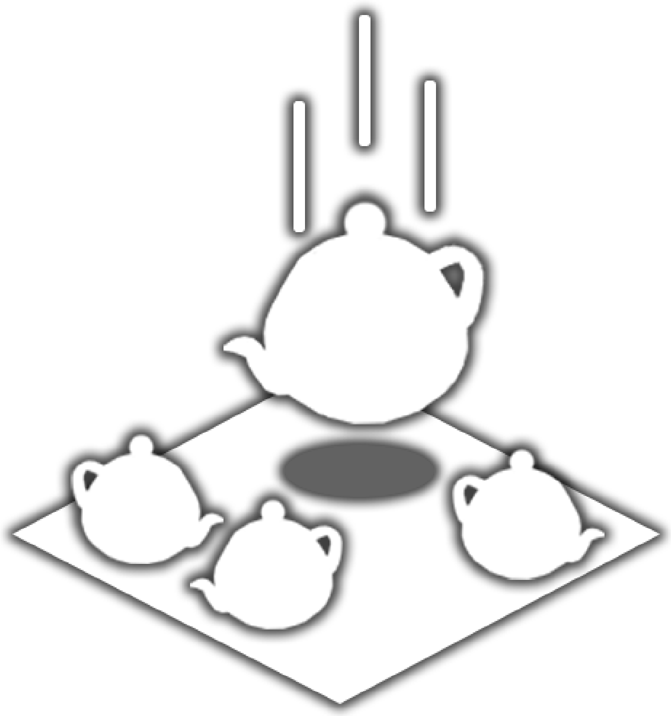


Object Distributor – Manual

Created By
Nick Alves – www.nickalves.com – @NotAdvisable

v2018.03.18

Table of Contents

Introduction 1

General Usage 2

Placement Methods 4

- True Random 4
- Avoid Edges 4
- Brightness Texture Based 5

Introduction

Hi there and welcome to the marvellous world of tool documentation!

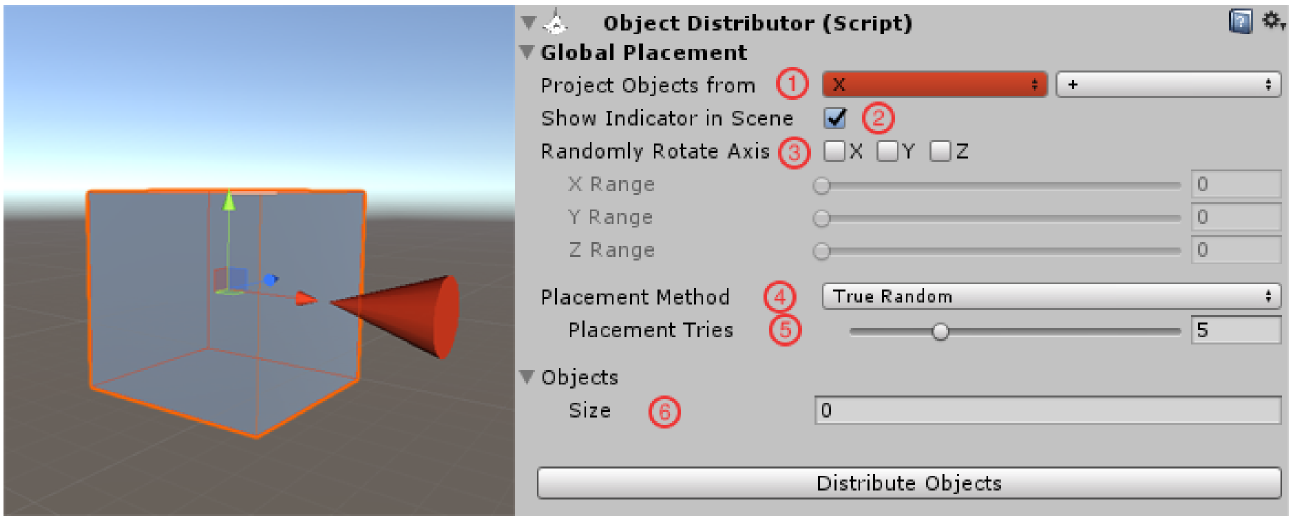
This tool is about placing stuff on stuff. It tries to randomly find a position on another object/surface and places the desired object aligning it to the surface's normal.
It works with anything that has at least a renderer component or a collider component and therefore works with normal objects and Unity terrains.
There are plenty of settings you can set like random rotation or different placement methods which I will explain in detail on the following pages.
I also included a couple of examples including self-made models and textures you can play around with.
Generally this documentation should cover most questions that might occur, though if there is anything else, feel free to contact me.

And now, written in 4000 teapots on a giant plane using this:

General Usage

Using this tool is pretty straightforward. After adding the package to your project, you'll find the necessary script under ObjectDistributor/Scripts. Now just add the ObjectDistributor.cs to any GameObject you want to distribute other Objects on.

These should be the setting you see when you first add the script to an object:



Object Distributor (Script)

Global Placement

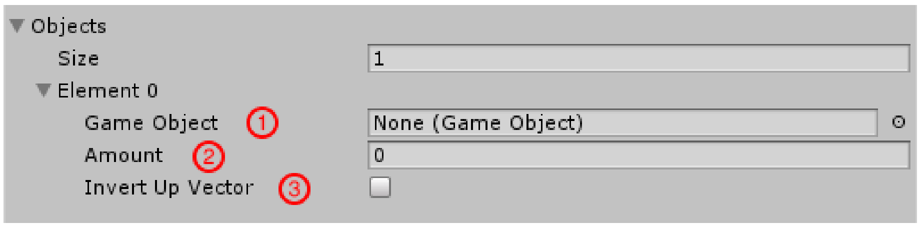
Project Objects from	1	X / +
Show Indicator in Scene	2	☑
Randomly Rotate Axis	3	☐X ☐Y ☐Z
X Range		0
Y Range		0
Z Range		0
Placement Method	4	True Random
Placement Tries	5	5

Objects

Size	6	0

1. Select the world space axis you want to place object from.
2. When set, the tool will display a cylinder from the selected direction and a wireframe box indicating the objects bounds.
3. Here you can set random rotations along any axis. Setting any value means that the object will be randomly rotated between the positive and the negative of that value.
4. Here you can select the method you want to use.
5. Since the placement happens randomly based on the object's bounds, there is a chance that the script doesn't hit the object if there are holes in it or it's weirdly shaped. Here you can set how often the script should try to hit the object. There's a cap on it to safe performance, though usually 5 should more than suffice.
6. Set the number of unique objects you want to place.

When you increase the size of unique objects, you'll presented with more options to set on a per object basis.


Objects
Size: 1
Element 0
Game Object (1): None (Game Object)
Amount (2): 0
Invert Up Vector (3): ☐

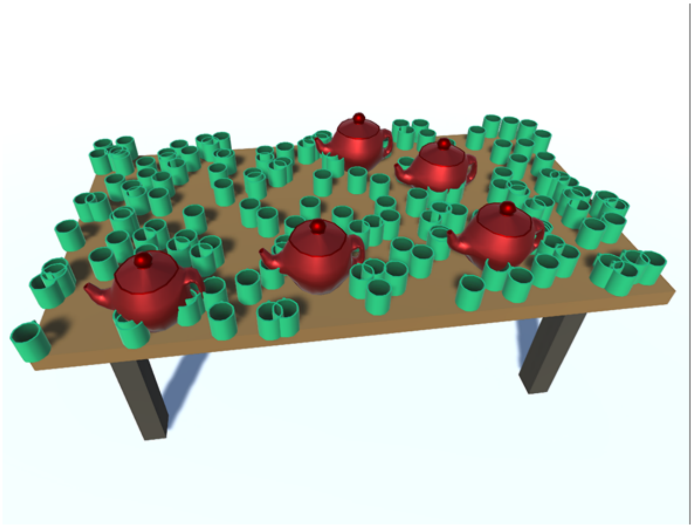
1. The gameObject you want to place.
2. How many of that particular gameObject you want to place.
3. If the up vector of that object should be inverted, effectively flipping the object upside down. I only ever needed this option for meshes that were intended to be used in another engine, but I wanted to keep that option in case it happens with any of your objects.

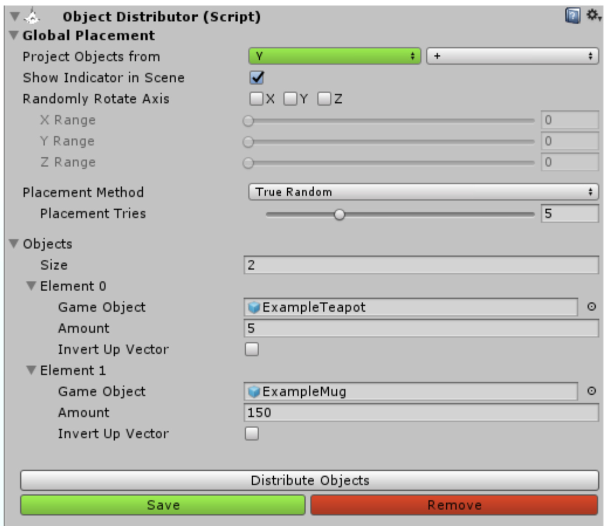
Finally, Just hit the “Distribute Objects” button until you like the random placement you see. You then also have the option to save the placement and add more on top of it.

Placement Methods

So far, there are three methods you can choose from. I will describe them a bit and show a few (maybe a bit excessive) examples to show their behaviours.

True Random



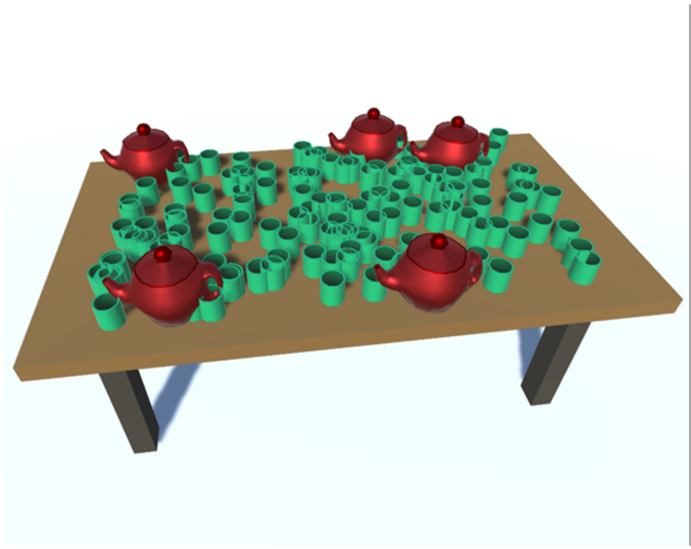


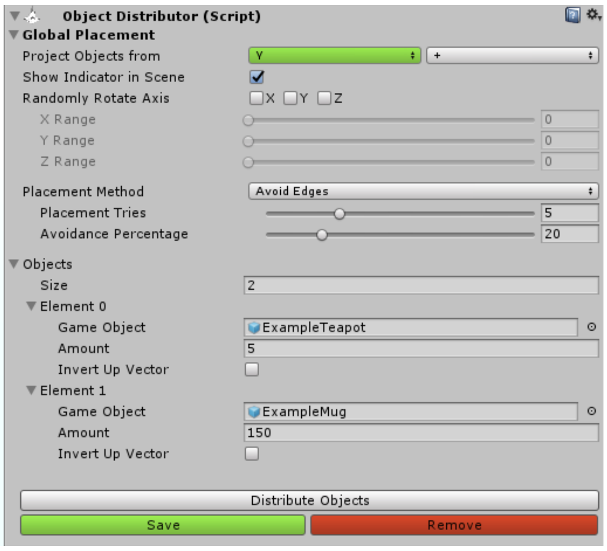
```
Object Distributor (Script)
Global Placement
  Project Objects from        Y          +
  Show Indicator in Scene     ☑
  Randomly Rotate Axis        ☐X ☐Y ☐Z
    X Range                              0
    Y Range                              0
    Z Range                              0
  Placement Method            True Random
    Placement Tries                      5
Objects
  Size                        2
  Element 0
    Game Object               ExampleTeapot
    Amount                    5
    Invert Up Vector          ☐
  Element 1
    Game Object               ExampleMug
    Amount                    150
    Invert Up Vector          ☐
```

The most basic method.

Objects are placed on the target completely by chance. Though this means that objects might overlap or are close to the edge.

Avoid Edges

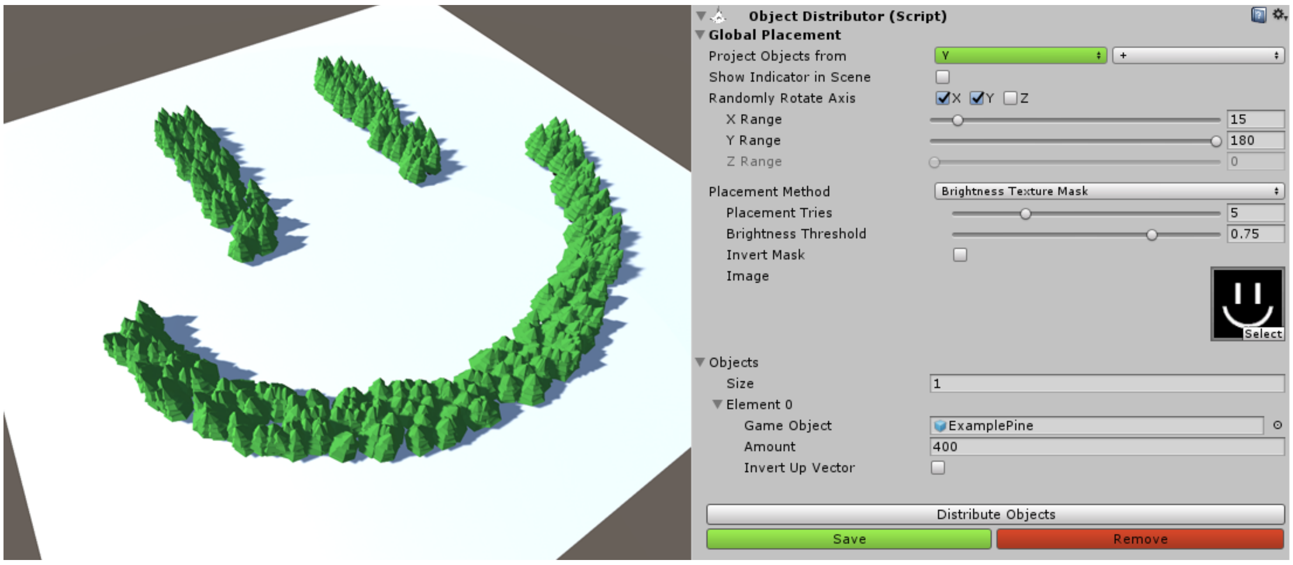




```
Object Distributor (Script)
Global Placement
  Project Objects from        Y          +
  Show Indicator in Scene     ☑
  Randomly Rotate Axis        ☐X ☐Y ☐Z
    X Range                              0
    Y Range                              0
    Z Range                              0
  Placement Method            Avoid Edges
    Placement Tries                      5
    Avoidance Percentage                 20
Objects
  Size                        2
  Element 0
    Game Object               ExampleTeapot
    Amount                    5
    Invert Up Vector          ☐
  Element 1
    Game Object               ExampleMug
    Amount                    150
    Invert Up Vector          ☐
```

This method pretty much functions the same way the first one does, the difference being that it avoids edges by a settable percentage.

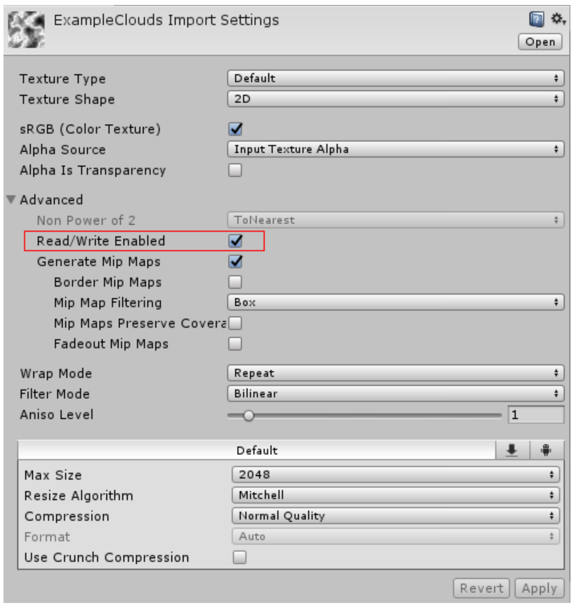
Brightness Texture Based



```
Object Distributor (Script)
Global Placement
  Project Objects from        Y   +
  Show Indicator in Scene     ☐
  Randomly Rotate Axis        ☑X ☑Y ☐Z
    X Range                   15
    Y Range                   180
    Z Range                   0
  Placement Method            Brightness Texture Mask
    Placement Tries           5
    Brightness Threshold      0.75
    Invert Mask               ☐
    Image
Objects
  Size                        1
  Element 0
    Game Object               ExamplePine
    Amount                    400
    Invert Up Vector          ☐
```

This method is certainly the most exciting one since it enables you to place objects based on custom textures. They don’t have to be black and white, though the script samples the brightness value of each pixel so they might as well be. There are three additional settings you get:

- Brightness Threshold: This is the cut-off brightness value which the pixels have to pass in order to be considered for placement.
- Invert Mask: By default, the script tries to place objects in bright areas. Inverting the mask means that it will try to place them in dark areas instead without you having to invert the texture manually.
- Image: The texture you want to sample from.



```
ExampleClouds Import Settings
Texture Type                  Default
Texture Shape                 2D
sRGB (Color Texture)          ☑
Alpha Source                  Input Texture Alpha
Alpha Is Transparency         ☐
Advanced
  Non Power of 2              ToNearest
  Read/Write Enabled          ☑
  Generate Mip Maps           ☑
    Border Mip Maps           ☐
    Mip Map Filtering         Box
    Mip Maps Preserve Covera  ☐
    Fadeout Mip Maps          ☐
Wrap Mode                     Repeat
Filter Mode                   Bilinear
Aniso Level                   1
Default
Max Size                      2048
Resize Algorithm              Mitchell
Compression                   Normal Quality
Format                        Auto
Use Crunch Compression        ☐
```

IMPORTANT: Read/Write has to be enabled in the texture’s import settings.

With this method you can write texts using objects, or distribute them more organically based on, for example, Perlin noise.