buttons, one to accept the values and finally add the statistic to the character and another to discard this statistic.
11. **Character events**: The events that the character contains. They are what the Character Class happens to them when generating the character.

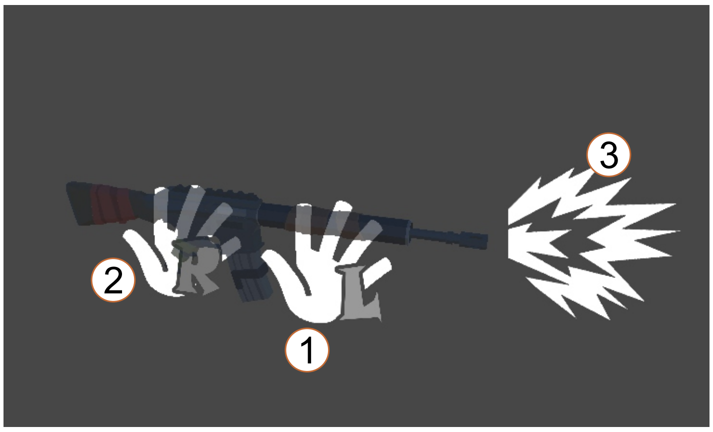
Each generated character will contain a component called RandomCharacterBehaviour in this component all the information, statistics and events defined in the CharacterClass will be located.
Public methods:

- public void InvokeEvent (string eventName)
  Summons an event on the character.
  eventName: Name of the event that is called to invoke.

- public CharacterStat GetStat (string statName)
  Locate a stat within the character.
  statName: Name of the statistic to search for.

WEAPON CONTROLLER

Weapon controller. This controller is a base class from which you can derive the other classes you decide to create. This class only contains the elements necessary to perform the inverse kinematics of the hands and a reference point to instantiate shots in case the weapons are ranged. It can be added to any weapon. At the moment of being loaded it will automatically instantiate these points in the parent object of the hierarchy.



Public variables:

1. **L_HandPosition**: Left hand position.
2. **R_HandPosition**: Right hand position.
3. **shootPoint**: Shooting point for ranged weapons.