bake-wrangler-docs

Oct 16, 2021

Contents

Bake Wrangler is an add-on for blender that provides a *nodes based* interface for texture and material baking.

Current Features:

- Easy to use node system
- Background batch baking
- Supports all the blender internal bake passes and multires
- Many additional bake passes not found in blender for PBR and other work-flows
- Full set of post processing nodes for splitting, joining, mixing and mathing
- UDIM tile support and output
- Cage creation
- Productivity tools such as auto sorting of high to low poly models
- Basically everything you will want or need for texture baking

Come visit blenderartists.org/t/bake-wrangler-node-based-baking-tool-set/ to get involved and make suggestions :)

Bake Wrangler is available from:

Gumroad:	gum.co/bake-wrangler
Blender Market:	blendermarket.com/products/bake-wrangler

CHAPTER 1

Install & Update

There are no special steps required to install the add-on. Simply open your blender preferences, navigate to the *Add-ons* tab (left column) and click the '*Install...*' button near the top right. In the '*File View*' that opens navigate to the location of the Bake Wrangler zip file and click '*Install Add-on*' with it selected or double click on the zip file.

Once installed you need to check the box on the add-on to enable it.

Updating:

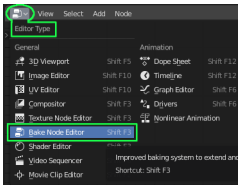
When updating follow the same procedure (making sure the '*Overwrite*' box is checked in the '*File Viewer*' when selecting the new version). Sometimes changes are made the require blender to be restarted, though I try to avoid this, it is a good idea to do so after updating if things don't seem to be working right.

Recipes from older versions of the add-on can be updated to the latest version by using the '*Update*' button in the right side ('N') panel found in the '*Bake Wrangler*' tab.

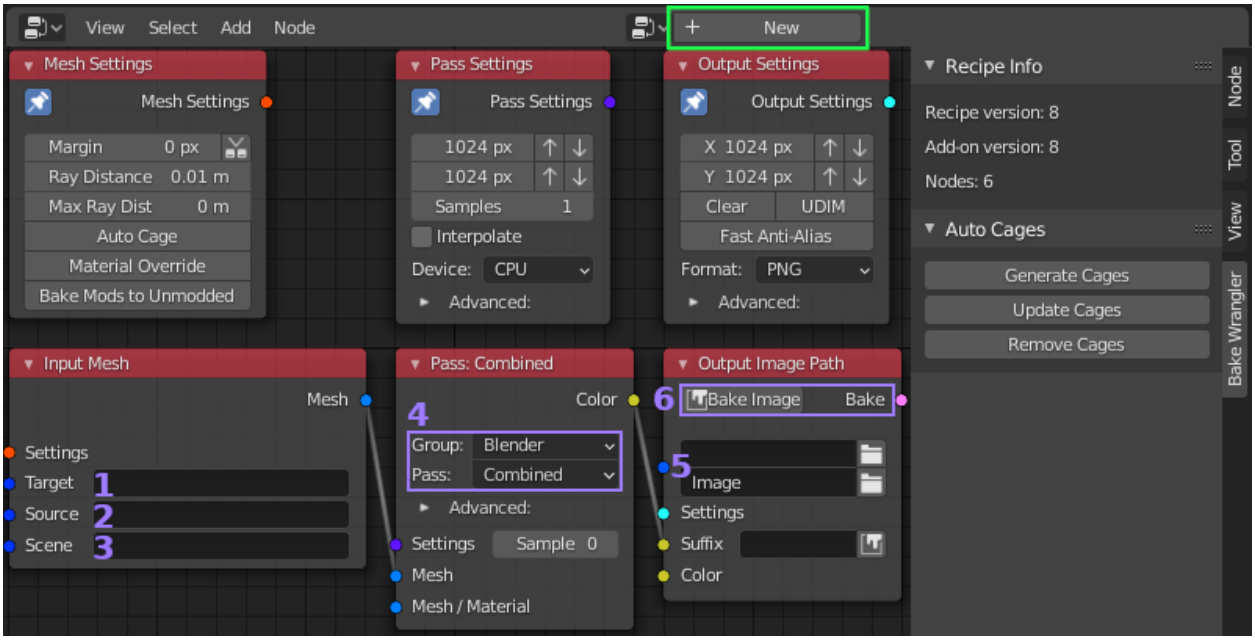
CHAPTER 2

Quick Start Guide

Once the add-on is installed and enabled a new '*Editor Type*' will become available (the drop down list in the top left title bar of every area) called '*Bake Node Editor*':



Switch to the *Bake Node Editor* and click '*New*' to create a recipe. Your recipe will start with the most simple bake node configuration loaded. This guide will cover this simple configuration, for details on each node take a look at the *Nodes Section* and for more complicated set ups hop on over to the *Examples*.



There are three primary node types: *Inputs, Passes and Outputs*. Each of these node types has a '*Settings*' socket and an associated node with settings for that part of the process. The *Pin* in the top left of a settings node causes it to be applied to all nodes with an empty '*Settings*' socket of that type.

Note that the side ('N') panel also has additional information and buttons. If your recipe was of an older version, an '*Update*' button would also appear here. Some nodes not shown here also display information in the side panel.

1. **Target:** is the object you want to **bake to**. It must be a '*MESH*' type object with a UV Map. A list of objects can also be connected here which would cause each object in the list to be baked with the same settings. This field must be filled.
2. **Source:** is the object you want to **bake from**. The field is optional and if left empty the data will be taken from the target object. Most renderable objects can be used as the target and a list of objects can be connected. If multiple objects are selected then **all** of them will be applied to each target (if there are multiple targets).
3. **Scene:** is used for including light emitters and shadow casters. It is important to note that some passes will not produce any output if lighting is not included. This is true of the standard blender passes *Combined, Diffuse, Glossy, etc*. Lighting could also be included by the '*World*' material which can be set under the advanced options in the '*Pass Settings*'.
4. **Group & Pass:** selects the type of data you want to bake. Some passes will display additional options below them when selected. In the example picture the '*Albedo*' pass in the '*PBR*' group has no additional options.
5. **Image Path & Name:** selects where the baked image will be saved. Relative locations are supported and will be expanded on bake (*Eg. Using // for the current path*). File extensions are added automatically to the name. There are more advanced options for splitting and naming the outputs not covered here.
6. **Bake Image Button:** will first validate the settings and then begin a background process to create the image. In the bottom right corner of Blenders status bar a 'BW' icon shows the current state: Green when the last operation completed successfuly, Blue when a background process is running and Red when the last process had an error. You can click on this icon to bring up a log which will update in real time. There are also preferences to control the log and have it automatically display when you start a bake.

Most of the other settings a fairly self explanatory and can often be left at their defaults. More node types and details of each setting are covered in the *Nodes Section*

CHAPTER 3

Bake Passes

Following is a list of currently supported bake passes and a short description of their function:

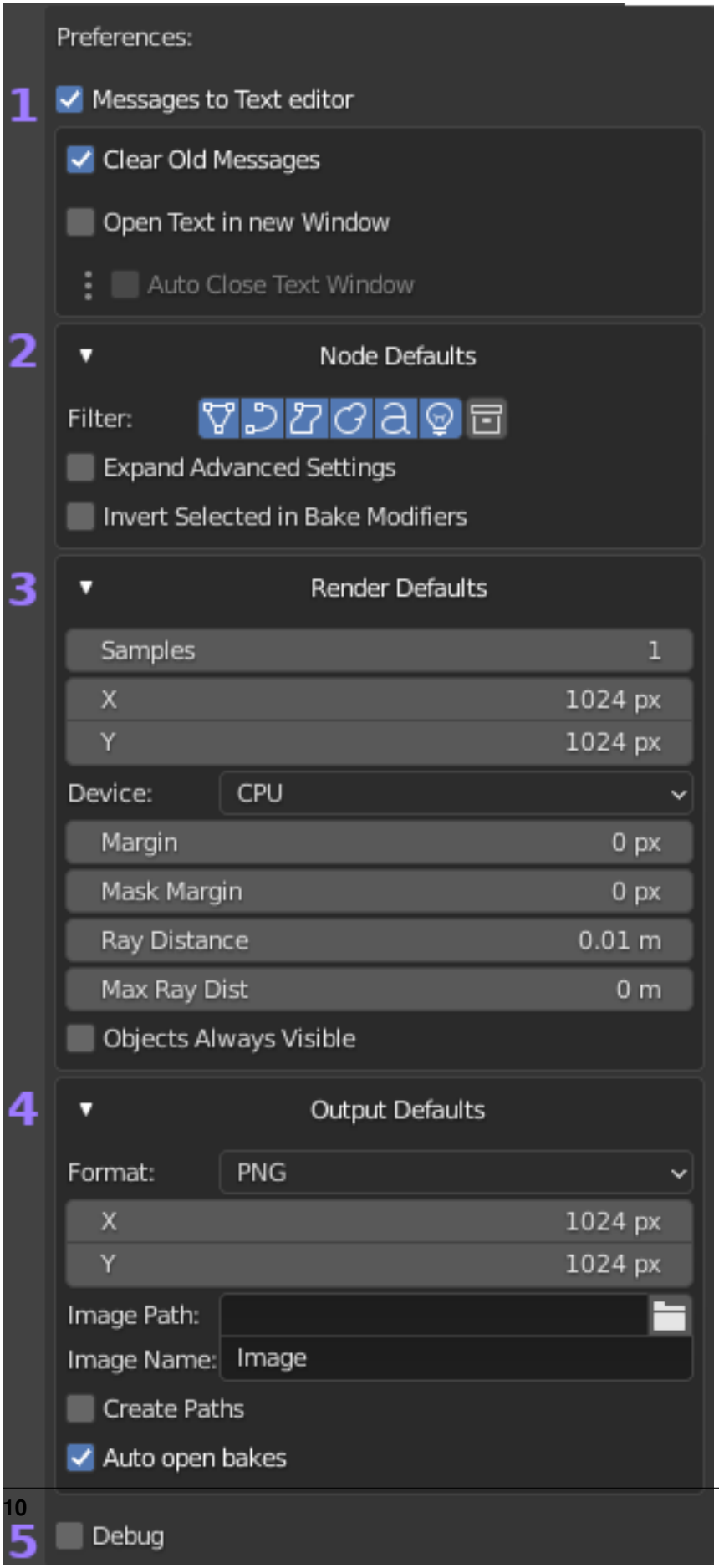
- **All Blender internal passes:** The standard passes and all their options are supported including the Multiresolution passes: Combined, AO, Shadow, Normal, UV, Roughness, Smoothness, Emit, Environment, Diffuse, Glossy, Transmission, Multiresolution Normals & Displacement.
- **Full set of PBR passes:** These passes all collect specific object information without influence from other inputs or lighting. The current list is: Albedo, Subsurface, Subsurface Radius, Subsurface Color, Metallic, Specular, Specular Tint, Roughness, Smoothness, Anisotropic, Anisotropic Rotation, Sheen, Sheen Tint, Clear Coat, Clear Coat Roughness, Clear Coat Normals, Transmission IOR, Transmission, Transmission Roughness, Emit, Alpha, Texture Normals (no geometry influence) and Object Normals (no texture influence).
- **Bevel Mask:** Generates a map with beveled areas masked in white.
- **Bevel Normals:** Creates a normal map with just the beveled areas (can be used with above mask to mix with other maps).
- **Cavity & Edges:** Used to get a grey scale map of cavities or edge locations.
- **Curvature:** Produces a grey scale map of surface angles with options for how values should be mapped.
- **Height Map:** Another grey scale map giving the distance between the target surface and other surfaces around it.
- **Island ID:** Outputs each UV island in a different flat color to differentiate them.
- **Material ID:** The areas covered by each material are output with a different color assigned to each material.
- **Object Color:** Simply outputs each object using the color assigned to it in the viewport settings.
- **World Position:** A tri-color vector map giving the location of the object from the origin.
- **Thickness:** Gets the distance between internal faces, representing how thick an area of the object is.
- **Vertex Colors:** Outputs whatever data is in the selected vertex colors block as a texture.

There is always the possibility of more passes being added by user request or additional features being adding to existing ones.

CHAPTER 4

Preferences

Below are the current available preferences and their default settings:


Preferences:
1
Messages to Text editor
Clear Old Messages
Open Text in new Window
Auto Close Text Window
2
Node Defaults
Filter:
Expand Advanced Settings
Invert Selected in Bake Modifiers
3
Render Defaults
Samples 1
X 1024 px
Y 1024 px
Device: CPU
Margin 0 px
Mask Margin 0 px
Ray Distance 0.01 m
Max Ray Dist 0 m
Objects Always Visible
4
Output Defaults
Format: PNG
X 1024 px
Y 1024 px
Image Path:
Image Name: Image
Create Paths
Auto open bakes
5
Debug

1. **Message Settings:** Deal with where and how information is reported back to the user about progress and errors.
    - *Messages to Text Editor*: Causes messages and errors produced by Bake Wrangler to be written to a text file named '*BakeWrangler*' in the current project. When disabled these messages would only be visible in the console. This also enabled the three options below and causes the BW icon to display in the bottom right of the status bar.
    - *Clear Old Messages*: Clears the text file prior to each bake, so that messages are only relevant to the current/last process.
    - *Open Text in new Window*: Will open a new window when a bake process starts, displaying the text file (which is continually updated). The size and location of the window will match the Bake Node Editor from which the bake was started. The intention is to allow you to create your own work space with the text file already open and so disable this pop-up.
    - *Auto Close Text Window*: Will close the above pop-up automatically when a bake completes successfully.
2. **Node Defaults:** Sets the default display state of some nodes. Here the initial filter settings of the *Objects* node can be set. You can also have nodes with collapsed advanced options start with them expanded instead. (Applies only to new nodes)
    - *Invert Selected in Bake Modifiers*: Causes the modifier selection method to be inverted when using this option in the Mesh node. Viewport hidden modifiers will be baked down instead of shown modifiers.
3. **Render Defaults:** Has settings related to the Mesh and Pass nodes, allowing you to configure their defaults.
    - *Objects Always Visible*: Causes Bake Wrangler to ignore the visibility settings of an object in Blender. When enabled all objects selected as part of a bake will be made visible.
4. **Output Defaults:** Has settings related to the Output Image Path node, allowing you to configure the defaults for newly created nodes.
    - *Create Paths*: Causes Bake Wrangler to attempt to create the output path if it doesn't exist.
    - *Auto open bakes*: Will open created images in Blender after a successful bake, if those images aren't already open.
5. **Debug:** Adds more detailed messages to each process and if a bake fails with an error condition a complete process log will be opened in a new window. **Please post this log when reporting a bug.**

CHAPTER 5

Nodes

Detailed information on each node in the order they appear in the '*Add*' menu *(Note: All node settings have mouse over tool-tips with additional info)*:

Settings

Mesh Settings	Settings for your meshes and objects
Pass Settings	Settings relating to bake passes
Output Settings	Settings controlling the output format

Bake

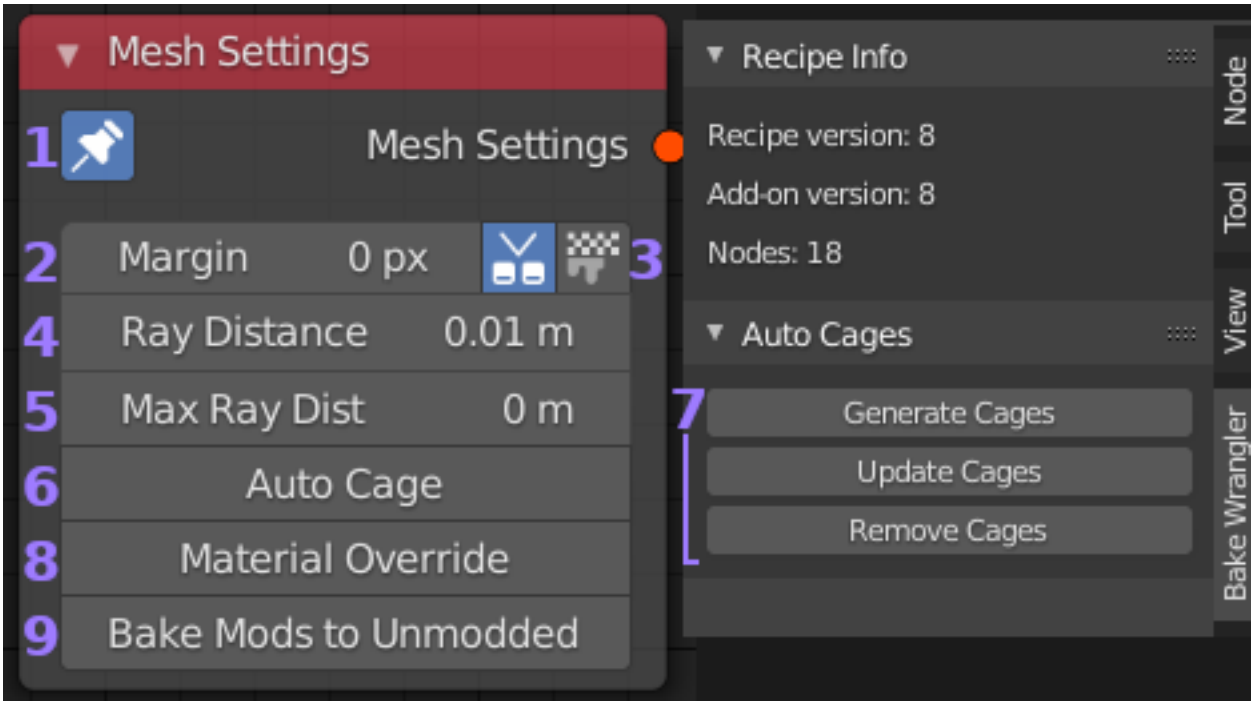
File Names	File path and name input node (used by output)
Objects	Container for groups of objects (used for input)
Input Material	Used to bake a material directly to a plane
Input Mesh	Sets relationships of input objects to be baked
Auto Sort Meshes	Sorts and groups inputs by name in various ways
Bake Pass	Selects the type of data to bake from inputs
Output Image Path	Outputs baked data to image files
Batch Bake	Groups outputs into a single process

Post

Channel Map	Allows mapping of image channels without split/join
Mix RGB	Standard mix node for two color inputs
Split RGB	Standard split node to separate R, G, B channels
Join RGB	Standard join node to combined R, G, B channels
Math	Standard math node with various functions
Gamma	Standard gamma node for applying gamma transform

5.1 Mesh Settings

The *Mesh Settings* node controls *Input Mesh* nodes and can be pinned to apply to all such nodes that have an empty '*Settings*' socket. Only one node can be pinned at a time to apply in this way.



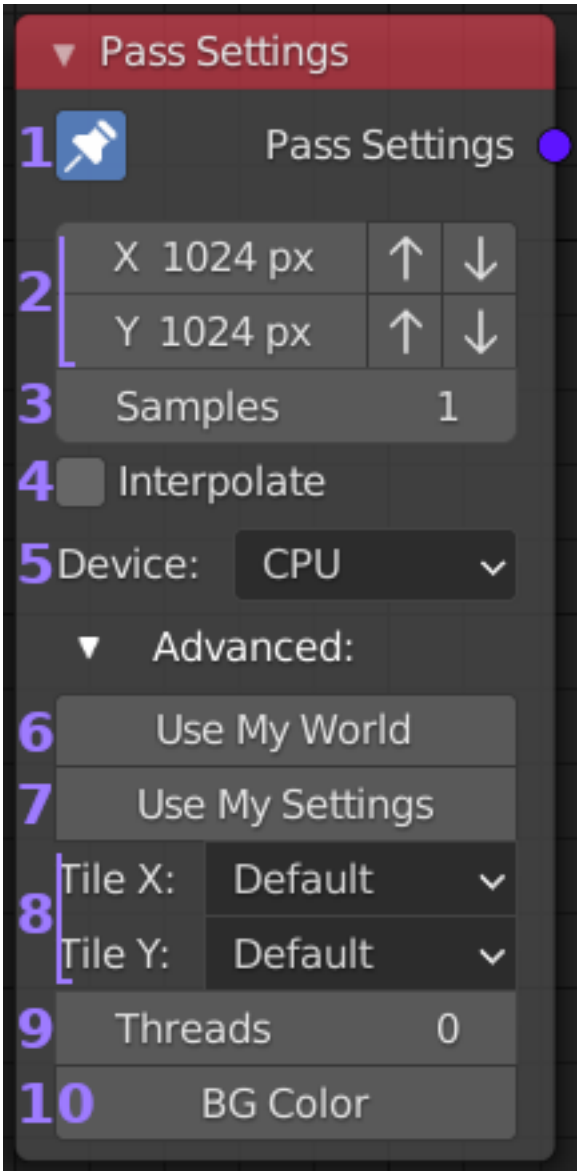
1. **Pin:** When pinned this nodes settings will be applied to all *Input Mesh* nodes with an empty '*Settings*' socket in the current recipe. Only one *Mesh Settings* node can be pinned at any time and the previously pinned node will become unpinned if another is pinned.
2. **Margin:** Number of pixels to extend out around the edges of baked UV islands. This is used to prevent bleeding of background colors into texture island edges when a coordinate lies on a pixel boundary or when a texture is re-sampled at a different resolution.
3. **Marginer Options:** You can choose to use the alternative margin generator by using the toggle button. When enabled there is also a fill option that can be enabled to fill all empty space with margin instead of using a set pixel amount. The alternative method is much slower, adding an average of 20 seconds to the time it takes to produce the final image. However it will not overlap the UV islands of other objects when using an atlas of multiple objects with a single UV map. It solves problems the default much faster margin generator can cause.
4. **Ray Distance:** Sets the distance in blender units that baking rays should be cast from above the objects surface. This only applies when baking from one or more *Source* objects. It needs to be far enough away to hit any details that extend above the surface of the *Target*. When finer control is required a *cage* object should be specified on the *Target* object.
5. **Max Ray Distance:** Limits how far a bake ray can travel to hit a surface after which it will be ignored.
6. **Auto Cage:** This will create a cage for every target object that doesn't have it's own cage specified. When enabled an expansion and smoothing angle value will appear. The expansion value determines how far out (or in with a negative value) the cage will be created from the original object. The smoothing angle is the angle below which normals will be smoothed. Complicated objects may not result in a good cage and small values for expansion should be used.
7. **Cage Generation Buttons:** In the side ('N') panel on the Bake Wrangler tab are three buttons that interface with the '*Auto Cage*' setting. The '*Generate*' button will create and display cages using the active cage settings

for objects in the current scene that are found in the current recipe. These cages can then be edited and will be used automatically when baking. A modifier is used on the cage objects, which will be updated to reflect any changes in the '*Auto Cage*' settings when the '*Update*' button is pressed. Note that if you removed the modifier and edited the cage, your changes will be lost if you press this button (Hide objects you don't want changed and shift-click the buttons in that case). Use the '*Remove*' button to delete any generated cages. The idea of these buttons is to allow for easy visualization of what the '*Auto Cage*' will look like and also for easy editing of the cage.

8. **Material Override:** This will replace all materials on your objects with the selected material during baking. Use this to either bake alternative materials without modifying your scene or to bake your own material based passes (such as OSL shaders).
9. **Bake Modifiers to Unmodified:** This can be used to bake the effect of a modifier onto the base object. Mainly this is for convenience and achieves the same result as duplicating an object, stripping the modifiers you wanted to bake and then baking the original object onto the stripped version. This can be useful to bake a bevel modifier for example. It is also possible to exclude modifiers by disabling their viewport visibility (this behavior can be inverted in the preferences). If for example you had a mirror modifier followed by a bevel you would hide the mirror (it will now get applied to both objects before baking) and leave the bevel visible so that only the difference created by the bevel would be baked.

5.2 Pass Settings

The *Pass Settings* node controls *Bake Pass* nodes and can be pinned to apply to all such nodes that have an empty '*Settings*' socket. Only one node can be pinned at a time to apply in this way.



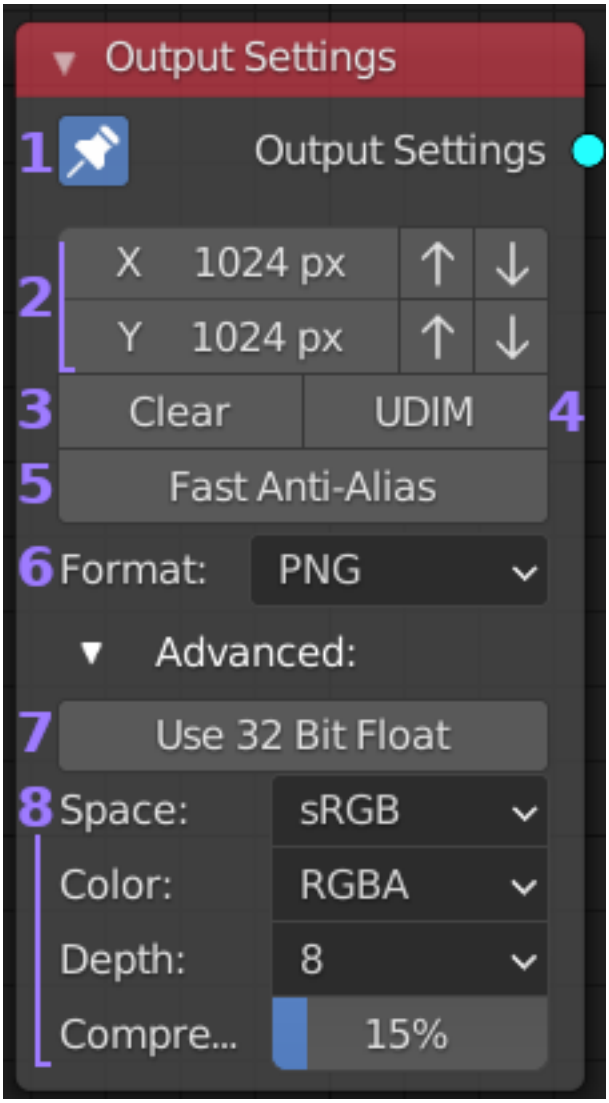
1. **Pin:** When pinned this nodes settings will be applied to all *Bake Pass* nodes with an empty '*Settings*' socket in the current recipe. Only one *Pass Settings* node can be pinned at any time and the previously pinned node will become unpinned if another is pinned.
2. **Resolution:** The X (width) and Y (height) to bake the data. Up and down arrows to the right increment the size in powers of two for convenience.
3. **Samples:** This is the number of samples taken for each pixel in the bake. For most *passes* (any PBR map, normals, etc) **one** sample is sufficient. When lighting information is needed (eg. AO *pass*) more samples will be needed to produce a good result. If in doubt start with one sample and increase if the result isn't good enough.
4. **Interpolate:** When enabled each pixel from the bake data will be interpolated when written to the output. This is mostly useful when your bake and output are different sizes. Edges will gain a very soft anti-aliased look.
5. **Device:** Simple choice between *CPU* or *GPU* as rendering device. Your *GPU* must be properly configured in Blenders settings and supported. If you don't get the expected results try changing settings (eg. switch from Optix to CUDA) or try using *CPU* instead.
6. **Use My World:** If you want to use lighting information from your scenes world in the *pass*, you need to enable

this option and select the '*World*' you want to use (if left blank, but enabled the currently active scenes world is used). By default Bake Wrangler uses a world that contributes no light.

7. **Use My Settings:** Enabling this causes Bake Wranglers default rendering settings to be replaced with the settings from the '*Scene*' you select (if left blank, but enabled the currently active scenes settings are used). The default render settings used by Bake Wrangler are optimized to quickly render maps without lighting information and complex data like hairs and caustics. If you need to bake lights, hairs or rays that are changed by passing through objects (glass, fog, etc) then you will need to use this setting with appropriate *Cycles* values set in your scene. For any PBR maps and most data maps this should be disabled for best performance.
8. **Tile Size:** Sets the X and Y render tile size. Essentially this breaks your pass into smaller tiles that can be processed in parallel. The default settings are quite good for most cases, however the tile to image size setting can be faster for *GPU* passes without lighting.
9. **Threads:** Number of render threads. Set to zero for automatic (which will be one for each core in your system).
10. **Background Color:** The default is black. This is the color that will appear in the gaps between UV islands. It is also possible to set this on the *Bake Pass* directly if you wish to only change one particular pass.

5.3 Output Settings

The *Output Settings* node controls *Output Image Path* nodes and can be pinned to apply to all such nodes that have an empty '*Settings*' socket. Only one node can be pinned at a time to apply in this way.



1. **Pin:** When pinned this nodes settings will be applied to all *Output Image Path* nodes with an empty '*Settings*' socket in the current recipe. Only one *Output Settings* node can be pinned at any time and the previously pinned node will become unpinned if another is pinned.

2. **Resolution:** The X (width) and Y (height) to output the data. Up and down arrows to the right increment the size in powers of two for convenience. Using a higher bake size than output size allows for down sampling.

3. **Clear:** When enabled and if the target image already exists, it will clear the image to black (and transparent if supported by image settings) before writing bake data.

4. **UDIM:** Assumes UV maps follow the UDIM format and will output tiles based on their UV location. Each tile will have their number appended to the file name. The result for tiles outside of the UDIM range is undefined. A UV space with no tiles will result in a single tile output.

4. **UDIM:** Assumes UV maps follow the UDIM format and will output tiles based on their UV location. Each tile will have their number appended to the file name. The result for tiles outside of the UDIM range is undefined. A UV space with no tiles will result in a single tile output.

6. **Format:** Drop down list of supported image formats. This will default to the output format selected for your current scene when placing the node. The options in section *[8]* will change depending on the format chosen.

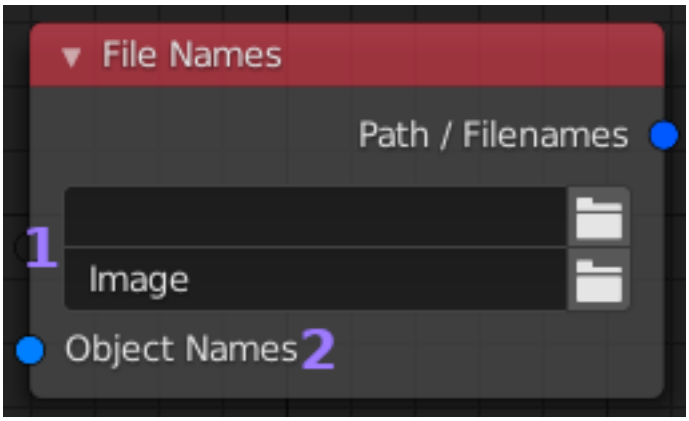
7. **Use 32 Bit Float:** Normally blender renders images using 8bits per channel per pixel (24bpp or 32bpp with

Alpha). If you want to save your output in a format with more data per pixel than that you need to enable this option. All contributing *Bake Pass* will then use 92bpp (128bpp with Alpha). This uses up 4x more memory than standard, but can be useful for *data* maps (eg. normals) to allow more variations in color (and hence a more accurate representation of the data). For plain color maps it generally doesn't provide any advantages. You must also use an image format and bit depth with higher than 8bpp or the extra data will be lost and colors may appear different than expected as they are remapped to a lower bit depth.

8. **Format Options:** This group of options are specific to the chosen image format and will change accordingly. Almost all formats support '*Color Space*', if you are unsure what to pick use '*sRGB*' for color information and '*Non-Color*' for data maps like normals. For any other settings either use the defaults or check the tool-tip for more information if you are unsure.

5.4 File Names

The *File Names* node can be used as input to *Output Image Path* to set the path and file name used to save the output. This is simply to allow multiple outputs to be controlled from one node. It also accepts a list of objects. When a list of objects is set, the output will be split by those objects and their names are used.

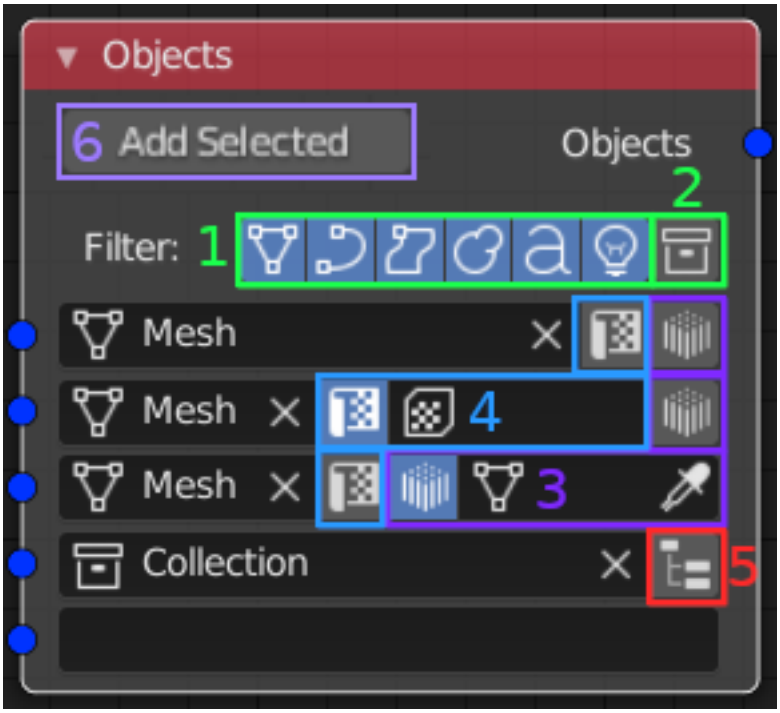


1. **Path & Name:** The file path (which can be relative: eg. // for current) and the starting portion of the file name. A suffix and file extension may be added by the output node. If the path doesn't exist (and the '*Create Paths*' preference is set) an attempt will be made to create the path when the output is baked.
2. **Object List:** Any *blue* colored socket that would normally present a list of objects can be connected here (note: you can skip adding this node and connect the object list directly to a *Output Image Path* node and get the same result). Having a list of objects connected will change how output files are generated. Firstly only objects that appear in the list will result in an output, other objects provided to the bake input but not appearing in this list will be included as a '*background*' to every object that **is** in the list. List objects not in a bake are ignored. Additionally each list object that is also a bake object will result in a separate file with the objects name appended. This feature is intended to greatly reduce the number of nodes needed to create many separate files.

5.5 Objects

The *Objects* nodes role is to contain lists of objects and collections which are used by the Mesh nodes input sockets. *Objects* nodes can be chained together and the node itself will continue to expand as objects are added, allowing for any number and combination of objects.

Some objects have additional options that can be set once they are added. This is where **Cages** and **UV Maps** are configured.



1. **Filter:** Selects what object types will be shown in the search boxes and starts with all types enabled.
2. **Collections:** Toggles between selecting objects or collections. The node can contain a mixture of both, but can only search for one or the other.

Note: If an object is contained in both a selected collection and specifically listed, the specific listing will take preference. It is also possible to list the same object multiple times with different settings (UV Map and/or Cage), in which case it will be evaluated multiple times.

3. **Cage:** When the selected object can support a cage (*'MESH' type objects*), this button will appear at the end of its row. Toggling the button will enable using a cage for that object when baking (*if using a cage makes sense for the bake*). The cage object must have the same number of vertex as the original after all modifiers are taken into account. On click it will automatically search for and select an object that has the same name as the original plus a spacing character follow by 'cage' (*Eg. cube.cage*).
4. **UV Map:** Allows selection of a specific *UV Map* to use for the object. This option only appears when the object is able to support multiple maps (*Currently that is 'MESH' type objects*). When left blank or disabled the currently active map is used. Removing or renaming the referenced map on the object will break the reference and must be manually updated before baking can start.
5. **Recursive Selection:** When the selected item is a collection this option will appear. Enabling it will cause any and all sub collections of the primary selection to also be included.
6. **Add Selected:** Clicking this button will add all currently selected objects to the node, respecting the current filter. Duplicate items will not be added unless SHIFT is held while clicking. It is not possible to add collections in this way.

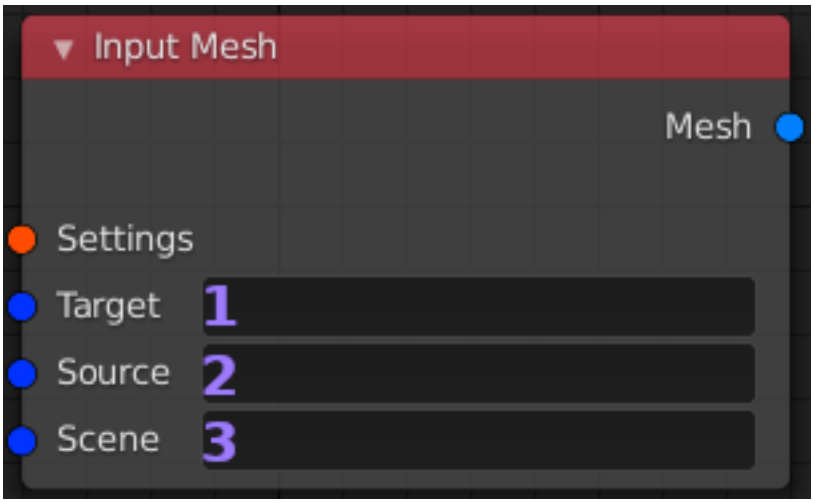
5.6 Input Material

The *Input Material* node is an alternative input to a *Bake Pass* which will supply materials instead of objects. Each material is baked to a plane of the set size.

1. **Width & Height:** The width and height of the plane used to bake the material on to. You need to set this correctly for some procedural textures as it will effect what appears on the plane.
2. **Materials from Objects:** Takes any list of objects, but will only consider the materials on those objects and bake them to the plane.
3. **Materials:** Allows selecting of any material in the current blend file to bake to the plane.

5.7 Input Mesh

The *Input Mesh* node sets the relationship between inputs. They are either '*Targets*', '*Sources*' or '*Scene* elements'. The settings for this node are found in the *Mesh Settings* node which can either be attached to the '*Settings*' socket or left empty to instead take the settings from a pinned node. It can take *Objects* nodes as any of its inputs, and outputs to a pass node (which can be through a *Auto Sort Meshes* node). Only objects that are linked to this node will be included in a bake pass and will be completely isolated from all other objects in the .blend file.



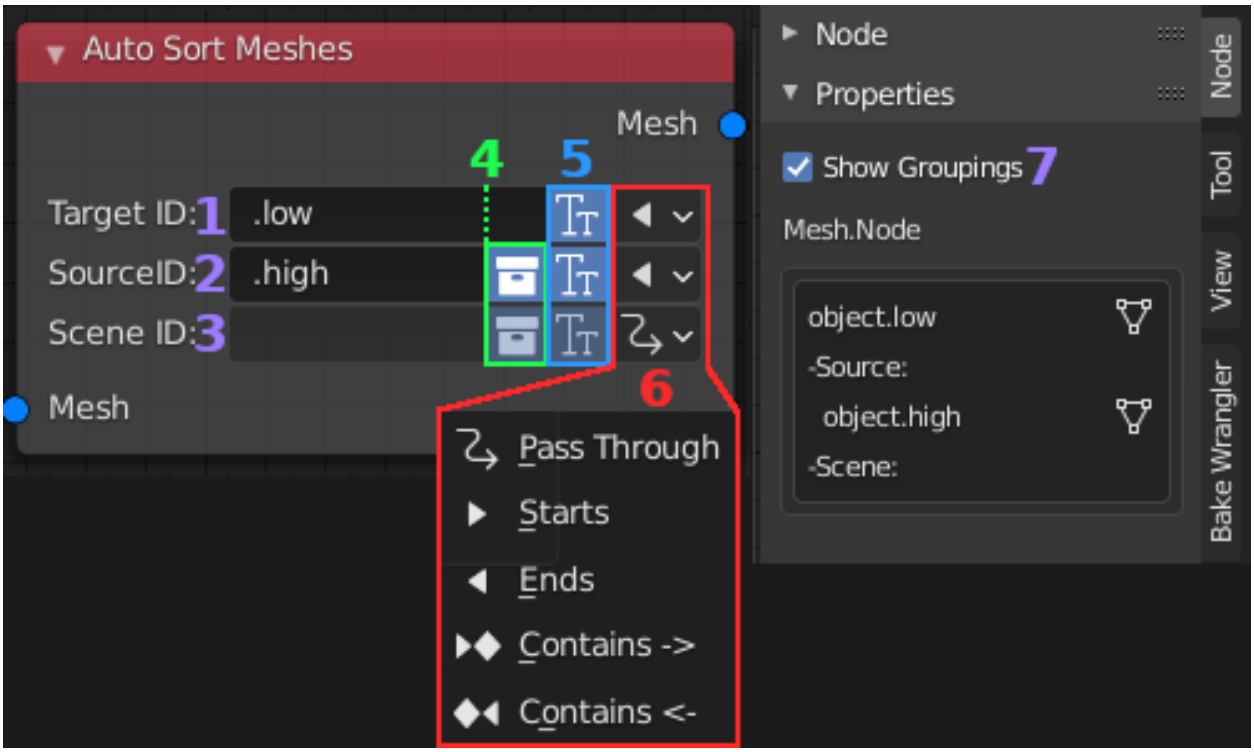
1. **Target:** May be a single 'MESH' type object or taken from a *Objects* node. When a list of objects is used, only 'MESH' type objects will be considered with other types ignored. Each *Target* will be baked in its own pass but shares the *Mesh* nodes settings. At least one valid object must be selected for a bake to valid.
2. **Source:** Optional field that can take a single object of most types or a list of objects from a *Objects* node. All of *Source* objects will have their surface data projected onto to the *Target(s)* (if sensible for the selected bake pass).

Normally a ray distance greater than zero is needed to capture everything correctly.

3. **Scene:** Optional field that can take a single collection or a list of objects from a *Objects* node. This is the only input that will consider lights as valid objects. This input is used to set up lighting, shadow casting objects and anything that indirectly influences the pass but isn't directly mapped to the *Target*. Generally unless you are trying to capture lighting information this input is not needed. It's important to note that the '*Combined, Diffuse and Glossy*' passes all require lights and will be blank if you don't have any.

5.8 Auto Sort Meshes

The *Auto Sort Meshes* node sits between inputs and *Bake Pass* nodes. In the most simple configuration it can just be used to group inputs. It's main role is to automatically group together objects that have a relation with each other by matching something in their names. The most natural case for this is high-poly to low-poly baking, but can be applied in other cases.

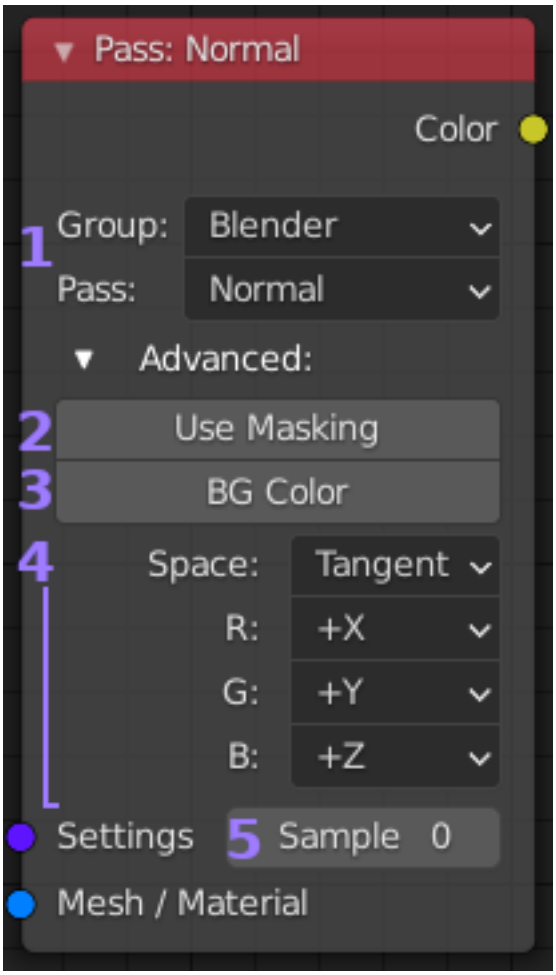


1. **Target ID:** String used to identify target objects. Can be empty to match for example 'object' to 'object.hi'.
2. **Source ID:** String used to identify source objects. Can also be an empty string.
3. **Scene ID:** String used to identify scene objects, for completeness.
4. **Match Collections:** When enabled, the name of an entire collection will attempt to be matched with the target before single items inside the collection. This allows for multiple items to be assigned to a single target by putting them in a collection.
5. **Case Sensitive:** Enables or disables case sensitive matching of the ID string.
6. **Search Direction:** The ID can be looked for at the start or end of the name as well as with-in the name searching from the front or back. The '*Pass Through*' option means no matching will be done on that field and it will be passed on unchanged from the input node. By having all the fields set to '*Pass Through*' the node can be used to group any number of inputs into a single node.

7. **Show Groupings:** In the right side ('N') panel a preview of what groupings will be made can be shown by ticking this box. For performance reasons it isn't shown by default as each time the area is redrawn it must perform the name matches which could slow down the UI.

5.9 Bake Pass

The *Bake Pass* node is where the type of bake (*pass*) is selected and any pass specific options are set. It can take any number of *Input Mesh*, or *Input Material* nodes as input (adding inputs will cause it to keep expanding to accept more). The inputs can also pass through a *Auto Sort Meshes* node. The settings for this node are found in the *Pass Settings* node which can either be attached to the '*Settings*' socket or left empty to instead take the settings from a pinned node.



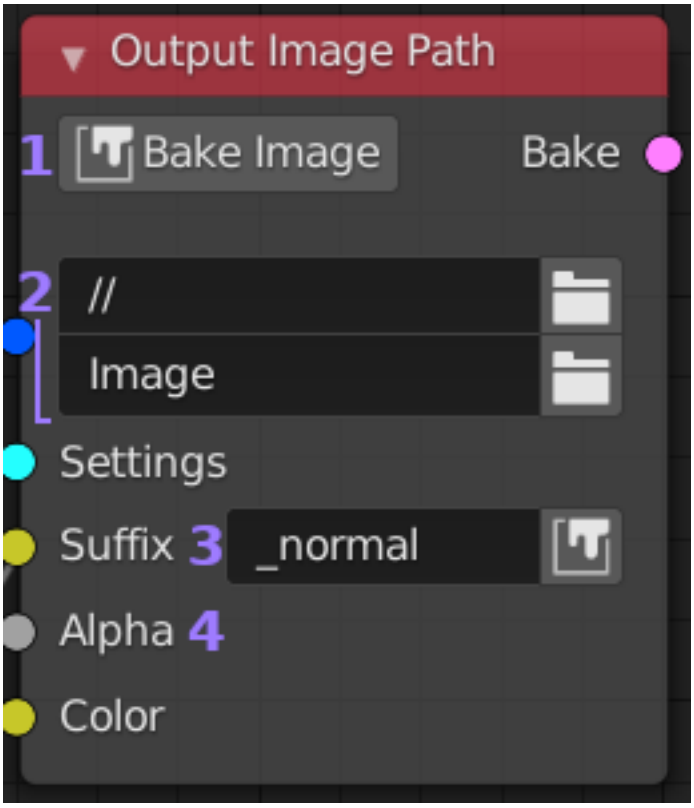
1. **Group & Pass:** Drop down list to select bake *pass*. Tool-tips are used to explain their functions. Additionally some passes require a *Principled BDSF* based material on the objects, the tool-tip on the *pass* should list any requirements. Some passes may have additional options which will appear under '*Advanced*' when selected (see 4).

2. **Use Masking:** Enabling this option will create a second image of the same size, where white pixels (*1s*) map to the UV islands of the bake and black pixels (*0s*) map to pixels that aren't part of an island. This map will automatically be used when writing to the final output to leave unmasked pixels unchanged. While many bakes don't require this, it allows for much greater flexibility in layering *passes* and objects into a single final image. The time taken to generate the mask is negligible, take a look at the *Masking* for an example.

3. **Background Color:** The default is black. This is the color that will appear in the gaps between UV islands. It is also possible to set this on the bake_settings node if you want it to apply to more than just this pass.
4. **Additional Pass Settings:** If the selected pass has additional options they are displayed here. Use the tool-tips to see what they do, or if in doubt leave at default.
5. **Samples:** When using a pinned *Pass Settings* node, the sample count can be set differentely on each pass. A value of zero uses the pinned setting. This is because often only one or two passes will need a different sample count while still using the same settings for everything else, saving the need for extra nodes.

5.10 Output Image Path

The *Output Image Path* node as you may expect outputs your bake data to an image file. The output settings are controlled by a *Output Settings* node which can either be attached to the '*Settings*' socket or left empty to instead take the settings from a pinned node. The path and name can be set from a *File Names* node. A list of objects can be connected to the blue socket instead, which is described on the *File Names* page.



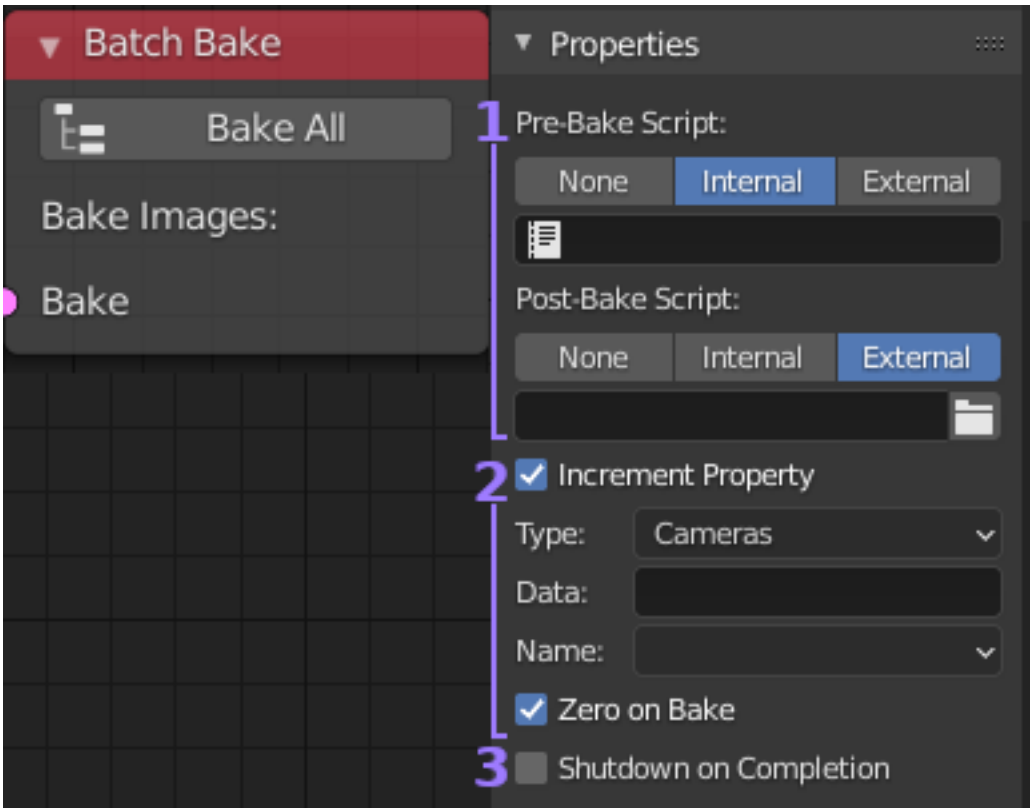
1. **Bake Image Button:** After validating settings this will cause all connected *Bake Pass* nodes to be processed and any post processing to be done. Each Color socket will be saved separately and have the Suffix if any added to the name as well as the file extension. If the created image is not open in the current blend file and the '*Open Images*' preference is set, the image will also be opened.
2. **Image Path:** Simply used to select the path where the image will be saved. Relative paths may be used (*eg. // to refer to the path the .blend file is in*). A *File Names* node can be connected instead. It's also possible to use a list of objects either directly or via the *File Names* node which changes how things are split up into output files. Read the *File Names* page for details on that feature.
3. **Color/Suffix:** When a color is connected to this socket it will change to '*Suffix*', allowing you to enter a file name suffix that will be appended to the end of the file name before the dot extension. There is also a single

input bake button to the right of this field for baking only that input (useful when testing settings, etc). An empty socket will always be added to the bottom of the node to allow as many inputs with suffixes as you want.

4. **Alpha:** If the output image format supports an alpha channel and '*RGBA*' mode has been selected, then each connected socket will also have an '*Alpha*' input below it. A value or color can be used as input (if a color is connected a drop down will appear to select what channel to use, the default being pixel value). Alpha channels are always created in Linear space as this is what OpenGL, etc expects.

5.11 Batch Bake

The *Batch Bake* node is primarily for grouping a set of outputs into a single process. Pressing the '*Bake All*' button will validate and process all the attached nodes, generating them in the order they are connected. However it also provides a couple of advanced features found in the right side ('N') panel.



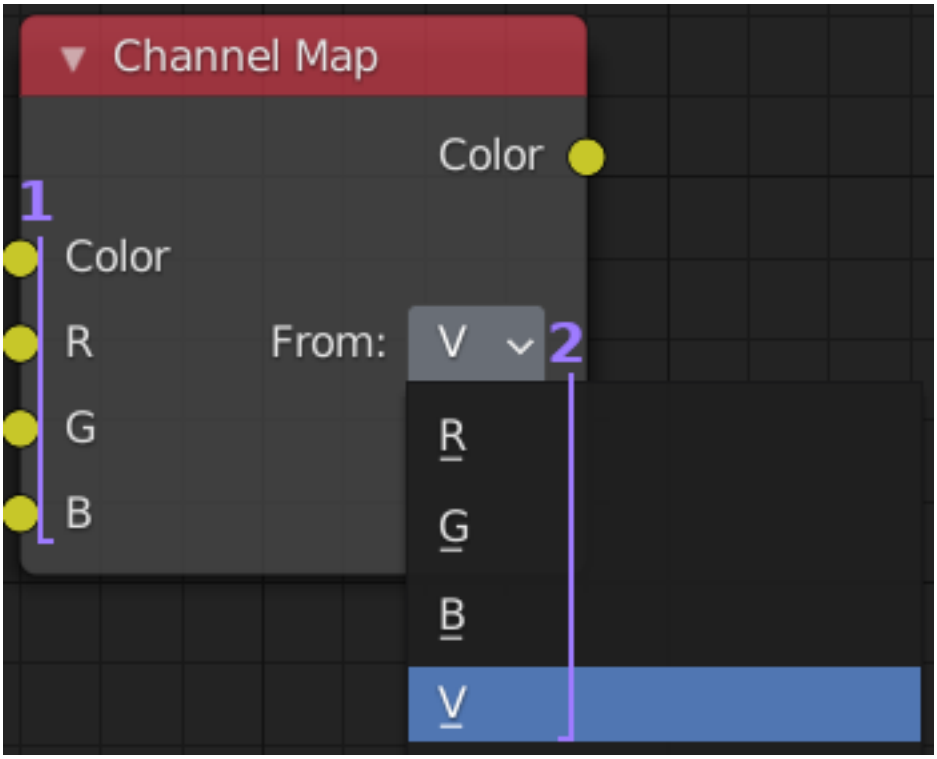
1. **Pre & Post Bake Scripts:** Python scripts can be executed before and after the batch process, using either external py files or internal text blocks. The scripts will have access to two global variables named '*BW_TARGETS*' and '*BW_SOURCES*'. The variables contain a list of unique objects which have been evaluated as targets and sources. If you require other data, consider making a feature request.

2. **Increment Property:** This is a somewhat advanced feature. Essentially you provide a custom number property which Bake Wrangler will increase by 1 each time it completes one of the batch inputs. The check box will set it to zero at the start if enabled. This would let you change things in the scene between each batch input using drivers connected to the supplied property. This allows for a large degree of automation.

3. **Shutdown on Completion:** Will attempt to power off the system once the batch has completed. Useful when running very long processes unattended. It should work fine on Windows systems, but MacOS and Linux require the user to be able to run 'sudo shutdown' without entering a password.

5.12 Channel Map

The *Channel Map* node is a short-cut for mapping the RGB values of a color removing the need for separate split and join operation in most cases.



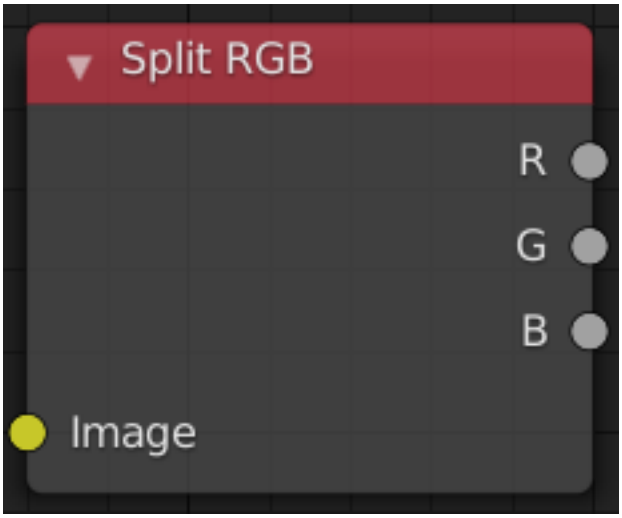
1. **Inputs:** There are four inputs, the first being color which sets all channels to its values. The subsequent single channel inputs will override the first input if used. All the inputs accept RGB and provide a drop down once connected to select the source for the channel seen in (2). This allows for lots of single node color channel manipulations.
2. **From:** When any of the RGB single channel inputs have a color connected to them a drop down is displayed to select where to take the value from. This can either be from the inputs R, G, or B values. Or V for overall pixel value, which is the default.

5.13 Mix RGB

The *Mix RGB* node works in exactly the same way as it does in a material. The possible ways to combine the inputs are: Mix, Add, Subtract, Multiply, Divide, and Overlay.

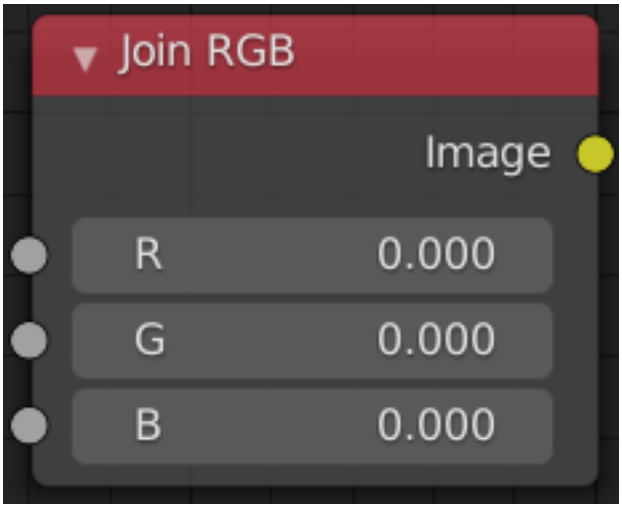
5.14 Split RGB

The *Split RGB* node separates the input color into single R, G, and B values.



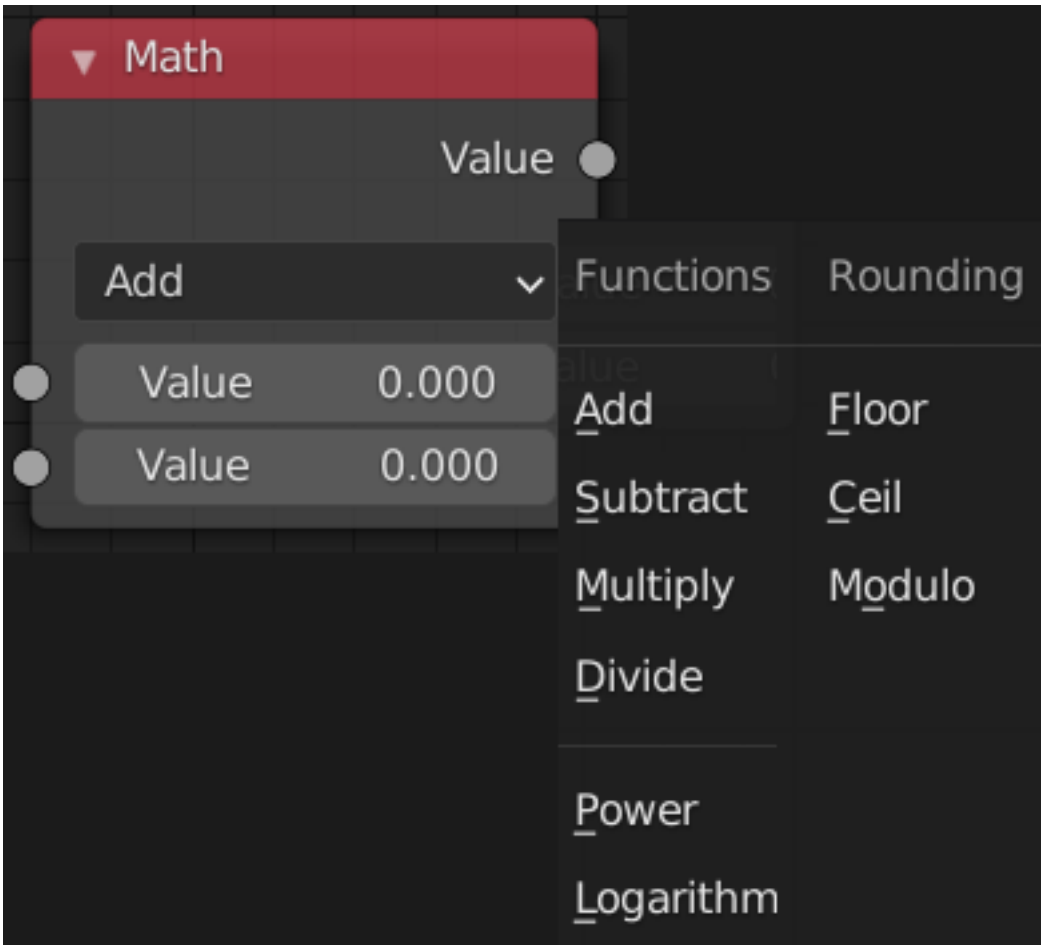
5.15 Join RGB

The *Join RGB* node combines separate R, G, and B values into a single color output. A constant value for any of the channels can be entered instead of using some input.



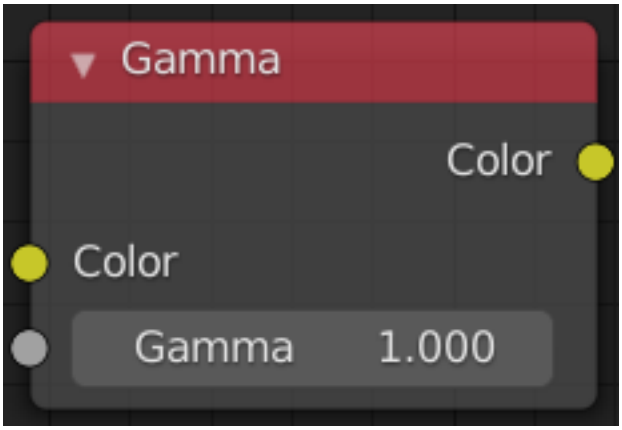
5.16 Math

The *Math* node works in the same way as it would in a material. The possible functions are: Add, Subtract, Multiply, Divide, Power, Logarithm, Floor, Ceil, and Modulo.



5.17 Gamma

The *Gamma* node works as you would expect, applying a gamma transformation to the input colors.



CHAPTER 6

Examples

Bake Wrangler is capable of many complex configurations and provides two paths to generate output. The following examples better illustrate how some features work and give some ideas of what is possible:

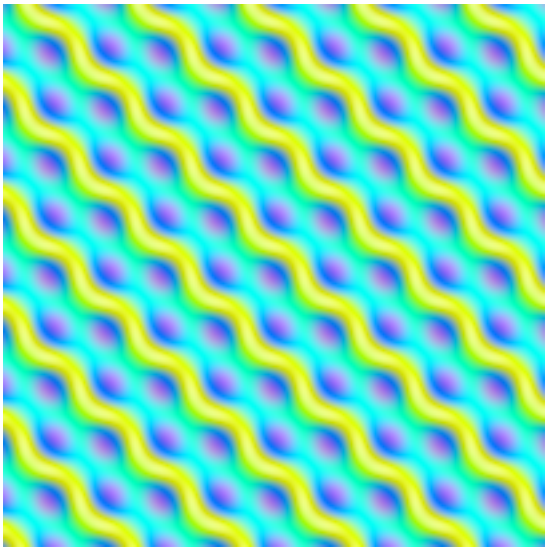
Masking	Using the '*Masking*' feature

6.1 Masking

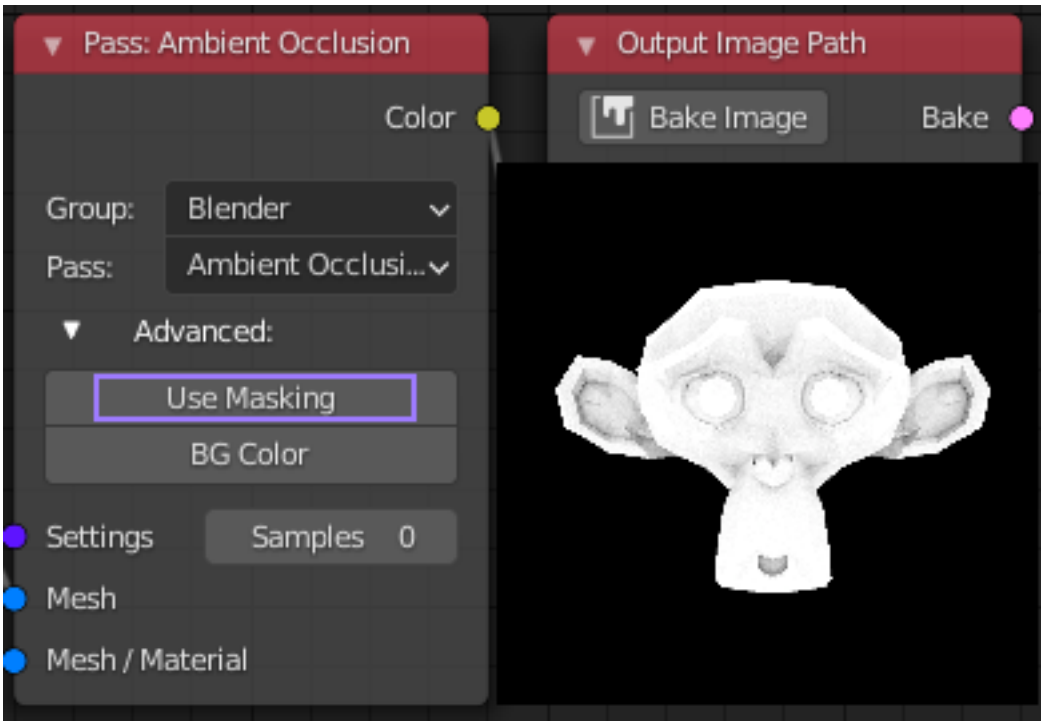
One of the advanced options on the pass node is to use *Masking*. For most tasks there is no need to use this setting, but it allows for many configurations that would otherwise be impossible.

When enabled a simple black and white map of the UV space used by the bake pass will be created and used to determine what pixels in the output get modified. This simple example will demonstrate what this looks like.

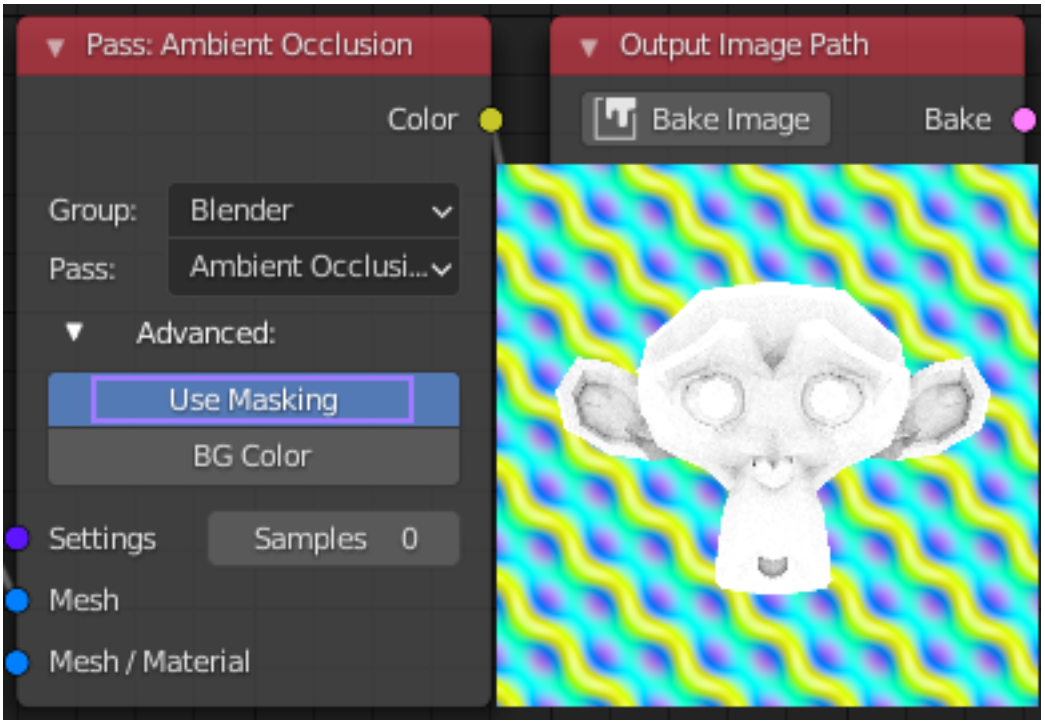
We will start with the following image and then bake the AO pass of a Suzanne over the top of it with *Masking* turned off and then on:



With *Masking* disabled, the original image/channel is completely replaced:



However when *Masking* is enabled, only the pixels which contribute to the pass are replaced, leaving the original unchanged in the rest of the image:



There are many ways in which this can be used, either with images created by external software that you want to add to. Or with layering of passes of different types or with different settings. But it is completely unnecessary when layering passes of the same type with the same settings.

CHAPTER 7

Bug Reports

Have you found a bug? *Please* report it here (blenderartists.org/t/bake-wrangler-node-based-baking-tool-set/) following these guidelines:

Always post the steps to reproduce the bug.

User interface bug: Along with describing what you did, post a screen shot if possible/relevant and any error messages produced in the console.

Bake process bug: Make sure to enable '*Debug*' in the *Preferences*. This should cause a complete log of the process to open in a new window when the process fails. Please post this text with your report. If a log doesn't open you may need to post your .blend file.

Bake output bug: If the results of a bake pass don't match what is expected, first verify your node tree set up and check the tool-tip for the pass you are using to see if it has any special requirements. If the result is still incorrect post your node tree along with the expected and actual outputs. I will likely need your .blend file also.

Thanks for helping to improve Bake Wrangler!