

# Setting Vertex Colors with Python:

Tutorial by [Scott B.](#)  
The [Blender 3D Club](#).

---



Someone came to my site a while back looking for a way to set vertex colors using python. Here's a brief tutorial that shows how you can do it.

```
1
2 import Blender
3 from Blender import *
4
5
6 print '===== starting ====='
7
8 mesh = Blender.Mesh.Get('Cube')
9
10
11 mesh.addColorLayer('layer1')
12 colnames = mesh.getColorLayerNames()
13
14 print 'vertexcolors =',
15 print mesh.vertexColors
16
17 faces = mesh.faces
18
19
20
21 for d in faces[0].verts:
22     print d
23 for d in faces[0].col:
24     print d
25
26 faces[0].col[0].r = 0
27 faces[0].col[1].g = 0
28 faces[0].col[2].b = 0
29
30 mesh.update()
31
```

Here is the complete script. It shows how to change a few of the vertices in a face of a cube.

```
import Blender
from Blender import *
```

These first few lines import the Blender Module.

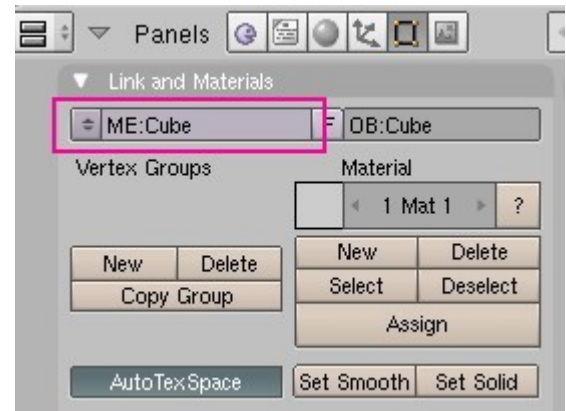
I started with a basic Blender cube. **Space Bar - Add - Cube**. I left the default name for the mesh as 'Cube'.

```
mesh = Blender.Mesh.Get('Cube')
```

This line assigns our mesh to the variable 'mesh'. Make sure you replace 'Cube' with the name of your Blender *mesh*.

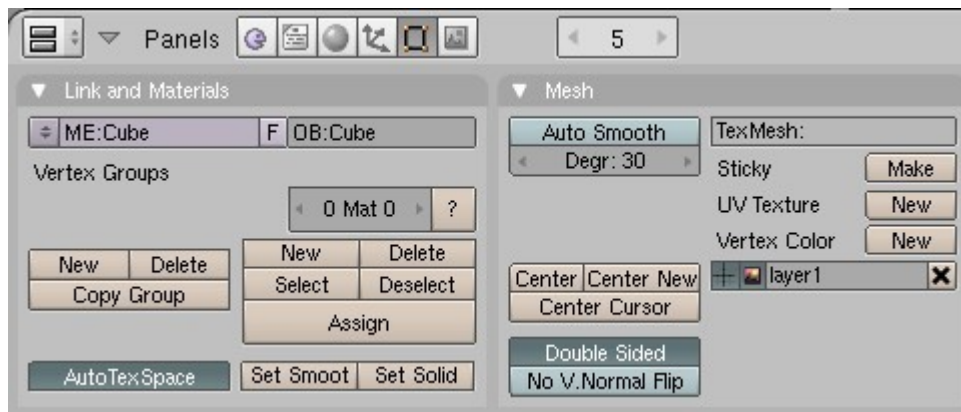
---

The name of your mesh will be located here. With your object selected, switch to '**Editing (F9)**'. You'll see it listed under the **Links and Materials** tab.



```
mesh.addColorLayer('layer1')
colnames = mesh.getColorLayerNames()
```

The next line adds a new vertex color layer and names it 'layer1'. The name can be anything. You'll see this name updated under your **Mesh** tab while you are in Editing mode with your object once the script completes.



The next line gets the list of all of the vertex color layers and assigns them to the variable *colnames*. This line is not necessary to run the script, since we are only working with the newly created layer and there are no other layers at this time.

```
print 'vertexcolors =',
print mesh.vertexColors
```

These next two lines simply print out the current value of the *vertexColors* variable. This is a boolean value which shows whether vertex colors are active or not for this mesh. It will equal 0 or 1. 1 meaning vertex colors are active. When we created our new Color Layer, this value automatically changed from 0 to 1.

These lines are not needed for this script, but may be useful if you need to know if a mesh has vertex colors active before you begin to work with it.

---

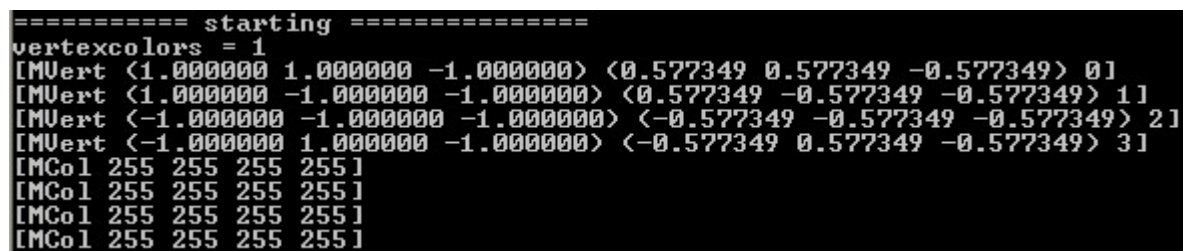
```
faces = mesh.faces
```

This next line assigns all of the faces in our mesh to the variable *faces*.

---

```
for d in faces[0].verts:
    print d
for d in faces[0].col:
    print d
```

This next few lines, again, are not necessary for the script to run, but they show you a sample of what you have to work with in your mesh. All of the faces in a mesh are numbered starting with 0. A mesh with 4 faces will be numbered from 0 - 3. The lines above print out a sample from face # 0 only.

A terminal window with a black background and white text. The text shows the output of a script. It starts with a separator line '==== starting ====='. Below that, it prints 'vertexcolors = 1'. Then it prints four lines of vertex data: '[MVert (1.000000 1.000000 -1.000000) (0.577349 0.577349 -0.577349) 0]', '[MVert (1.000000 -1.000000 -1.000000) (0.577349 -0.577349 -0.577349) 1]', '[MVert (-1.000000 -1.000000 -1.000000) (-0.577349 -0.577349 -0.577349) 2]', and '[MVert (-1.000000 1.000000 -1.000000) (-0.577349 0.577349 -0.577349) 3]'. Finally, it prints four lines of color data: '[MCol 255 255 255 255]', '[MCol 255 255 255 255]', '[MCol 255 255 255 255]', and '[MCol 255 255 255 255]'.

```
==== starting =====
vertexcolors = 1
[MVert (1.000000 1.000000 -1.000000) (0.577349 0.577349 -0.577349) 0]
[MVert (1.000000 -1.000000 -1.000000) (0.577349 -0.577349 -0.577349) 1]
[MVert (-1.000000 -1.000000 -1.000000) (-0.577349 -0.577349 -0.577349) 2]
[MVert (-1.000000 1.000000 -1.000000) (-0.577349 0.577349 -0.577349) 3]
[MCol 255 255 255 255]
[MCol 255 255 255 255]
[MCol 255 255 255 255]
[MCol 255 255 255 255]
```

```
for d in faces[0].verts:
    print d
```

These lines print out a list in the console of all of the vertices in face #0.

```
for d in faces[0].col:
    print d
```

These lines will print out all of the color values of each vertex in face #0. Values are from 0 - 255. (0 = 0% and 255 = 100%). The values will be for **r**, **g**, **b** and **a**.

```
r = red
b = blue
g = green
a = alpha
```

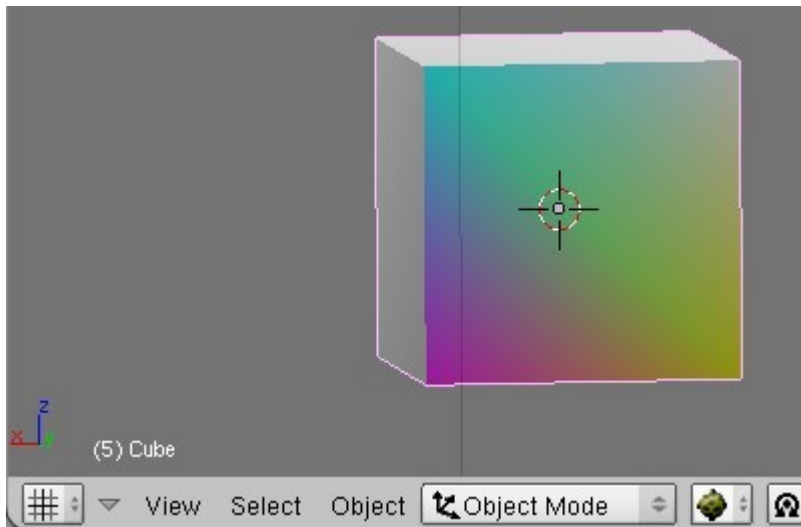
---

The default color values are 255 which means each vertex will be white.

```
faces[0].col[0].r = 0
faces[0].col[1].g = 0
faces[0].col[2].b = 0
```

These lines alter the default values for 3 vertices from 255 to 0.

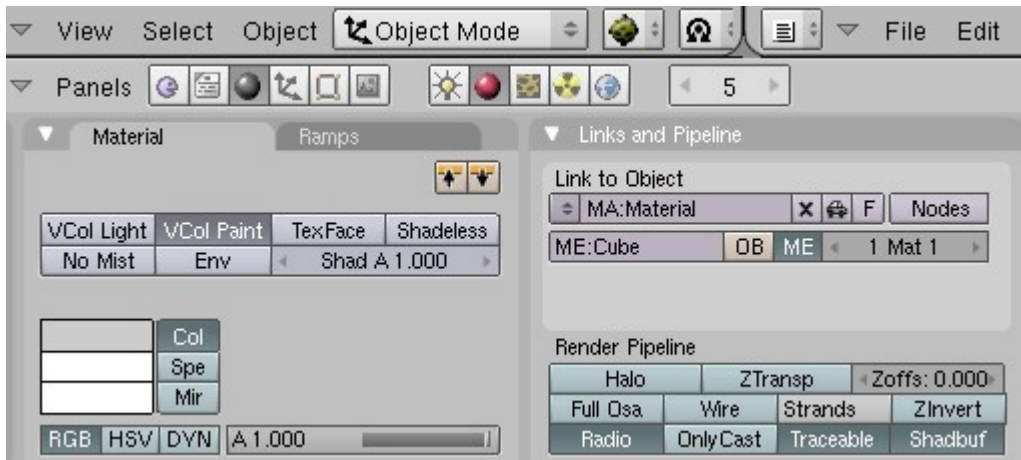
```
col[0].r represents the red color value from vertex # 0.
col[1].g represents the green color value from vertex # 1.
col[2].b represents the blue color value from vertex # 2.
```



The result on face #0 looks like this. You must be in the Textured view mode (ALT-Z) in order to see the vertex coloring.

---

In order to see vertex coloring in your rendered image, assign a new material to your object.



Under the Material tab, click on the button labeled VColPaint. Your vertex coloring will now show up in your rendered images.



Good luck with your modeling.  
[Scott](#)