Python Fu #2: Functions tutorial

September 13, 2015November 30, 2015 | jacksonbates
See the other tutorials in this series. (https://jacksonbates.wordpress.com/python-fu-gimp-scripting-tutorial-pages/)

In the last tutorial we used the type() function. We can also create our own functions and it is one of the core ideas central to this type of programming. This tutorial is the text version of the following video:

Let's say I want to print some text to the screen:

```
>>>print "Hello, world!"
Hello, world!

>>>print "The old programming cliche!"
The old programming cliche!
```

And then lets for some reason decide I want to do that again and again. Rather than continually writing out all the print statements, I can create a function that does it for me.

```
>>>def hello_world():
...  print "Hello, world!"
...  print "The old programming cliche!"
```

So now all I need to do is call my function to do it's thing:

```
>>>hello_world()
Hello, world!
The old programming cliche!
```

You will notice that this is similar to the type function we used before, with the name of the function followed by parentheses (round brackets). But the type function had something in those parentheses, so what was that?

Well, some functions take an input that they work on. Those inputs are called arguments or parameters, and they come in the form of variables.

For example, I could create a simple program that takes a couple of numbers, adds them together, and then prints the result.

```
>>>def sum(a, b):
...  print a + b
```

Now to call that function, I can't just write sum(), or sum(a, b), I actually have to pass appropriate variables in to the parenthesis like this:

```
>>>sum(1, 2)
3
```

or:

```
>>>sum(1.0, 2.0)
3.0
```

Actually, you may have noticed that I can simply write sum(a, b) if those variables had already been defined. For example if we write:

```
>>>a = 1
>>>b = 2
>>>sum(a, b)
3
```

We could also write:

```
>>>sum("a", "b")
```

What result do you think that would have? Try it and see if you were right. If you understand why it behaved that way, you've learned everything we need so far.

In the next video, we'll start to dig into GIMPs Procedural Database to see what functions GIMP comes loaded with.

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