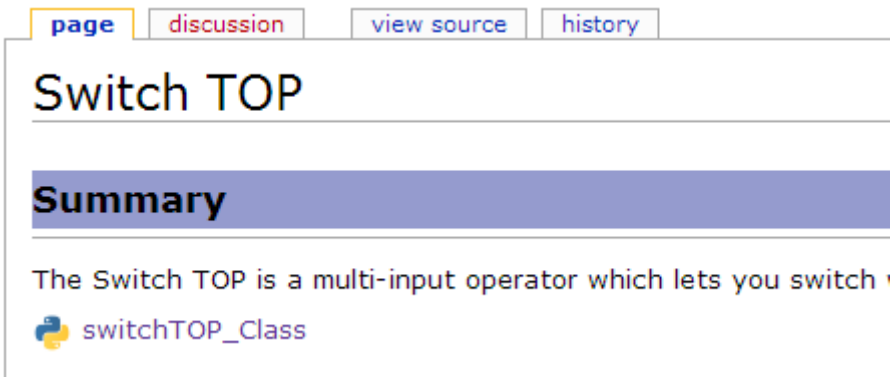
Matthew Ragan

Media Maker and Interactive System Designer

Understanding Referencing Part II | TouchDesigner

I love me some referencing. The more you work with TouchDesigner, the more you'll find that you need a solid understanding of how referencing connects the various parts of your network. Better still is getting a better handle on how Python scripting works in TouchDesigner – especially dot notation. At the heart of what we're after is making sure that our networks can start to feel a little more interconnected. Hard coding values makes for tedious programming, especially if you're building something you'd like to reuse. By starting to think about how to build some logic into the system we're making we begin to build reusable tools. If you're still getting a handle on how referencing works, then it's a good idea to get started here (Understanding Referencing) to get your bearings.

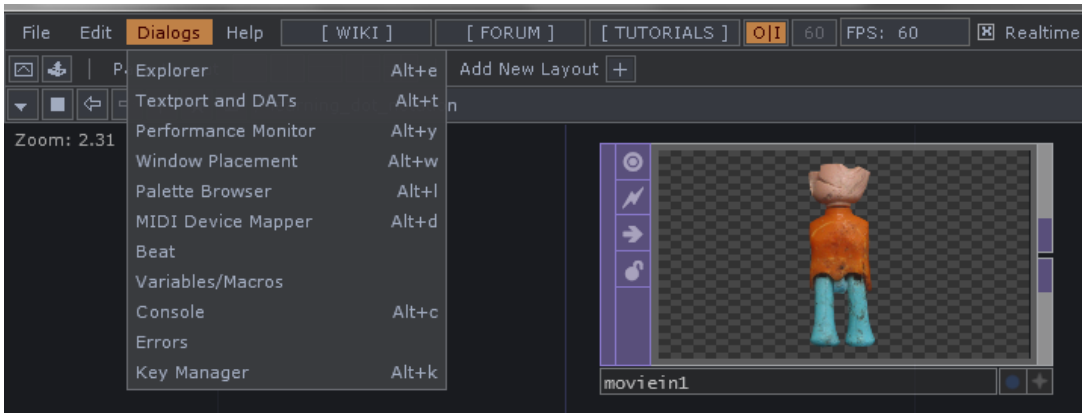
Let's start with a simple example. I've got five images that I'd like to cycle through when I click a button. I know that I can use a Count CHOP and a Switch TOP to help with this but how can I make it so that my Count CHOP knows when I add another input to my Switch TOP? In a perfect world I could add or remove images or movies, and the network would just "know" what was going on, right? So how can we program something like this? Let's start by taking a moment to visit my favorite part of working with TouchDesigner, the wiki. Specifically we need to look at our Switch TOP's page.



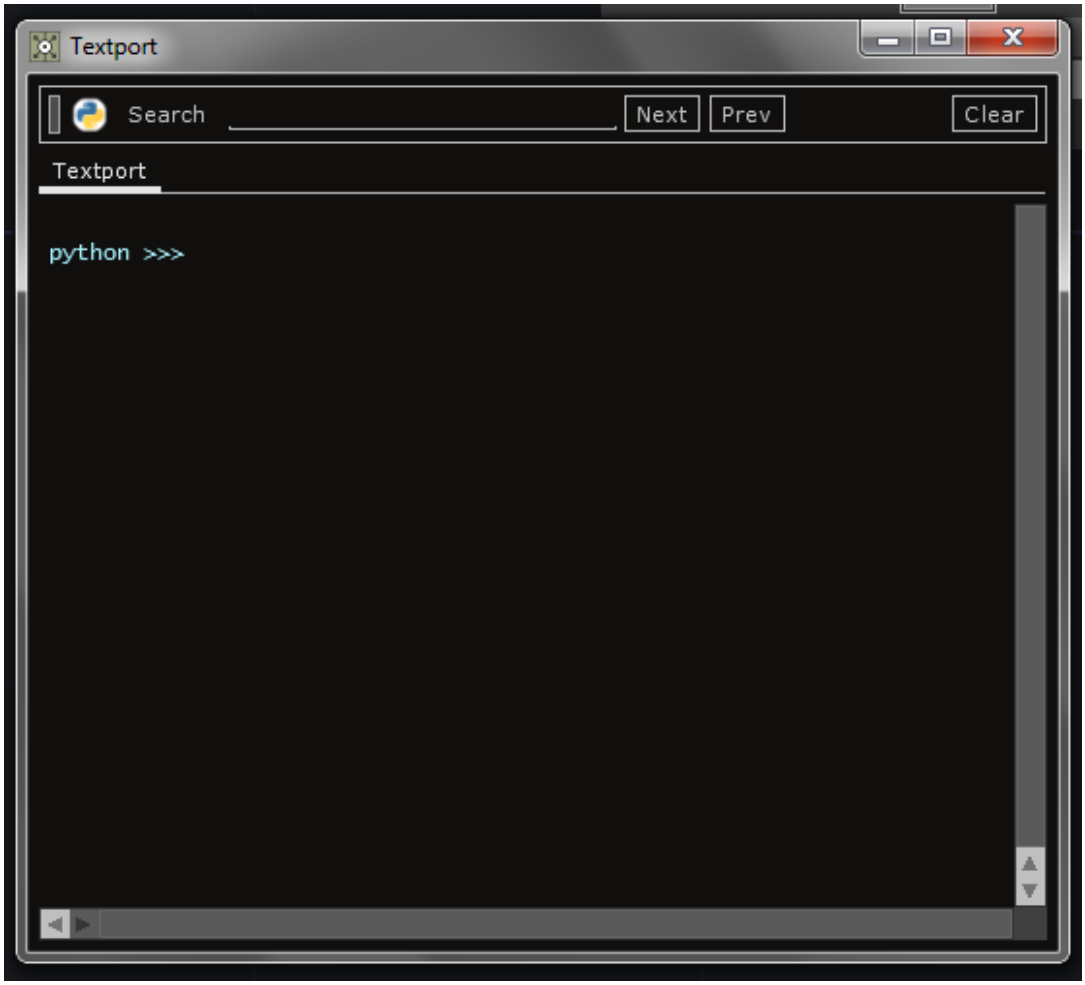
The most important part of this page for us today is here at the top where it says "switchTOP_Class" next to the Python symbol. In object-oriented programming a class is a kind of "extensible template for creating objects." The exciting part of knowing this is that a class comes with all sorts of methods and values that we can quickly call when scripting. With that in mind let's take a closer look at the Switch TOP Class page. At this

point you're either beside yourself with excitement, or weeping with confusion. Let's take a look at a quick example, and see if we can make some sense out of what we're reading.

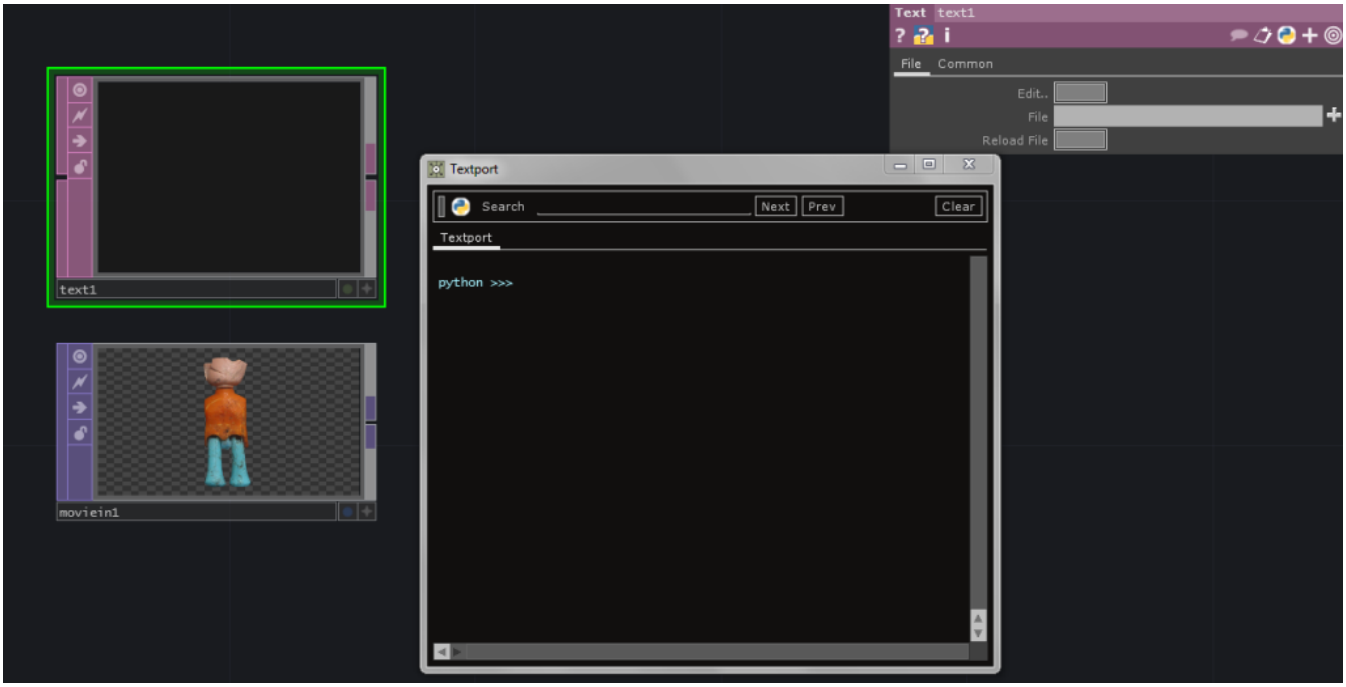
One of the first things on the Switch TOP Class page is a sentence that says "This class inherits from the TOP class." This means that there are several things that work for the Switch TOP that also work for all TOPs. This means we can practice on a regular movie in TOP, and what we learn should also work for our Switch TOP. Okay, let's open up a new network in TouchDesigner, and let's add a movie in TOP and a Text DAT. We'll also need to open up our Textport.



When you open your textport you should see a floating window that looks like this:

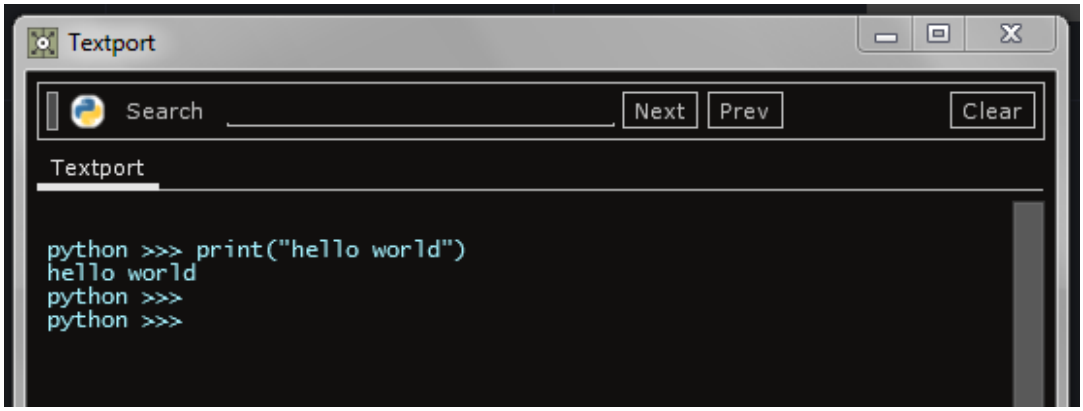


Great! With your Movie In TOP, Text DAT, and Textport you should be looking at something like this:



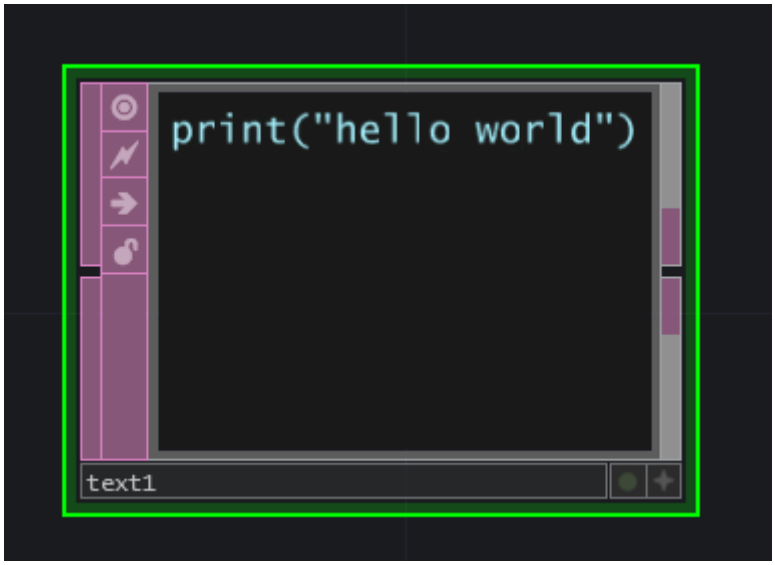
In our Text DAT we're going to write a very simple little script that's going to print in our textport. In Python we can print something to the textport with the command ***print()***. Whatever we put in the parenthesis gets printed to the textport. You can practice this in the textport, just make sure that you use quotes around your text. For example:

print("hello world")

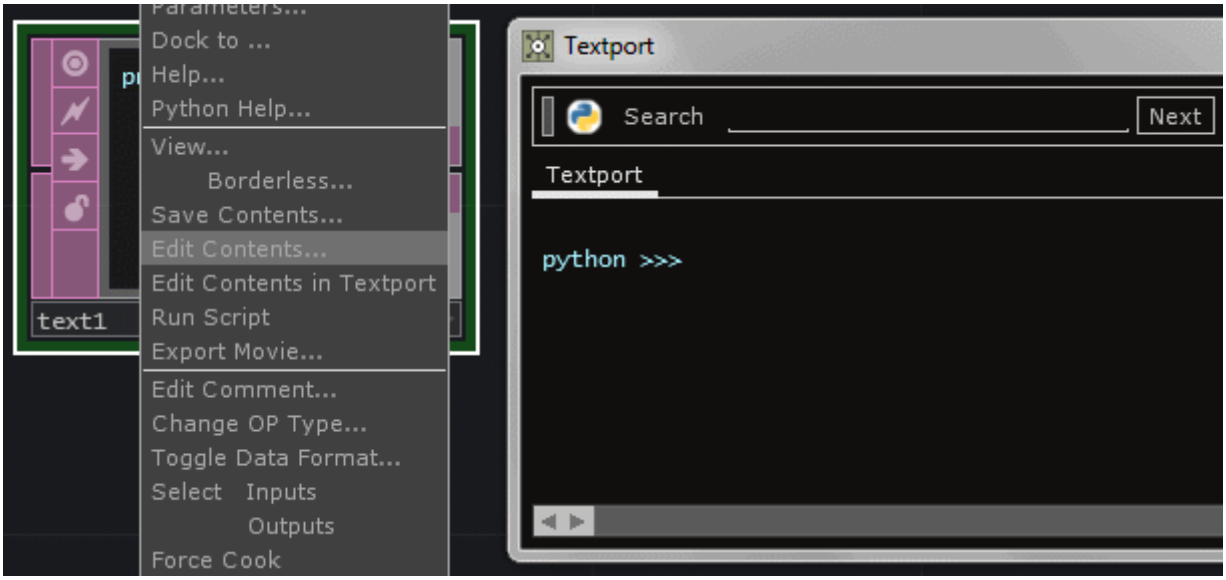


We can also complete this same operation using our text DAT. In the text DAT write:

print("hello world")



Now right click on the DAT, and select “Run Script” from the drop down menu. You should see the text appear in your textport:



Congratulations, you just ran a Python Script from a Text DAT. We’re going to use our ability to print to the textport to learn a few more things about our TOP class. We should still have a Movie In TOP in our network. My Movie In TOP is called “moviein1”, knowing the name of our operators is how we call point to them when we’re referencing. Looking back to the general TOP Class Wiki Page, I can see that there’s a member called ***width*** and one called ***height***.

Members

width	**(Read Only)** Texture width, measured in pixels.
height	**(Read Only)** Texture height, measured in pixels.

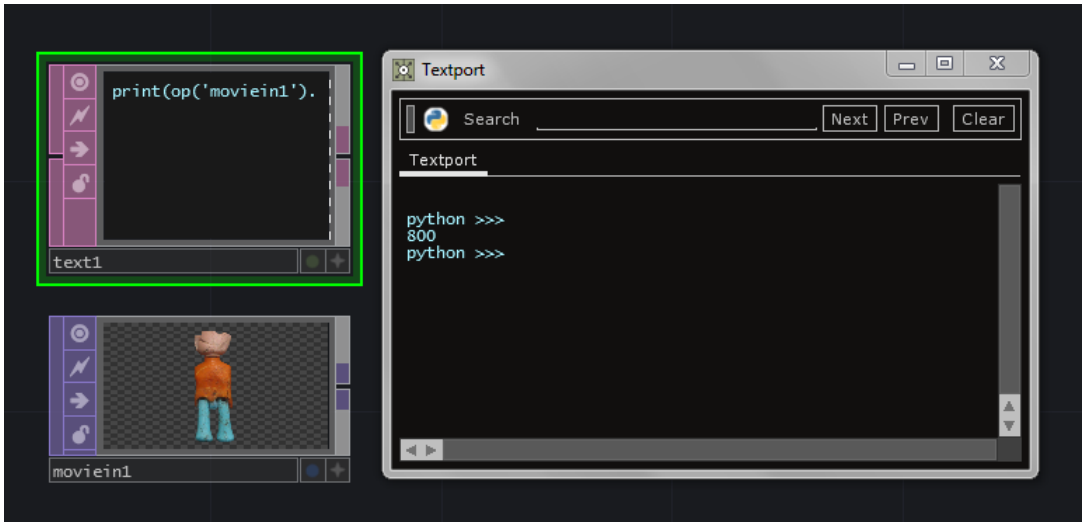
For this next step we’re going to use dot notation to get to the width and height information about our operator. For example, I can get that information by starting with the operator name, followed by a dot, followed by the member I want:

op(‘moviein1’).width

Let's copy that exact line of code into our Text DAT. Next right click, and select Run Script. Nothing happened… what gives?! In this case, just because we asked for that value, doesn't mean that TouchDesigner knew that we wanted it to be printed to the text port. In order to see these values in the textport we need to encapsulate our script in the print command. When you do this, you should have a script that looks like this:

print(op('moviein1').width)

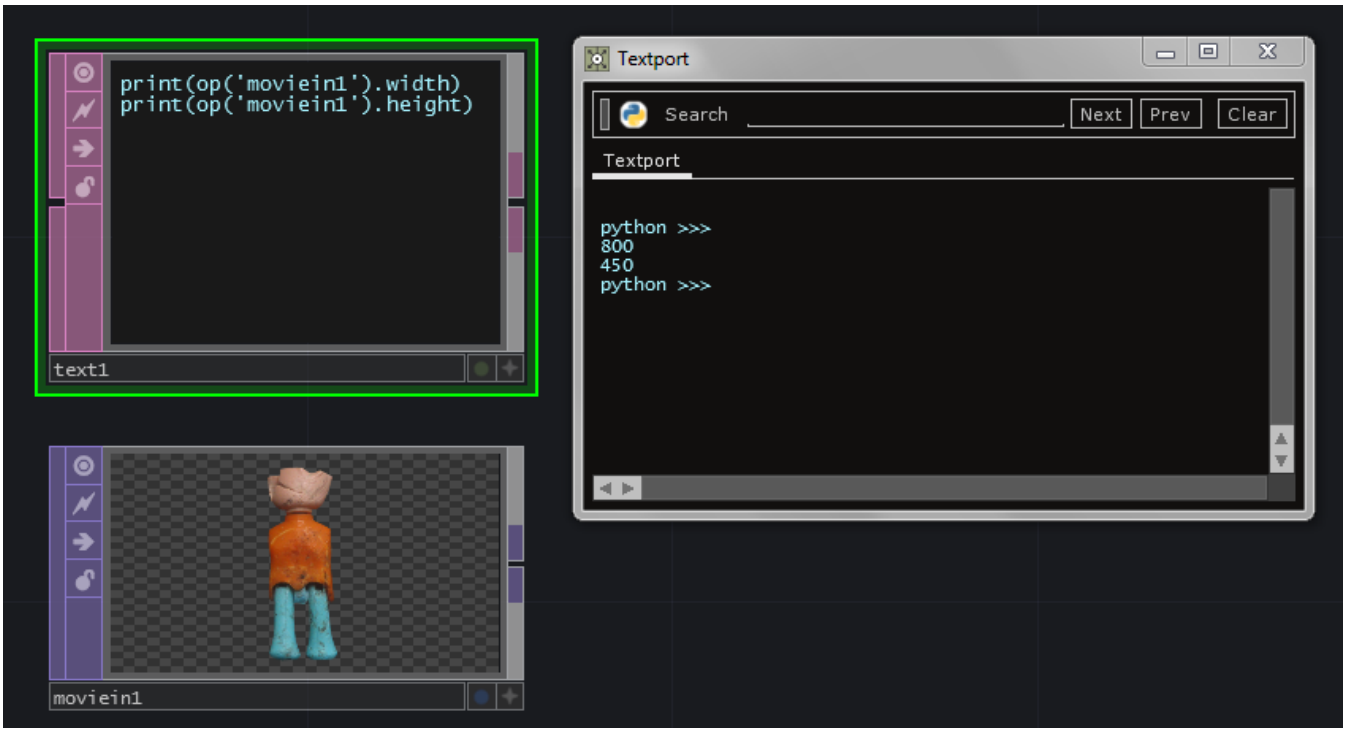
Now if you run the script you should see:



Let's try one more. Let's add one more line of code:

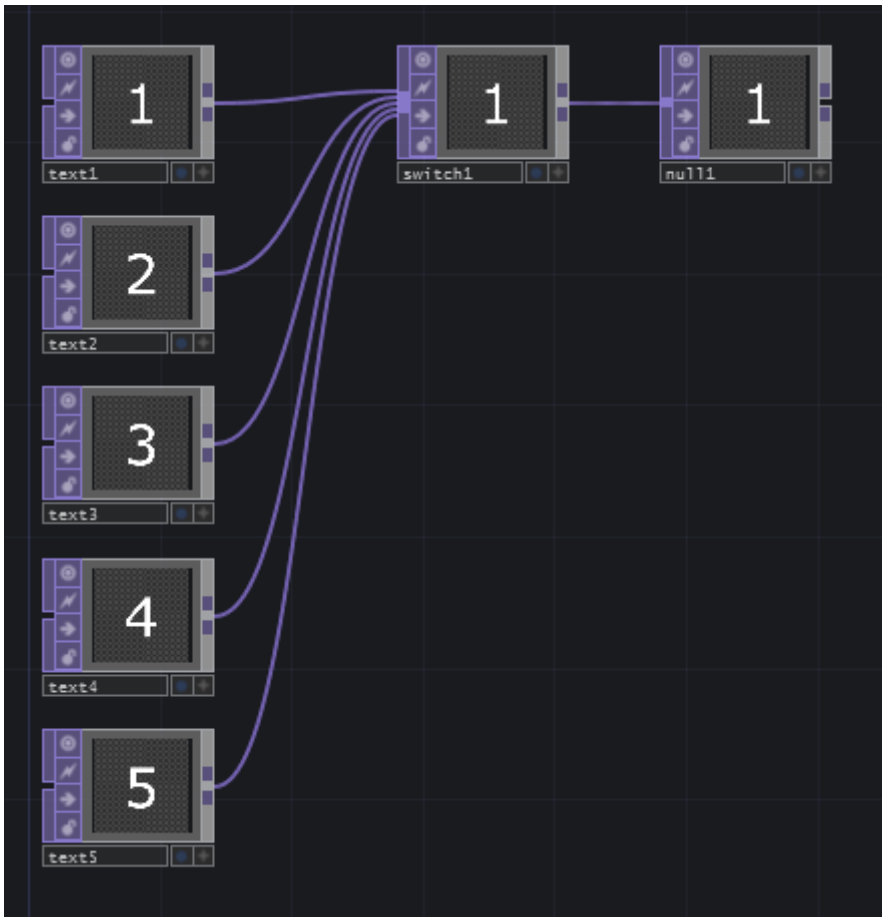
print(op('moviein1').height)

Let's run the script one more time and you should have something that looks like this:



Believe it or not, we just made some huge progress. Understanding how dot notation works gives you access to a HUGE treasure trove of information that you can use when scripting or writing references.

Let's return to our example and see if we can't use what we've just learned. Let's start by adding a few things to our network. I'm going to add five Text TOPs, and I'm going to set them to be the numbers 1 – 5. Next I'm going to wire all of those to a Switch TOP, and then finally to a Null TOP,



Now let's add a Text DAT to our network, and then take a quick moment to go back to our Switch TOP Class Page. Of particular interest to me is the Connections subsection:

Connection

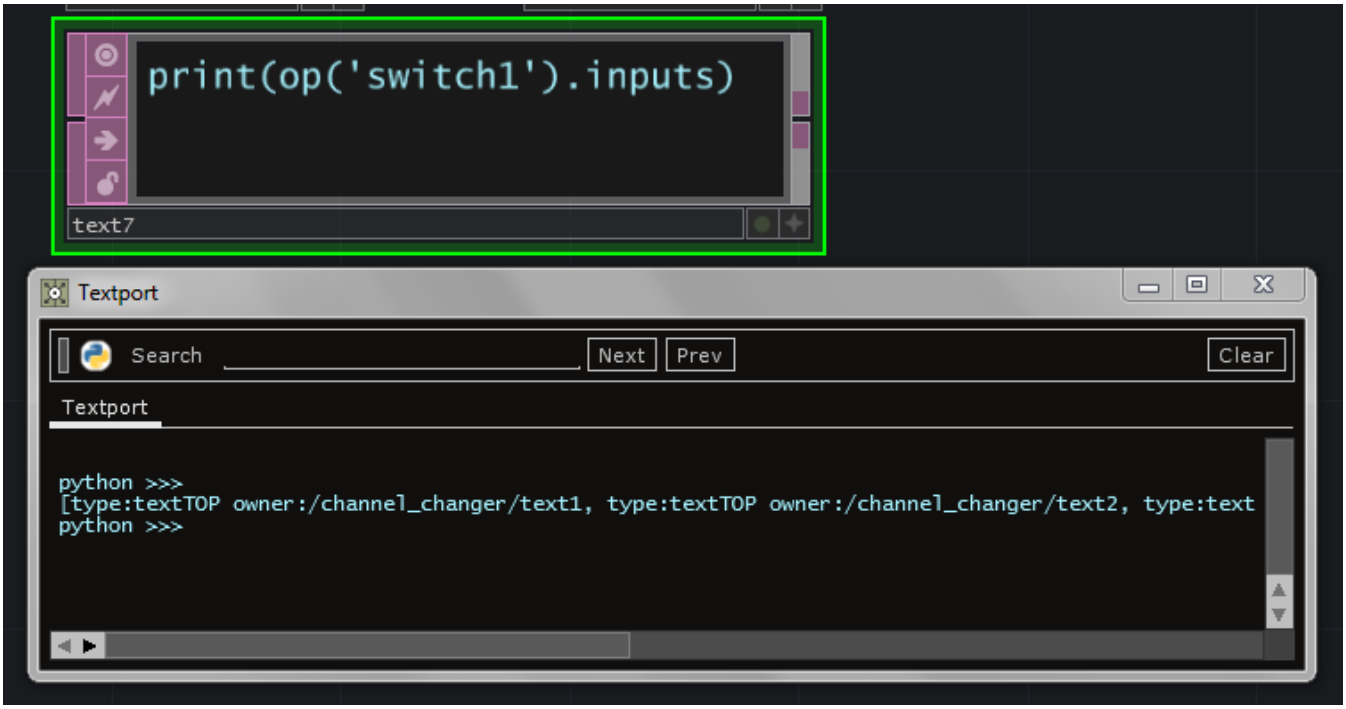
See also the OP.parent() function. To connect components together see COMP_Class#Connection section.

inputs	**(Read Only)** List of input operators to this operator.
outputs	**(Read Only)** List of output operators from this operator.
inputConnectors	**(Read Only)** List of input connectors asssociated with this operator.
outputConnectors	**(Read Only)** List of output connectors associated with this operator.

Using what we just learned, write the following script into your text DAT:

print(op('switch1').inputs)

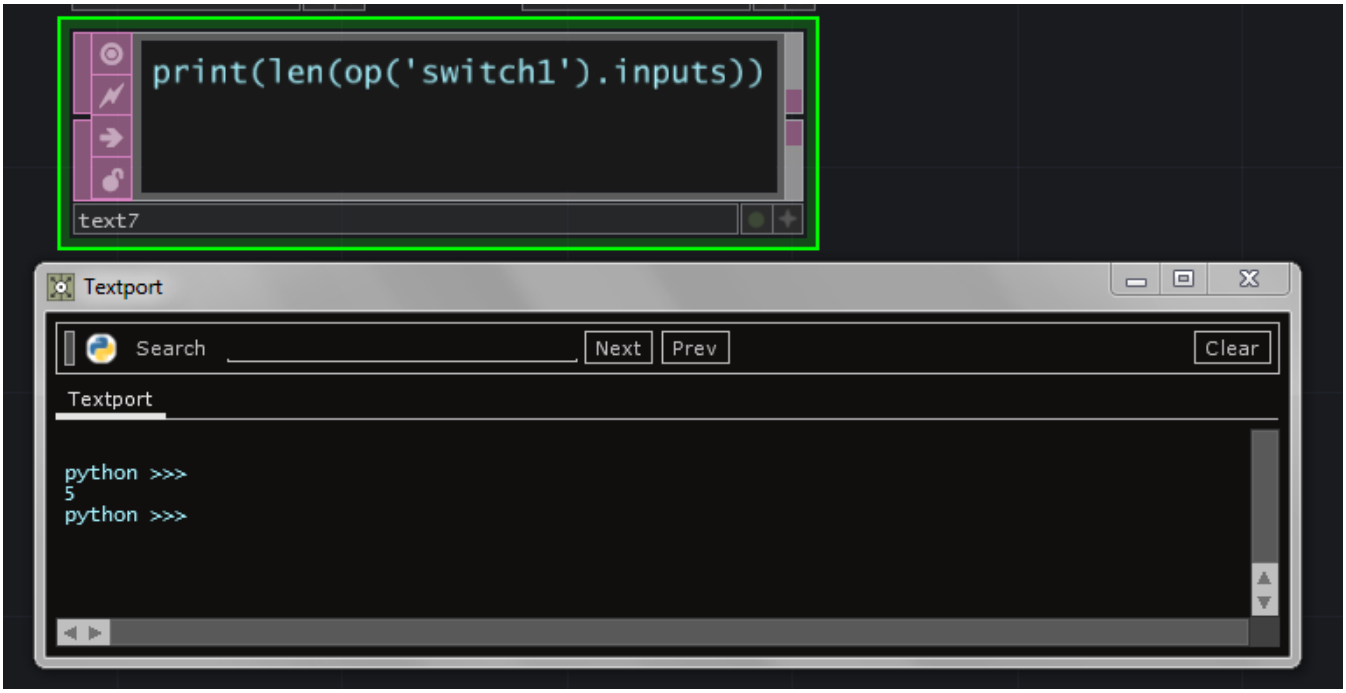
When you run this script you should see something like this:



Well, that doesn't look useful at all, does it? In fact, it's tremendously useful! Python uses lists to store information, and what we've printed out is a list of connections to the Switch TOP. We don't really want all of this information that we're getting about the inputs, we just want to know how many connections there are. As luck would have it, there's a way to get just that information. In Python we can ask for the length (the number of entries) of a list with the ***len()*** command. Alright, let's make a quick change to our script and add the length command. You should have a script now that looks like this:

print(len(op('switch1').inputs))

When you run the script you should see this:

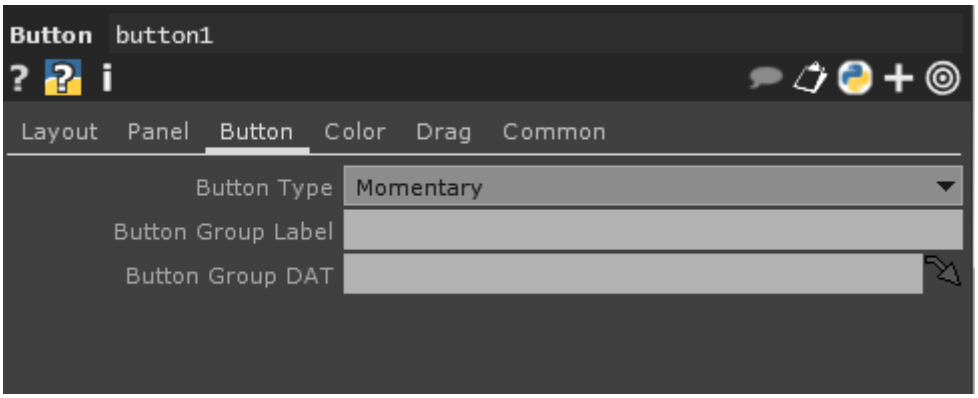


Alright! Now we can start to really make some magic happen. Using ***.inputs*** and the ***len()*** command we're able to now see the number of connections to our switch TOP. If you disconnect one of the inputs and run

the command again you'll see the number go down. Add a few more inputs, run the command, and you'll see the number go up.

How do we use this? Okay, well let's think back to where we started. We want to use a button to toggle through all of our inputs, and then cycle back to the front of the line. Let's add a Button component to our Network, and a Count CHOP. Make sure to wire the Button to the top most input on the Count CHOP:

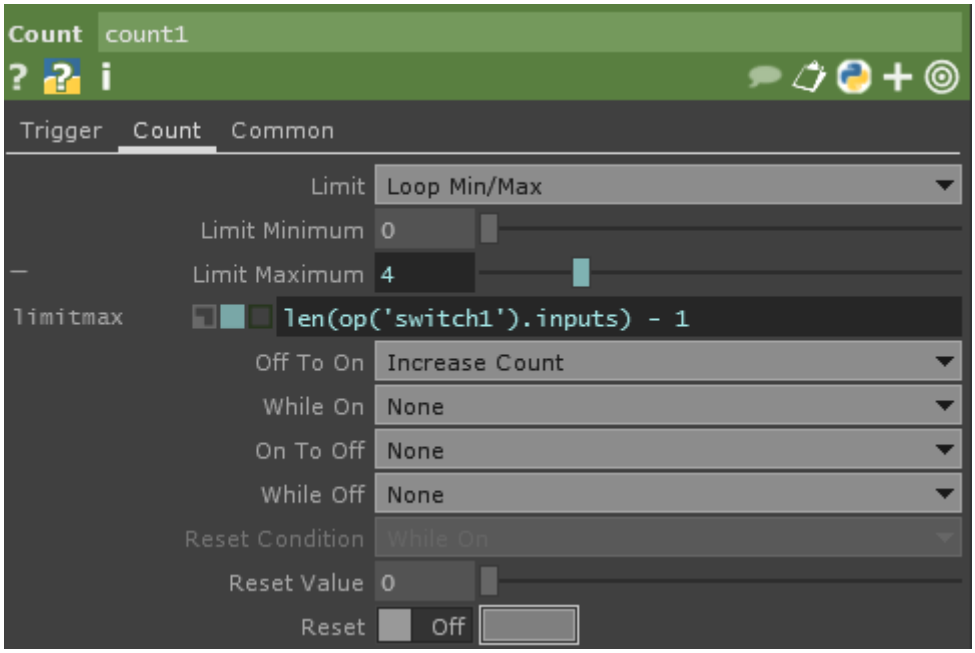
Let's also make sure that the button is set to momentary:



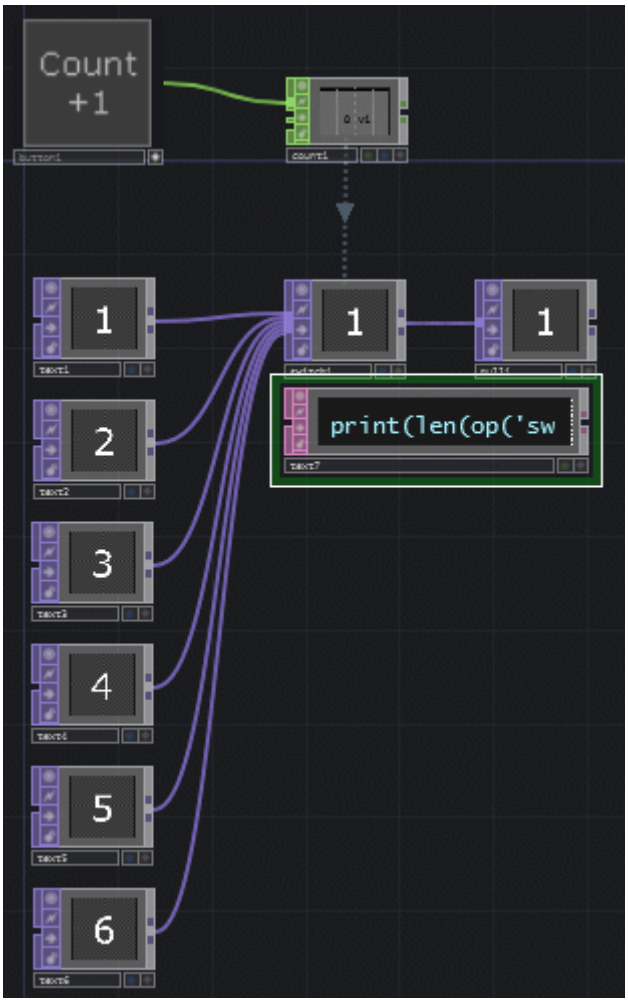
Looking at our Count CHOP we want to set the Limit Method to "Loop Min/Max." For the Limit Maximum value we're going to use the script that we just wrote (only without the print command). We're also going to subtract 1 from that number. We do this because 0 is still a valid input number for our switch, so while there are five total inputs those are associated with the numbers 0 1 2 3 4. The reference in the Limit Maximum field should look like this:

len(op('switch1').inputs) – 1

Your Count CHOP parameters dialog should look something like this:



And now we have a little bit of black magic happening. If you add more inputs to your switch you'll see the Limit Maximum number go up automatically, remove some inputs, and you'll see it go down. Clicking on the button should cycle you through all of the inputs:



Now you know a little more about referencing, dot notation, and how to find more information in the wiki. Happy programming!

Check out the TOE file if you'd like to see what I made: Referencing Part 2

This entry was posted in How-To, Programming, Software, TouchDesigner and tagged Arizona, Arizona State University, ASU, Derivative, Grad School, graduate school, Live Performance, programming, Python, TouchDesigner, TouchDesigner Tutorial on June 27, 2014 [https://matthewragan.com/2014/06/27/understanding-referencing-part-ii-touchdesigner/] .

One thought on “Understanding Referencing Part II | TouchDesigner”

Pingback: Python in TouchDesigner | Writing Python References | TouchDesigner | Matthew Ragan

Comments are closed.