Open Brush
User Variant Brushes

Author: Tim Aidley (CaptainXap)

Summary

Users are very excited about opportunities for making custom brushes, but at the moment making a custom brush requires building Open Brush in Unity, and offers no system for compatibility if a sketch is made with one.

This outlines a first pass of an approach that could give people easy-to-author custom brushes that will not require special builds of Open Brush to work.

Authors: **Tim Aidley**
Status: **Draft**
Last Updated: **2021-02-12**

Problem

The availability of experimental brushes in the open source version of Tilt Brush has generated quite a lot of excitement, as has the possibility of new, custom brushes. Several people have added brushes to their own builds of Tilt Brush, and guides have been put up online about how to do it.

However, the issue with these new brushes is that they can't easily be shared; a brush's data would have to be integrated into the main Open Brush codebase, and a new build of Open Brush would have to be made. People using an older version of Open Brush would not be able to see strokes made with the new brush.

As well as all of this, the Open Brush UI is not well suited to many brushes, and one can easily foresee us ending up with hundreds of brushes.

User Variant Brushes

An interesting thing to note about the brushes I've seen online so far is that they are largely all just 're-skins' or variants of existing brushes. Only a couple of people have experimented with custom geometry creation and shaders so far.

Therefore an interesting first step would be to allow users to create *variants* of existing brushes - they would be able to switch out textures and sounds, and be able to override some parameters.

An advantage of this scheme would be that a brush creator would not have to download Unity, and would not have to download the Open Brush source. The technical skills required should be fairly low.

Limitations

The variant brush approach does not allow people to make new brushes with new shaders, or geometry creation code. We should look into systems that would allow that, but it doesn't seem to me that brush variants would interfere with that.

Anatomy of a Variant Brush

A variant brush would be either a folder or a packed file (a .zip, or perhaps a .zip renamed to something else... an .obb? .obvb? .obv? .brush? .obbrush?), which would contain a Brush.cfg file, along with any new textures or sounds required by the brush. The Brush.cfg would at a minimum contain:

- The built-in brush the variant is based upon.
- The name of the new brush.
- A unique GUID for the new brush.

It could also contain some other parameters that override the parameters on the base brush, like:

- Shader parameter overrides.
- Material texture overrides.
- Parameters to do with brush sizes etc.

The brushes would be stored in a folder called ‘Brushes’ in the main user OpenBrush folder.

Fallback Brushes

What happens when you load a .tilt file that contains a variant brush that isn’t in your folder? Ideally we would notify the user, and show those brush strokes in some form of plausible fallback stroke from the built-in brush types. One issue is that without the variant brush data, how do we know which brush it is based upon?

I suggest that we adjust `SketchWriter.WriteMemory` to create a `baseBushGuids` list as well as a `brushGuids` list. This would contain whatever each brush is based upon, so that if a particular guid was not found, we could revert to it. We would probably also want to adjust `StrokeData` to store the base brush guid as well, so that when the scene gets saved out it still retains the correct stroke guids, even if some brushes are not present.

Distributing Variant Brushes

People could obviously distribute brushes by sending the packages files to each other, or from downloading them from a website and putting them into their Brushes/ folder. However, it would be nice if you didn’t have to go and search for all the correct brushes and install them.

Enclosing brushes in the .tilt file

I suggest that variant brushes be included in the .tilt file itself. This will mean that everything required to render the .tilt file is bundled with the sketch[1] and no extra work will be required by the user to view them.

This does increase the size of the .tilt file somewhat, but not by *too* much. In my tests a variant brush with two textures stored as .png took up around 270KB. It would be less with .jpg textures.

Downloading variant brushes from a sketch

We should make it so that brushes contained within a sketch are available on the brush panel for the user to use. We could add a little ‘download’ icon to allow the user to download it to their brushes folder.

Protecting artists’ work

When creating a variant brush, we should give the creator the option of making a brush sharable or not sharable. Some artists may spend a lot of time creating their brushes, and may not want them to be able to be copied freely. At a minimum this should be a flag within the Brush.cfg file

---

[1] Okay, not *everything*. We should look at including reference images and models as well, but that’s another project.

which prevents the brush from appearing on the brushes panel, and therefore prevents it from appearing for download.

We can never make it impossible for people to extract non-sharable brushes, but we should at least prevent it from being simple. Perhaps encrypt non-sharable brushes when they are added to a sketch?

Implementation

So far a basic system has been implemented that works with subfolders in the Brushes/ folder, and supports overriding the main texture and normal map. Saving is supported, although enclosing brushes in .tilt files is not.

Work in progress can be found here:
https://github.com/TimAidley/open-brush/tree/features/simple-brushes

(but be aware that I often rewrite history on that branch!)

There is a Trello following progress:
https://trello.com/c/Y6oHQ49n/39-user-variant-brushes

UserVariantBrush

When a *UserVariantBrush* is created, it instantiates a duplicate of its base brush's *BrushDescriptor*, and then alters any values that are overridden in the config file. Currently user brushes are only loaded once at startup (in App.Awake), but we should make a file watcher that will update brushes to ease brush development.

Brush Layout

Brush folders are currently laid out as so:

```
OpenBrush/
    Brushes/
        Brush.cfg
        main.(png/jpg)
        normal.(png/jpg)
```

Brush.cfg format

Brush.cfg is a JSON-formatted file - here is an example:

```
{
	"Name" : "Goofy",
	"Guid" : "98e8906e-4e7a-4ed8-966f-15e81aa78e7b",
	"VariantOf" : "Splatter",
}
```

Note that currently the main and normal textures are overridden just by having specifically named textures. I am thinking about changing that so that they get specified in the .cfg file.

Issues

The brushes panel doesn't work properly with non-atlased textures. I think for now I will make them non-atlased until we get closer to release with it.

Further Thoughts

Once this basic system is up and running, we should probably think about some other things, like:

- Perhaps we could have a brush library hosted online at Icosa?
    - Perhaps downloaded at runtime, a bit like Poly stuff is now?
- We may need to be able to favorite brushes and organize them.
- People may want to distribute 'brush packs', which we should support.
- People may want to sell brush packs, as is common with Photoshop.

Possible Overridable Brush Descriptor properties

- GUI
    - Durable Name - Human readable - not necessarily unique
    - ButtonTexture
    - Description
    - Extra Description?
- Audio
    - Audio clips
    - Maximum pitch shift
    - Maximum Volume
    - Volume up speed
    - Volume down speed
    - Volume Velocity Range Multiplier
    - Audio reactive? - unsure if it makes sense to expose this
    - Button audio clip
- Material
    - [Material Properties]
    - Texture Atlas V

    - Tile Rate
    - Use Bloom swatch on color picker - unsure if we should expose this
- Size
    - Brush Size Range - this will affect existing art if changed
    - Pressure Size Range - this will affect existing art
    - Size Variance
    - Preview Pressure Size Min
- Color
    - Opacity
    - Pressure Opacity Range
    - Color Luminance Min
    - Color Saturation Max
- Particle
    - Particle Speed
    - Particle Rate
    - Particle Initial Rotation Range
    - Randomize Alpha
- QuadBatch - this has the comment ‘to be removed’! Should we keep it!?
    - Spray Rate Multiplier
    - Rotation Variance
    - Position Variance
    - Size Ratio
- M11 Compatibility - we should not allow this one to be set
- Tube
    - Sold Min Length (meters)
    - Tube Store Radius in Tex Coord 0Z
- Misc
    - Render Backfaces
    - Back Is Invisible
    - Backface Hue Shift
    - Bounds Padding
    - Playback at Stroke Granularity
- Export Settings
    - Emissive Factor
    - Allow Export - probably skip this one?
- Simplification Settings - not sure if we should expose these
    - Supports Simplification
    - Head Min Points
    - Head Point Step
    - Tail Min Points
    - Tail Point Step
    - Middle Point Step

Also - add something to override material properties. Textures should override with filenames for textures within the brush; other values should be in the json. Should we be able to override shaders? Why not, I guess.

Rough JSON outline:

- Metadata?