Python Fu #6: Accepting user input

September 14, 2015November 30, 2015 | jacksonbates | gimp, gimp scripting, python, python-fu, tutorial
See the other tutorials in this series. (https://jacksonbates.wordpress.com/python-fu-gimp-scripting-tutorial-pages/)

In this tutorial we will take the plug-in we made from the last tutorial, and allow the user to pass their own parameters into the function. The code for this tutorial can be found at: UI Examples (https://gist.github.com/JacksonBates/44be64ffe89130f4a348) and extreme_unsharp_desaturate_options.py (https://gist.github.com/JacksonBates/4f70cfc24fc4ac0c73cf).

The first thing I’ll do is change the name and register arguments for this script.

```
def extreme_unsharp_desaturation_options(image, drawable):
    # function code here..


register(
    "python-fu-extreme-unsharp-desaturation-options",
    "Unsharp mask and desaurate image, with options",
    "Run an unsharp mask with variables set by user",
    "Jackson Bates", "Jackson Bates", "2015",
    "Extreme unsharp and desaturate options...",
    "RGB",
    [
        (PF_IMAGE, "image", "takes current image", None),
        (PF_DRAWABLE, "drawable", "Input layer", None),
        (PF_SLIDER, "radius", "Radius", 5, (0, 500, 0.5)),
        # note extra tuple (min, max, step)
        (PF_SLIDER, "amount", "Amount", 5.0, (0, 10, 0.1)),
        (PF_RADIO, "mode", "Set Desauration mode: ", DESATURATE_LIGHTNESS,
            (
                 ("Lightness", DESATURATE_LIGHTNESS),
                 ( "Luminosity", DESATURATE_LUMINOSITY),
                 ("Average", DESATURATE_AVERAGE)
            )
         )
    ],
    [],
    extreme_unsharp_desaturation_options, menu="/Filters/Enhance")
```

Now I need to decide what options I'll let the user change. For the unsharp mask, I will let them change the radius and amount variables, and for the desauration I'll let them choose the mode.

The first thing I need to do is easy enough. I just need to add the arguments to the main function so that the plug-in is expecting them.

To do this simply add sensible variable names to the arguments for the function – radius, amout, and mode. Remember to comma seperate them.

```
def extreme_unsharp_desaturation_options(image, drawable, radius, amount, mode):
```

Then I go down to the parameters section of the register arguments and add those three variables as GUI inputs the user can play with. The great thing about gimpfu is that it comes with it's own GUI library that is set up to work with minimal fuss by the developer. Simply throw in a few details and it will build the GUI for you.

We will add 3 user input options, one for each variable. To decide which ones we'll use, we should consult the documentation: http://www.gimp.org/docs/python/ (http://www.gimp.org/docs/python/)

Scroll down to the section called Plug-in Framework and you'll see a whole bunch of options beginning PF_.

Each parameter is held in something that is a little bit like a list. If you remember from the last tutorial, a list is a collection of values, all assigned to the same label, and they are set by square brackets. Well, if you look closely at the parameters, they are nested within square brackets – and there is another type of grouping within that list. The first item in the parameter list is the collection in parentheses from PF_Image through to None, then there is a comma, telling us that we are moving on to item two in the list…But each of the things in parentheses looks like a list, too. Well it sort of is. These collections in parentheses are called tuples, and they are used in slightly different ways. We don't need to get bogged down in that for now, but all we need to know is that the parameters argument of the register function accepts a list of tuples. That sounds more complicated than it really is to a beginner, so just copy copy what I do for now.

Firstly, I need to take a value from the user for radius. Looking at the documentation, and remembering what I know from the PDB entry, I shouldn't pick the INT options, because radius is supposed to be a float. There is a FLOAT option, and if you are familiar with the different user interface options in GIMP, you might be able to picture what that one looks like. There are some others you might be able to recall as well, like PF_SLIDER.

There is a dummy script called pyui.py, created by Akanna Peck which demonstrates what each parameter looks like in a GUI and you can see what kinds of arguments they need. You can get a copy here: http://gimpbook.com/scripting/ (http://gimpbook.com/scripting/)

However Peck's script doesn't fit on my screen, and you might find the labelling of each parameter a little confusing as a beginner. So I have made two smaller scripts (https://gist.github.com/JacksonBates/44be64ffe89130f4a348) that show half of the parameters each, and I've labelled them a little clearer. Also, the variable names I've picked and a few other details are a little clearer for absolute beginners, so if you find those helpful you can download those from the Github Gist I've saved them at. Let's take a look at mine to see what each parameter looks like and to pick the most suitable one for the radius and amount.

Well, looking at the first half of parameters, it looks like PF_FLOAT will be the best option – the user can type a float in easily. So that might be the best choice. Looking at the second set, though, we have some other options. Firstly, spinner and adjustment do the same thing, so that means One of those or slider look promising too. Lets take a look at the code for all three to further evaluate which one will best suit our needs.

So Float takes the following settings: the variable name which is called by the main function (in my script this is called float_var), the label for the GUI (my script just calls this float), then a default value.

Both Spinner and Slider have the same first three items, variable, label, default, but then there is a final setting which is another one of those tuples we introduced earlier. This tuple has three elements to it: a minimum value, a maximum value, and a step value. This means for these two we can set maximum values so the user can't accidentally enter a number out of range and crash the script. The original unsharp mask plug in has a maximum radius of 500 and a maximum amout of 10. Setting these as our maximums will save the user from problems, making either of these a better choice than pf_float.

After that, it's just a usability choice, and the step doesn't matter much beyond picking something reasonable. The step value describes how much the number will increase with each click of the spinner or movement of the slider. Experiment with it until you find values that suit your needs.

So I will pick a slider for radius and amount, since those are the ones on the original unsharp mask. Pay attention to all parentheses and commas!

```
    (PF_SLIDER, "radius", "Radius", 5, (0, 500, 0.5)),
    (PF_SLIDER, "amount", "Amount", 5.0, (0, 10, 0.1)),
```

For desatuaration mode, I'll use the same method that the GIMP developers picked there, too. So I'll set it as a radio option.

Now, I can't pull these code snippets from the PDB, so we have to remember them, keep a note of them somewhere or refer to sample scripts like the ones Akanna Peck and I have made.

Notice that RADIO keeps the value and label for each radio button in it's own tuple…you can have as many tuples as you need for this, and to confuse matters this group of tuples is also in a tuple! To make mine more readable, I put each separate tuple on a new line. And remember to convert dashes to underscores, and count all the opening and closing parentheses very carefully, and include commas in all the right places. If you have bugs, this is most likely where you will find them, assuming there are no typos in function names or variables.

```
   (PF_RADIO, "mode", "Set Desauration mode: ", DESATURATE_LIGHTNESS,
               (
                    ("Lightness", DESATURATE_LIGHTNESS),
                    ( "Luminosity", DESATURATE_LUMINOSITY),
                    ("Average", DESATURATE_AVERAGE)
               )
            )
```

So that’s how we take some user input. It’s pretty straightforward once you do it a few times, I promise.

Related posts: Python Fu GIMP tutorials (https://jacksonbates.wordpress.com/python-fu-gimp-scripting-tutorial-pages/)

Here’s the code for the completed plug-in.

```python
#!/usr/bin/env python

# Tutorial available at: https://www.youtube.com/watch?v=5Ld8Todog5s
# Feedback welcome: jacksonbates@hotmail.com

from gimpfu import *

def extreme_unsharp_desaturation_options(image, drawable, radius, amount, mode):
    pdb.gimp_image_undo_group_start(image)
    threshold = 0
    pdb.plug_in_unsharp_mask(image, drawable, radius, amount, threshold)
    pdb.gimp_desaturate_full(drawable, mode)
    pdb.gimp_image_undo_group_end(image)


register(
    "python-fu-extreme-unsharp-desaturation-options",
    "Unsharp mask and desaurate image, with options",
    "Run an unsharp mask with variables set by user",
    "Jackson Bates", "Jackson Bates", "2015",
    "Extreme unsharp and desaturate options...",
    "RGB",
    [
        (PF_IMAGE, "image", "takes current image", None),
        (PF_DRAWABLE, "drawable", "Input layer", None),
        (PF_SLIDER, "radius", "Radius", 5, (0, 500, 0.5)),
        # note extra tuple (min, max, step)
        (PF_SLIDER, "amount", "Amount", 5.0, (0, 10, 0.1)),
        (PF_RADIO, "mode", "Set Desauration mode: ", DESATURATE_LIGHTNESS,
            (
                ("Lightness", DESATURATE_LIGHTNESS),
                ( "Luminosity", DESATURATE_LUMINOSITY),
                ("Average", DESATURATE_AVERAGE)
            )
        )
    ],
    [],
    extreme_unsharp_desaturation_options, menu="/Filters/Enhance")

main()
```

3 thoughts on “Python Fu #6: Accepting user input”

1. **KRZYSIUNET SAYS:** Now try to have only sliders. Eh😦 I haven’t found any workaround. Some input fields have fixed width, some are relative. Slider is relative, they take 100% of the window – from labels to right side. As there’s no other element to make the window wider, they are unusable, as they take a few px. I need to send them that bug.
   December 9, 2015 at 10:17 am • *Reply »*
   - **JACKSONBATES SAYS:** Hmmm, I haven’t noticed that before. Could it be a screen resolution problem?
     December 9, 2015 at 10:53 am • *Reply »*
   - **JACKSONBATES SAYS:** But I haven’t tried only sliders, I think. Worth reporting the bug, I think.
     December 9, 2015 at 10:55 am • *Reply »*