# MeshBrush – Documentation

## Index





Need additional help? meshbrush@glitchedpolygons.com

## What is MeshBrush, and how do I get started?

MeshBrush allows you to paint meshes onto your GameObject's surfaces.

To start painting, you need to select the GameObject on top of which you want to paint meshes in the scene, then go to the GameObject menu above in the toolbar and click on "Paint meshes on selected GameObject" inside the MeshBrush menu.

Note that the script doesn't work (unless in global painting mode) if you don't have a collider on your GameObject, so make sure you have one attached.
If you don't have a collider attached, MeshBrush will ask you if you want to add a MeshCollider now. You can either say *yes please* and have a mesh collider attached, or if you want some other type of collider on your mesh you can say no and add one by yourself.

Now that you have added the MeshBrush script to your object, you will be able to hover over your GameObject with your mouse and already see the actual circle brush appear on your collider's surface.

To paint meshes, add the desired prefabs to the set of meshes to paint inside the inspector and start painting (the default key for painting is “P”).

For more detailed information about the usage and configuration of MeshBrush, make sure to check out the other chapters in this documentation. Also keep in mind that almost all fields in the MeshBrush inspector have a tool-tipped label, the source code is well documented and easy to read and there’s even a quick help section at the top of the inspector.

If after reading this documentation you still need assistance with the configuration and/or the usage of MeshBrush, feel free to contact me directly (*meshbrush@glitchedpolygons.com*). Also contact me if
you have any requests, suggestions, feedback, feature recommendations or bug reports, I always try to answer and help asap.
User feedback is treated confidentially and will be used to make MeshBrush a better product. New ideas for features submitted by users are implemented as good as possible within two to three versions. It’s only thanks to you awesome Unity users out there and your feedback that I got so far and up to this point with this asset :)

## Groups

Each MeshBrush instance can be named individually, such that the painted meshes can be further organized via these group names (the defined group name is also the name of the parent holder GameObject).

There is always one group per MeshBrush component, so if you want to have multiple groups, just add more than one MeshBrush instance to your object. The name of the

group can be modified at the top of the component's inspector (the changes will take place immediately).

Multiple groups per object means multiple brushes: you can help yourself and assign different colors to the various brushes, this way you can differentiate between them when painting. You can disable a group by disabling its MeshBrush instance via the corresponding "Enabled" toggle in the inspector.

For instance: if you have a terrain mesh with two MeshBrush instances on it (one that paints meshes from a set of stones, the other one paints bushes), you can disable the stones group and only paint bushes. It's that easy!

You can also collapse a MeshBrush instance with the "Collapsed" toggle; it cleans up the inspector window a lot after configuring your painting setup.

- **Note:**
  It is not recommended to have multiple groups on the same object sharing an identical name, as that can cause unexpected behaviours.

## Global painting mode

Since V1.5 there is a global painting mode, with which you can make use of MeshBrush without a collider. You can just paint your meshes directly into the scene. So if you choose this way over the traditional MeshBrush component way, just open up the GameObject menu and under MeshBrush click on "Global painting".

This will automatically place a MeshBrush entity set up for global painting in the scene. You'll notice that the inspector is different compared to the traditional MeshBrush: there is a "layer based painting" section on the top.

When painting meshes globally into the scene, you can still have control over where you want your meshes to be placed and where absolutely not to. You achieve this with layers: just deselect all the layers in your global mesh painting inspector where you don't want your meshes to end up.

## Templates

MeshBrush V1.5 came with a brand new template system. It allows you to save your favourite MeshBrush settings into useful template files that you can later reuse in your scenes and projects. So if you find one of your MeshBrush configurations particularly worth saving, just do so by clicking on the save disk icon in the templates menu inside the inspector. You can then choose where to save your template file: it can be stored anywhere inside your project folder.

To load a template, press the load button right next to the save button and select the template file you want to load up. The current settings will be overwritten by the loaded template.

You can even save your favourite templates inside the favourites list in the inspector. Note that deleting, renaming or moving a template that has been added to the favourites list can break the connection between the two, and you'll have to reassign the path in the inspector. You can click on the three dots next to the entry in the favourites list to do so.

- **Note:**
  In V1.9, the entire template system has been rewritten from scratch to make the exported template files more human-readable, maintainable and easier to parse and work with in code.

If you have templates exported from previous versions of MeshBrush, don't worry! I wrote a migration utility for you to help you port your old, obsolete templates over to the new format under "*GameObject/MeshBrush/Migrate selected templates*". Your new templates will be placed in the same folder together with the old template files with a "*__migrated*" suffix in the name.

## Set of meshes to paint

The set of meshes to paint contains all the meshes that you want to paint with one brush, and can be found under the "Meshes" foldout in the inspector.

You can choose between the modern and classic UI setup. While the classic UI works exactly like a standard array, the modern UI offers drag 'n' drop functionality, intuitive controls as well as preview images of the meshes to paint.

All controls have detailed tooltips when you hover your cursor over them, so don't worry about the confusing amount of buttons at first glance: it's very simple actually.

To add a mesh to the set, you can either drag and drop the mesh/prefab directly into the blue plus icon and it will automatically be added to the set. If you don't know what mesh to add yet, you can also just click the blue plus icon and add an empty field to the set; you can then either drag and drop a prefab into the null field to assign it, or you can click on the null field and select an object from the object picker popup.

Preview icons controls:

- Left-mouse-clicking a preview icon will ping the linked prefab inside the project view (or show the object picker window if it's null).
- Middle-mouse-clicking a preview icon will erase the field (making it null).
- Nullifying a field will not remove it from the set.
- Right-mouse-clicking a preview icon will remove it from the set of meshes to paint.

- **Note:**
  To paint meshes, all fields inside the set have to be assigned. If you have null fields inside the set, you can't paint. To fix this, you can press "Clean unused fields" to clean the set from any empty fields.

## The precision placement mode

The precision placement mode allows for very accurate and reliable placement of one single mesh into the scene. Once active, the precision placement mode will override the default circle brush and show a preview mesh at the current mouse cursor's location.

This entire procedure is divided into 4 simple steps:

1. **Location**

   Here you can choose which mesh to place (you can cycle through the meshes in your set with the N and B keys; N selects the next mesh in your set, whereas B cycles backwards) and confirm the desired location of your mesh directly in the scene view by pressing the paint button.

2. **Scale**

   This step lets you adjust the scale of your mesh by dragging your cursor away from (and to) the object's center.

3. **Rotation**

   Drag your mouse cursor horizontally to adjust the object's rotation along the Y-Axis.

4. **Offset**

   Drag your mouse cursor vertically to adjust the object's offset along the Y-Axis. Confirming the vertical offset automatically places the mesh into the scene "as is", terminates the placement mode and returns back to the standard mesh painting mode.

If you don't want it to return back to the normal painting mode, just hold down "Shift" whilst confirming step 4: the precision placement mode is reset and you can place another mesh with it.

## Key bindings

Under the "Key bindings" foldout in the inspector you can customize your keyboard shortcuts for use inside the scene view. Watch out not to assign the same key to more than one action, as that would cause a conflict between the two.

Since V1.8 you can assign "Mouse0" to your key bindings, which makes painting and sampling reference vectors in the scene so much more pleasant!

If you're painting with your mouse/tablet, make sure to tick "**Lock screen view**" at the top of the inspector, such that you don't accidentally deselect your MeshBrush object whilst painting (the scene view needs to be in focus in order for MeshBrush to work correctly!).

## Brush settings

Inside the brush settings menu (under the "Brush" foldout) you can change the color of your circle brush, radius, scattering and many other useful parameters to get the result you're looking for.

- **Note**:
  All sliders can be either dragged as a whole along their handle rails, or split into a ranged slider that will define the minimum and maximum bounds within which a random value will be chosen each paint stroke.

If you feel limited by the range bounds of the sliders, you can modify some of them under the MeshBrush component's context menu (the gear icon in the top-right corner of the inspector)... just click on "Edit range limits" to modify the slider bounds.

- **Color**

  The color is useful to distinguish your brush from other brushes on the same GameObject, or to make it more visible inside the scene (e.g. a green brush on a terrain full of grass is hard to see).

- **Radius**

  The radius value is expressed in meters and determines the overall size of your brush area.

- **Constant mesh density vs. fixed quantity**

  You can choose between painting a fixed number of meshes inside the brush area and a density-dependent mesh quantity. When painting a fixed number of meshes, the resulting density heavily depends on the size of the brush: a huge

brush that paints only five meshes will result in a lot of free space between the painted meshes.
If you want a persistent mesh density across multiple brush sizes, tick the "Constant mesh density" toggle in the inspector. This will keep the density of your painted meshes homogeneous, even when painting with different brush sizes.
The mesh density unit is **meshes/m²**.
A constant mesh density, no matter how small or big your brush is, will always result in the same amount of space between your painted meshes. Check out the toggle label's tooltip for a more detailed explanation including an example.

- **Offset**

  The "Offset amount" offsets your painted meshes away from their underlying surface (or into it). This is useful if you have your painted meshes stuck inside your geometry, but in general this option is not needed if you place your pivot points correctly inside your modelling application.
  That's also why this value is intended for micro-optimizations only, being in centimeters instead of meters.

- **Scattering**

  The "Scattering" is a percentage value that defines how far away from the center of the circle brush the meshes to paint will be scattered.
  A value of 100% would allow the painted meshes to be scattered out to the edge of the circle brush, while a value of 10% would keep all the meshes dense in the center.

- **Delay**

  The "Delay" setting determines the minimum delay (in seconds) between paint strokes. It's used when holding down your paint button and defines the speed at which you are going to be placing meshes.

- **Y-Axis tangent to surface**

  As you decrease the slope influence value, you can choose whether you want your painted meshes to be kept upright along the Y-Axis, or tangent to their underlying surface.

  - Keep this off when aligning meshes with the brush stroke direction!

- **Align with stroke**

  With this option, you can align the painted meshes with the direction of the paint stroke.

- **Note:**
  The "Align with stroke" option only works when there is no "Random Y rotation" set up (meaning 0%), since the rotation randomizers are calculated after the original placement and orientation of the meshes.

## Slope settings

With the "Slope settings" foldout settings you can control how you want the slopes and angled surfaces in your scenes to affect your MeshBrush.
The "Slope influence" percentage defines how much influence slopes have on the rotation of the painted meshes. A value of 100% for example would adapt the painted meshes to be perfectly perpendicular to the surface beneath them, whereas a value of 0% would keep them upright at all times.

You can use the "slope filter" to avoid having meshes painted onto slopes and hills and other steep areas of your scene's topology.

Adjust the slope filter's "Angle threshold" value to define the maximum angle above (or below) which no meshes may be painted.
You can also paint exclusively on slopes and hills; just invert the filter by toggling the "Inverse slope filter" toggle in the inspector.

The default reference vector used to calculate the slope angle on your painted meshes for the slope filter is the world's Y up vector.
In most cases, this is what you'll need (on terrains and other "flat" surfaces... and by flat I mean aligned with the world's XZ plane). But there are a few scenarios in which you would want to use a different directional vector as a reference for the slope filter, like when you are painting onto round shaped meshes, e.g. planets and such, or simply painting diagonally/upside down.

In these scenarios, you can manually sample a reference vector directly from your GameObject's topology; to do so, just enable "Manual reference vector sampling" in the inspector and activate the vector sampling mode by clicking on the button on the right.

This will switch your brush to the reference vector sampling mode: in this brush mode you can't paint meshes, you can only sample a vector from your mesh by pressing the regular paint button, or cancel the sampling process by hitting the "Escape" key.

Deselecting and re-selecting the GameObject will also cancel the sampling mode.
If you need a reference vector based on multiple (averaged) normal vectors, hold down "Shift" while sampling to add more vectors.
Once you have sampled a new reference slope vector, it will appear in your scene view where you sampled it as a visual reference arrow (unless you don't want that... in that case just turn it off in the inspector via the " Show sampled vector" option toggle).
The slope filter will now use that vector instead of the standard world Y up axis.

All slope settings can be reset back to their default values immediately with the "Reset all slope settings" button in the inspector.

## Randomizers

The randomizers let you spice up your painted meshes a little bit; because nothing in nature comes without at least some degree of randomness…

You can randomize the scale property (uniformly and non-uniformly) as well as the rotation amount.

A uniform random scale sets the same value for all three axes (X, Y and Z), whereas a non-uniform scale allows for more detailed control over each axis individually.

- There is also a "Random scale curve" and a "Variation" field.

The "Random scale curve" is a curve that controls the scale of the painted meshes based on their distance to the brush circle's center.

- Time = 0 (left side of curve) translates to the circle brush's center.
- Time = 1 (right side of curve) is the outer edge of the circle brush.

The scale value of the painted meshes will be multiplied by the established curve value, meaning that if the curve evaluates to 1, then the scale will be unaffected by the curve. If it evaluates to 2, the (already randomized) scale will be doubled.

This can be useful if you want to achieve an effect like for example shrinking the painted meshes the closer they get to the brush circle's edge.
Check out the provided demo scene for a practical example!

"Variation" is the allowed error margin for the "Random scale curve" value (a random value will be picked between [-variation] and [+variation] and added to the evaluated curve value).
It's best to keep this value low (between 0 and maybe 0.2).
Setting this value too high would make the scale curve meaningless in the first place, since it would randomize the scale of the painted meshes in such a way that the effects of the curve driven scale aren't even visible anymore.

To affect the rotation of the painted meshes around their local X, Y and Z-axis, use the "Random rotation amount" slider (value is a percentage).

- E.g. a value of 0% around local-Y would always paint your meshes with the same default Y rotation, and a value of 100% will always spawn your meshes with a different (random) rotation around their Y axis.

Since V1.9 there is even a "Randomizer brush", with which you can randomize meshes that have already been painted. The default key for the randomizer brush stroke is "J". Properties that can be affected and thus modified by the randomizer brush are "Position", "Rotation" and "Scale".

## Overlap filter

MeshBrush V1.6 included a new filter that avoids your painted meshes to overlap with each other. It's been added under the name of "Overlap Filter", and can be found in MeshBrush's inspector right below the "Randomizers" foldout.

To activate the overlap filter, tick the "Use overlap filter" toggle in the inspector and make sure the "Minimum absolute distance" value is greater than zero. This number indicates what absolute minimum distance (in meters) shall be kept and maintained between your painted meshes. The distance is measured between the painted meshes' transforms. Meshes closer to each other than the allowed min. absolute distance value won't be painted.

- **Warning:**
  The overlap filter can be a pretty performance heavy feature, whose operations may slow down your PC when used in large groups of meshes.

To reduce the overhead whilst painting, it is recommended to only activate this feature when it is really strictly needed and in small groups (it's way better to have many small MeshBrush groups instead of one huge container group with all painted meshes under it whatsoever).

## Additive scale

You can also choose to add an additive scale to your mesh below in the inspector, this will add the value you define to the painted meshes after they have been randomized.
Additively shrinking your models is possible by simply typing in a negative number, which obviously can't (and shouldn't) go below -1... as we don't want to invert the scale of our painted meshes... That'd be weird :S

Like in the "Random scale" parameter, you can choose here too to scale uniformly on all three axes or along each axis individually.

## Optimization

When you are done painting meshes, you can optimize your scene by flagging what you've painted so far as static , or even by combining everything into one single mesh (per material ).
You can also combine only the meshes inside the circle brush area; to do so, just

press the combine button (default key is “K”) and all painted meshes inside the brush area will get combined.

Combining the meshes will discard all of their components except their MeshRenderer and MeshFilter, and automatically flag the combined mesh as static afterwards.

The combined mesh is stored by default inside the scene’s root, but you can move the mesh data to the project folder and store it there for later re-usage. When you combine the meshes, the new combined object gets automatically selected (you can turn this off via the inspector toggle at the bottom) and gives you the option to save the mesh to disk.

If you want to re-use your combined mesh in other scenes (e.g. via a prefab) you have to save it to disk first, and then you can make a prefab out of it.

- **Warning:**
  Combining meshes cannot be undone, so be careful with this button. Only use it when you are really, really 100% sure that you're done painting meshes on your GameObject (or in a specific area of your GameObject); consider it a final "optimizing touch" to your scene.