Rules and Theory of GEGL Effects (for future maintainers/Gimp's team)

If you are reading this I assume you are part of Gimp's team maintaining my plugin. You are free to break all of these rules but you won't be helping Gimp progress that way. If you want to fork GEGL Effects and make it better I strongly urge you adhere to these rules.

1. The ultimate purpose of GEGL Effects is to resemble Adobe Photoshop's layer effects but with a unique twist for the FOSS world.

2. In default GEGL Effects upon launch must always do nothing to an image until the user specifies an option. No image modifying task should be enabled by default. Becareful to make sure that a GEGL node that double stack an image is not running. In particular gegl:nop's and normal blend modes can do this.

3. Multiply must always be the default blend mode for color overlay. This is because when white it is transparent, thus allowing GEGL Effects to work on both recoloring white text and blending image files of people, animals and objects in default. If it was a solid color fill then images of people, animals and objects would automatically be color filled, which is undersirable.

4. GEGL Effects is suppose to be a Gimp GEGL plugin hybrid. GEGL Effects uses both gegl operations and gimp exclusives operations like "gimp:layer-mode layer-mode=" this is because Gimp only blend modes have many more options then GEGL only blend modes. This means GEGL Effects will only work inside Gimp. An exception to this rule is future updates to GEGL blend modes that give them more options. As of now GEGL blend modes are inferior to Gimp only blend modes. Beaver is okay with the fact that these plugins don't work outside of Gimp. GEGL Effects would lose many features if it was GEGL only.

5. Renaming ENUM list, Changing the order of blend modes or options in the ENUM list and removing an ENUM list can break presets and or introduce a known bug that doesn't go away until a file associated with the filter inside (/config/GIMP/2.10/filters) is deleted. This problem only gets negated if you make a new operation name and title that disregards all presets. This is why my blend mode list in GEGL Effects are not all elegantly put in the same order. I can't reorder them without breaking every thing

5.2 Two ENUM list of any GEGL filter or plugin that is part of a Gimp install cannot share the same ENUM list name or it will create a bug where none of the ENUM list of any Gimp filters will show. The bug goes away when the ENUM receive a different name. Sometimes it creates a scenario where Gimp does not even start up.

6. Don’t make a GEGL operation that uses an operation name, binary name, or title I already chosen. Or else it will conflict with my plugins and break them or vice versa with my plugins breaking your plugins.

7. Do everything you can to make sure presets do not break. But if presets must break due to novel features, an attempt must be made to deem said current version of GEGL Effects to "Legacy Continual (date here)" It requires renaming the title, operation name, c file, ENUM list and reference hash to something new. As the goal is to make both versions of GEGL Effects co-exist. However, users should be hesistant to break presets. This is not required if the only issue is one presets loading invalid settings from an update. I am referring to scenarios where presets won't load.

8. InnerGlow, Bevel and Glass Over Text are suppose to be dependencies. Inner Glow, Bevel and Glass Over Text are stand alone GEGL plugins of Beaver's that can also run most of their options inside GEGL Effects. They are not meant to be baked inside GEGL Effects. Technically GEGL Effects has a ton of depedencies if you include native Gimp and GEGL operations.

Predicting how future Gimp updates can break GEGL Effects

To help future devs I will list predicted scenarios that I believe will lead to GEGL effects or any of my plugins breaking compatibility with future versions of Gimp.
--
First example of how GEGL effects can and eventually break is if Gimp team adds new blend modes.

In Example, Gimp's Split blend mode is **60**

```
split = gegl_node_new_child (gegl,
"operation", "gimp:layer-mode", "layer-mode", 60,
"composite-mode", 0, "blend-space", 0, "composite-space", 0, NULL);
```

If Gimp's team adds a new blend mode this will throw off all the gimp exclusive blend modes called by one number. Resulting in a scenario where all my plugins will not work (by using the wrong blend mode) until the number **60** is changed to **61** or **59**. (we'll have to figure it out when it happens). All Gimp exclusive blend modes in my codebase will have to be updated when this happens. That will break a ton of my plugins but the fix should be trivial.

--
The second most common reason GEGL Effects will break is if a GEGL operation changes its value. In example gaussian blur's values are

gaussian-blur std-dev-x=30 std-dev-y=30

but lets say a hypothetical update from the gimp team comes and changes it to

gaussian-blur x=30 y=30

Everything that uses gaussian blur will then break.

Until **std-dev-x** and **std-dev-y** are replaced with **x** and **y**. All plugins of mine that call gaussian blur will remain broken. This can happen to any operation not just gaussian blur.
--

The third most common reason GEGL Effects will break is if a operation is removed or renamed. It will then have to be rebuilt and shipped with GEGL Effects
hypothetical scenario would be if gegl:emboss got renamed to gegl:embosser; we'd have to udpate all my filters that call emboss and rename them. Or grab an older version of GEGL's source code and compile a hypothetical disregarded operation that my plugins use.

The fourth most common scenario is if Gimp's team adds a new filter that has either the exact same title name, operation name, or binary name. Then the fix would be to change the title, operation name and binary name of my plugin to get it working again.