





Grove - Chainable RGB LED is based on P9813 chip which is a full-color LED driver. It provides 3 constant-current drivers as well as modulated output of 256 shades of gray. It communicates with a MCU using 2-wire transmission (Data and Clock). This 2-wire transmission can be used to cascade additional **Grove - Chainable RGB LED** modules. The built-in clock regeneration enhances the transmission distance. This Grove module is suitable for any colorful LED based projects.

Version

Revision	Descriptions	Release	How to Buy
v1	Initial public release (beta)	May 5, 2011	<div style="background-color: #27ae60; color: white; padding: 5px; display: inline-block;">Get One Now </div> [https://www.seeedstudio.com/Chainable-RGB-LED-p-850.html]
v2	Replace P9813S16 with P9813S14 and change Grove connector from Vertical to horizontal	Apr 19, 2016	<div style="background-color: #27ae60; color: white; padding: 5px; display: inline-block;">Get One Now </div> [https://www.seeedstudio.com/%E2%80%93Chainable-RGB-Le-p-2903.html]



Specifications

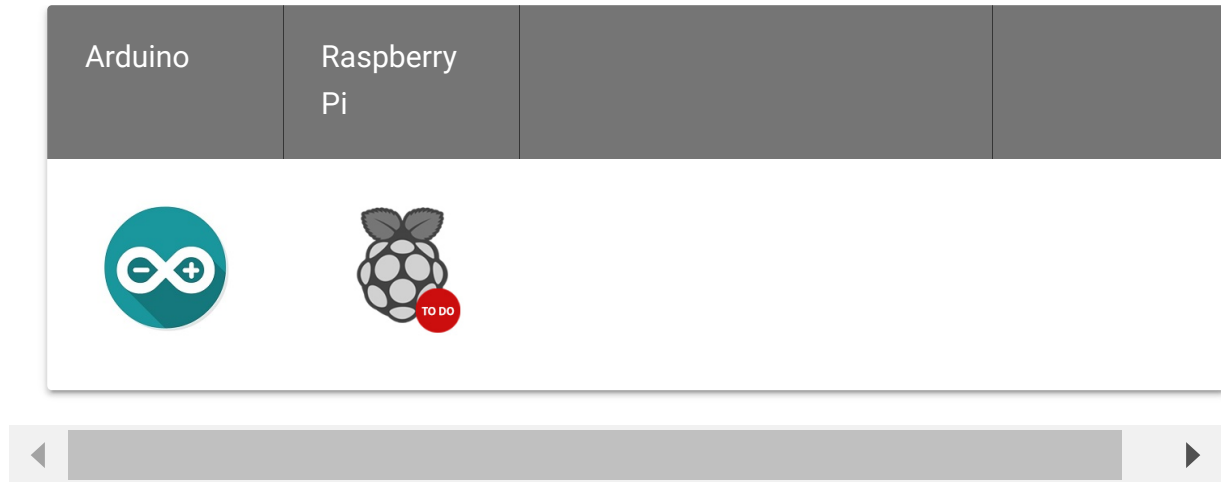
- Operating Voltage: 5V
- Operating Current: 20mA
- Communication Protocol: Serial



Tip

More details about Grove modules please refer to [Grove System](https://wiki.seeedstudio.com/Grove_System/)
[\[https://wiki.seeedstudio.com/Grove_System/\]](https://wiki.seeedstudio.com/Grove_System/)

Platforms Supported



Caution

The platforms mentioned above as supported is/are an indication of the module's software or theoretical compatibility. We only provide software library or code examples for Arduino platform in most cases. It is not possible to provide software library / demo code for all possible MCU platforms. Hence, users have to write their own software library.

Usage

Play with [Arduino](#) [/Arduino]

When you get Grove - Chainable RGB LED, you may think how I can light up it. Now we will show you this demo: all colors of RGB cycles in a uniform way.

To complete this demo, you can use one or more Grove - Chainable RGB LED. Note that the IN interface of one Grove - Chainable RGB LED should be connect to D7/D8 of [Grove - Base Shield](#) [/Base_Shield_V2] and its OUT interface connect to IN interface of another Grove - Chainable RGB LED, chainable more LED in this way.

- Download [Chainable LED Library](https://github.com/pjpmarques/ChainableLED) [https://github.com/pjpmarques/ChainableLED] and install it to Arduino Library. There is the course about [how to install Arduino Library](#) [/How_to_install_Arduino_Library] in wiki page.
- Open the example CycleThroughColors by the path:File->Examples->ChainableLED_master and upload it to Seeeduino.

```
1  /*
2   * Example of using the ChainableRGB Library for control
3   * This code cycles through all the colors in an uniform
4   */
5
6
7  #include <ChainableLED.h>
8
9  #define NUM_LEDS 5
10
11 ChainableLED leds(7, 8, NUM_LEDS);
12
13 void setup()
14 {
15     leds.init();
16 }
17
18 float hue = 0.0;
19 boolean up = true;
20
21 void loop()
22 {
23     for (byte i=0; i<NUM_LEDS; i++)
24         leds.setColorHSL(i, hue, 1.0, 0.5);
25
26     delay(50);
27
28     if (up)
29         hue+= 0.025;
30     else
31         hue-= 0.025;
```

```

32
33     if (hue>=1.0 && up)
34         up = false;
35     else if (hue<=0.0 && !up)
36         up = true;
37 }

```

You can observe this scene: colors of five LED will gradient consistently.

Extended application: Based on [Chainable LED Library](https://github.com/pjpmarques/ChainableLED)

[<https://github.com/pjpmarques/ChainableLED>], we have designed this demo: RGB color varies with the temperature measured by Grove - temperature. The RGB color vary from green to red when the temperature is from 25 to 32. The test code is shown below. Do it if you are interested in it.

```

1     // demo of temperature -> rgbLED
2     // temperature form 25 - 32, rgbLed from green -> red
3     // Grove-temperature plu to A0
4     // LED plug to D7,D8
5
6     #include <Streaming.h>
7     #include <ChainableLED.h>
8
9     #define TEMPUP 32
10    #define TEMPDOWN 25
11
12    ChainableLED leds(7, 8, 1); // connect to pin7 and p
13
14    int getAnalog() // get value from A0
15    {
16        int sum = 0;
17        for(int i=0; i<32; i++)
18        {
19            sum += analogRead(A0);
20        }

```

```
21
22     return sum>>5;
23 }
24
25 float getTemp() // get temperature
26 {
27     float temperature = 0.0;
28     float resistance = 0.0;
29     int B = 3975; //B value of the thermistor
30
31     int a = getAnalog();
32
33     resistance = (float)(1023-a)*10000/a; //get the
34     temperature = 1/(log(resistance/10000)/B+1/298.15);
35     return temperature;
36 }
37
38 void ledLight(int dta) // Light Led
39 {
40
41     dta = dta/4; // 0 - 255
42
43     int colorR = dta;
44     int colorG = 255-dta;
45     int colorB = 0;
46
47     leds.setColorRGB(0, colorR, colorG, colorB);
48 }
49
50 void setup()
51 {
52     Serial.begin(38400);
53     cout << "hello world !" << endl;
54 }
55
56 void loop()
57 {
58     float temp = getTemp();
59     int nTemp = temp*100;
60
61     nTemp = nTemp > TEMPUP*100 ? TEMPUP*100 : (nTemp
```

```
62     nTemp = map(nTemp, TEMPDOWN*100, TEMPUP*100, 0, 100);
63     ledLight(nTemp);
64     delay(100);
65 }
```

Play with Codecraft

Hardware

Step 1. Connect Grove - Chainable RGB LED to port D7 in a Base Shield

Step 2. Plug the Base Shield to your Seeeduino/Arduino.

Step 3. Link Seeeduino/Arduino to your PC via an USB cable.

Software

Step 1. Open [Codecraft](https://ide.chmakered.com/) [https://ide.chmakered.com/], add Arduino support, and drag a main procedure to working area.

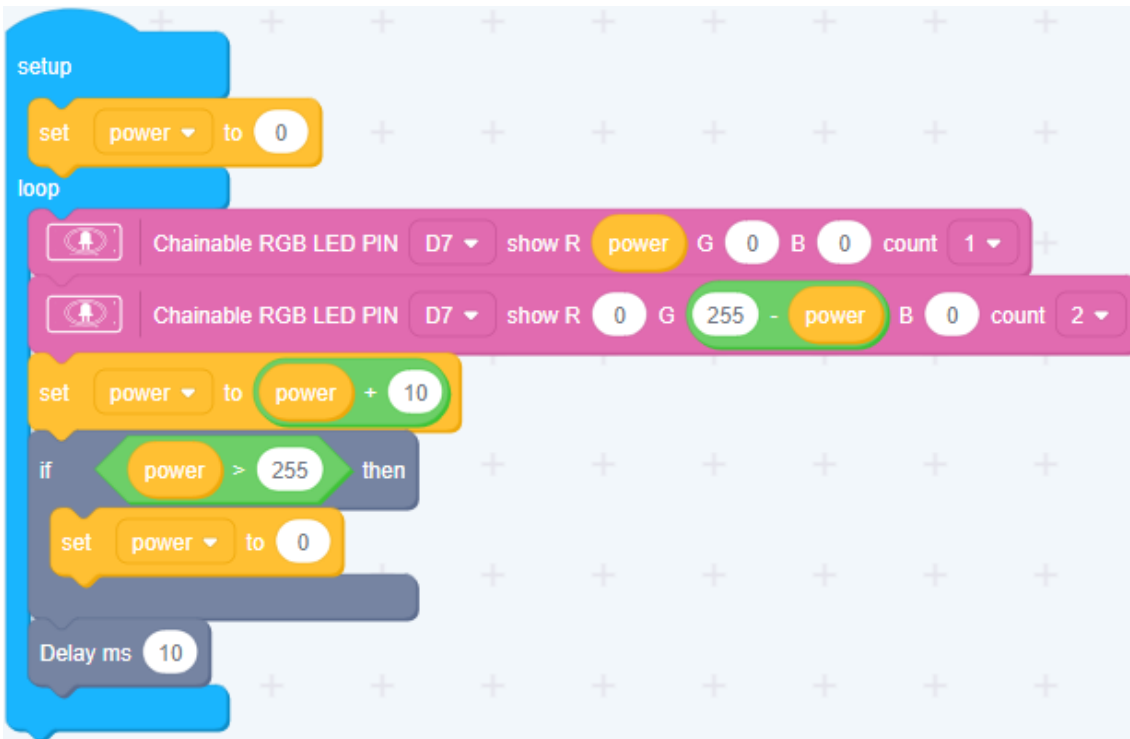


Note

If this is your first time using Codecraft, see also [Guide for Codecraft using Arduino](#)

[https://wiki.seeedstudio.com/Guide_for_Codecraft_using_Arduino/].

Step 2. Drag blocks as picture below or open the cdc file which can be downloaded at the end of this page.



Upload the program to your Arduino/Seeeduino.



Success

When the code finishes uploaded, you will see the LED fade in and fade out.

Play with Raspberry Pi



Attention

If you are using **Raspberry Pi with Raspberrypi OS >= Bullseye**, you have to use this command line **only with Python3**.

- 1.You should have got a raspberry pi and a grovepi or grovepi+.
- 2.You should have completed configuring the development enviroment, otherwise follow [here](#) [/GrovePi_Plus].
- 3.Connection

- Plug the sensor to grovepi socket D7 by using a grove cable.

4. Navigate to the demos' directory:

```
cd yourpath/GrovePi/Software/Python/
```

- To see the code

```
nano grove_chainable_rgb_led.py # "Ctrl+x" to exit
```

```

1  import time
2  import grovepi
3
4  # Connect first LED in Chainable RGB LED chain
5  # In: CI,DI,VCC,GND
6  # Out: CO,DO,VCC,GND
7  pin = 7
8
9  # I have 10 LEDs connected in series with the f
10 # First LED input socket connected to GrovePi,
11 numleds = 1
12
13 grovepi.pinMode(pin,"OUTPUT")
14 time.sleep(1)
15
16 # Chainable RGB LED methods
17 # grovepi.storeColor(red, green, blue)
18 # grovepi.chainableRgbLed_init(pin, numLeds)
19 # grovepi.chainableRgbLed_test(pin, numLeds, te
20 # grovepi.chainableRgbLed_pattern(pin, pattern,
21 # grovepi.chainableRgbLed_modulo(pin, offset, d
22 # grovepi.chainableRgbLed_setLevel(pin, level,
23
24 # test colors used in grovepi.chainableRgbLed_t
25 testColorBlack = 0 # 0b000 #000000
26 testColorBlue = 1 # 0b001 #0000FF

```

```
27     testColorGreen = 2    # 0b010 #00FF00
28     testColorCyan  = 3    # 0b011 #00FFFF
29     testColorRed   = 4    # 0b100 #FF0000
30     testColorMagenta = 5 # 0b101 #FF00FF
31     testColorYellow = 6   # 0b110 #FFFF00
32     testColorWhite = 7   # 0b111 #FFFFFF
33
34     # patterns used in grovepi.chainableRgbLed_patt
35     thisLedOnly = 0
36     allLedsExceptThis = 1
37     thisLedAndInwards = 2
38     thisLedAndOutwards = 3
39
40     try:
41
42         print "Test 1) Initialise"
43
44         # init chain of leds
45         grovepi.chainableRgbLed_init(pin, numleds)
46         time.sleep(.5)
47
48         # change color to green
49         grovepi.storeColor(0,255,0)
50         time.sleep(.5)
51
52         # set led 1 to green
53         grovepi.chainableRgbLed_pattern(pin, thisLe
54         time.sleep(.5)
55
56         # change color to red
57         grovepi.storeColor(255,0,0)
58         time.sleep(.5)
59
60         # set led 10 to red
61         grovepi.chainableRgbLed_pattern(pin, thisLe
62         time.sleep(.5)
63
64         # pause so you can see what happened
65         time.sleep(2)
66
67         # reset (all off)
```

```
68     grovepi.chainableRgbLed_test(pin, numleds,
69     time.sleep(.5)
70
71
72     print "Test 2a) Test Patterns - black"
73
74     # test pattern 0 - black (all off)
75     grovepi.chainableRgbLed_test(pin, numleds,
76     time.sleep(1)
77
78
79     print "Test 2b) Test Patterns - blue"
80
81     # test pattern 1 blue
82     grovepi.chainableRgbLed_test(pin, numleds,
83     time.sleep(1)
84
85
86     print "Test 2c) Test Patterns - green"
87
88     # test pattern 2 green
89     grovepi.chainableRgbLed_test(pin, numleds,
90     time.sleep(1)
91
92
93     print "Test 2d) Test Patterns - cyan"
94
95     # test pattern 3 cyan
96     grovepi.chainableRgbLed_test(pin, numleds,
97     time.sleep(1)
98
99
100    print "Test 2e) Test Patterns - red"
101
102    # test pattern 4 red
103    grovepi.chainableRgbLed_test(pin, numleds,
104    time.sleep(1)
105
106
107    print "Test 2f) Test Patterns - magenta"
108
```

```
109      # test pattern 5 magenta
110      grovepi.chainableRgbLed_test(pin, numleds,
111      time.sleep(1)
112
113
114      print "Test 2g) Test Patterns - yellow"
115
116      # test pattern 6 yellow
117      grovepi.chainableRgbLed_test(pin, numleds,
118      time.sleep(1)
119
120
121      print "Test 2h) Test Patterns - white"
122
123      # test pattern 7 white
124      grovepi.chainableRgbLed_test(pin, numleds,
125      time.sleep(1)
126
127
128      # pause so you can see what happened
129      time.sleep(2)
130
131      # reset (all off)
132      grovepi.chainableRgbLed_test(pin, numleds,
133      time.sleep(.5)
134
135
136      print "Test 3a) Set using pattern - this le
137
138      # change color to red
139      grovepi.storeColor(255,0,0)
140      time.sleep(.5)
141
142      # set led 3 to red
143      grovepi.chainableRgbLed_pattern(pin, thisLe
144      time.sleep(.5)
145
146      # pause so you can see what happened
147      time.sleep(2)
148
149      # reset (all off)
```

```
150     grovepi.chainableRgbLed_test(pin, numleds,
151     time.sleep(.5)
152
153
154     print "Test 3b) Set using pattern - all led
155
156     # change color to blue
157     grovepi.storeColor(0,0,255)
158     time.sleep(.5)
159
160     # set all leds except for 3 to blue
161     grovepi.chainableRgbLed_pattern(pin, allLed
162     time.sleep(.5)
163
164     # pause so you can see what happened
165     time.sleep(2)
166
167     # reset (all off)
168     grovepi.chainableRgbLed_test(pin, numleds,
169     time.sleep(.5)
170
171
172     print "Test 3c) Set using pattern - this le
173
174     # change color to green
175     grovepi.storeColor(0,255,0)
176     time.sleep(.5)
177
178     # set leds 1-3 to green
179     grovepi.chainableRgbLed_pattern(pin, thisLe
180     time.sleep(.5)
181
182     # pause so you can see what happened
183     time.sleep(2)
184
185     # reset (all off)
186     grovepi.chainableRgbLed_test(pin, numleds,
187     time.sleep(.5)
188
189
190     print "Test 3d) Set using pattern - this le
```

```
191
192     # change color to green
193     grovepi.storeColor(0,255,0)
194     time.sleep(.5)
195
196     # set leds 7-10 to green
197     grovepi.chainableRgbLed_pattern(pin, thisLe
198     time.sleep(.5)
199
200     # pause so you can see what happened
201     time.sleep(2)
202
203     # reset (all off)
204     grovepi.chainableRgbLed_test(pin, numleds,
205     time.sleep(.5)
206
207
208     print "Test 4a) Set using modulo - all leds
209
210     # change color to black (fully off)
211     grovepi.storeColor(0,0,0)
212     time.sleep(.5)
213
214     # set all leds black
215     # offset 0 means start at first led
216     # divisor 1 means every led
217     grovepi.chainableRgbLed_modulo(pin, 0, 1)
218     time.sleep(.5)
219
220     # change color to white (fully on)
221     grovepi.storeColor(255,255,255)
222     time.sleep(.5)
223
224     # set all leds white
225     grovepi.chainableRgbLed_modulo(pin, 0, 1)
226     time.sleep(.5)
227
228     # pause so you can see what happened
229     time.sleep(2)
230
231     # reset (all off)
```

```
232     grovepi.chainableRgbLed_test(pin, numleds,
233     time.sleep(.5)
234
235
236     print "Test 4b) Set using modulo - every 2"
237
238     # change color to red
239     grovepi.storeColor(255,0,0)
240     time.sleep(.5)
241
242     # set every 2nd led to red
243     grovepi.chainableRgbLed_modulo(pin, 0, 2)
244     time.sleep(.5)
245
246     # pause so you can see what happened
247     time.sleep(2)
248
249
250     print "Test 4c) Set using modulo - every 2,
251
252     # change color to green
253     grovepi.storeColor(0,255,0)
254     time.sleep(.5)
255
256     # set every 2nd led to green, offset 1
257     grovepi.chainableRgbLed_modulo(pin, 1, 2)
258     time.sleep(.5)
259
260     # pause so you can see what happened
261     time.sleep(2)
262
263     # reset (all off)
264     grovepi.chainableRgbLed_test(pin, numleds,
265     time.sleep(.5)
266
267
268     print "Test 4d) Set using modulo - every 3,
269
270     # change color to red
271     grovepi.storeColor(255,0,0)
272     time.sleep(.5)
```

```
273
274     # set every 3rd led to red
275     grovepi.chainableRgbLed_modulo(pin, 0, 3)
276     time.sleep(.5)
277
278     # change color to green
279     grovepi.storeColor(0,255,0)
280     time.sleep(.5)
281
282     # set every 3rd led to green, offset 1
283     grovepi.chainableRgbLed_modulo(pin, 1, 3)
284     time.sleep(.5)
285
286     # change color to blue
287     grovepi.storeColor(0,0,255)
288     time.sleep(.5)
289
290     # set every 3rd led to blue, offset 2
291     grovepi.chainableRgbLed_modulo(pin, 2, 3)
292     time.sleep(.5)
293
294     # pause so you can see what happened
295     time.sleep(2)
296
297     # reset (all off)
298     grovepi.chainableRgbLed_test(pin, numleds,
299     time.sleep(.5)
300
301
302     print "Test 4e) Set using modulo - every 3,
303
304     # change color to yellow
305     grovepi.storeColor(255,255,0)
306     time.sleep(.5)
307
308     # set every 4nd led to yellow
309     grovepi.chainableRgbLed_modulo(pin, 1, 3)
310     time.sleep(.5)
311
312     # pause so you can see what happened
313     time.sleep(2)
```



```
314
315
316     print "Test 4f) Set using modulo - every 3,"
317
318     # change color to magenta
319     grovepi.storeColor(255,0,255)
320     time.sleep(.5)
321
322     # set every 4nd led to magenta
323     grovepi.chainableRgbLed_modulo(pin, 2, 3)
324     time.sleep(.5)
325
326     # pause so you can see what happened
327     time.sleep(2)
328
329     # reset (all off)
330     grovepi.chainableRgbLed_test(pin, numleds,
331     time.sleep(.5)
332
333
334     print "Test 5a) Set level 6"
335
336     # change color to green
337     grovepi.storeColor(0,255,0)
338     time.sleep(.5)
339
340     # set leds 1-6 to green
341     grovepi.write_i2c_block(0x04,[95,pin,6,0])
342     time.sleep(.5)
343
344     # pause so you can see what happened
345     time.sleep(2)
346
347     # reset (all off)
348     grovepi.chainableRgbLed_test(pin, numleds,
349     time.sleep(.5)
350
351
352     print "Test 5b) Set level 7 - reverse"
353
354     # change color to red
```

```

355     grovepi.storeColor(255,0,0)
356     time.sleep(.5)
357
358     # set leds 4-10 to red
359     grovepi.write_i2c_block(0x04,[95,pin,7,1])
360     time.sleep(.5)
361
362
363     except KeyboardInterrupt:
364         # reset (all off)
365         grovepi.chainableRgbLed_test(pin, numleds,
366         break
367     except IOError:
368         print "Error"

```

- Notice that there's something you have to concern of:

```

1     pin = 7           #setting up the output pin
2     numleds = 1      #how many leds you plug

```

- Also all methods you can see in grovepi.py is:

```

1     storeColor(red, green, blue)
2     chainableRgbLed_init(pin, numLeds)
3     chainableRgbLed_test(pin, numLeds, testColor)
4     chainableRgbLed_pattern(pin, pattern, whichLed)
5     chainableRgbLed_modulo(pin, offset, divisor)
6     chainableRgbLed_setLevel(pin, level, reverse)

```

5.Run the demo.

```

sudo python3 grove_chainable_rgb_led.py

```

6. This demo may not work if your grovepi doesn't have the newest firmware, update the firmware.

```
1 cd yourpath/GrovePi/Firmware
2 sudo ./firmware_update.sh
```

With Beaglebone Green

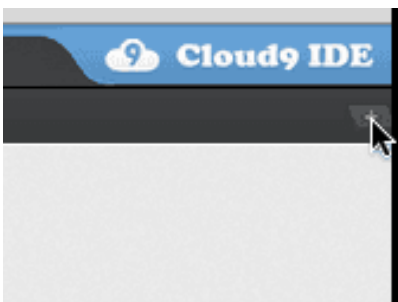
To begin editing programs that live on BBG, you can use the Cloud9 IDE.

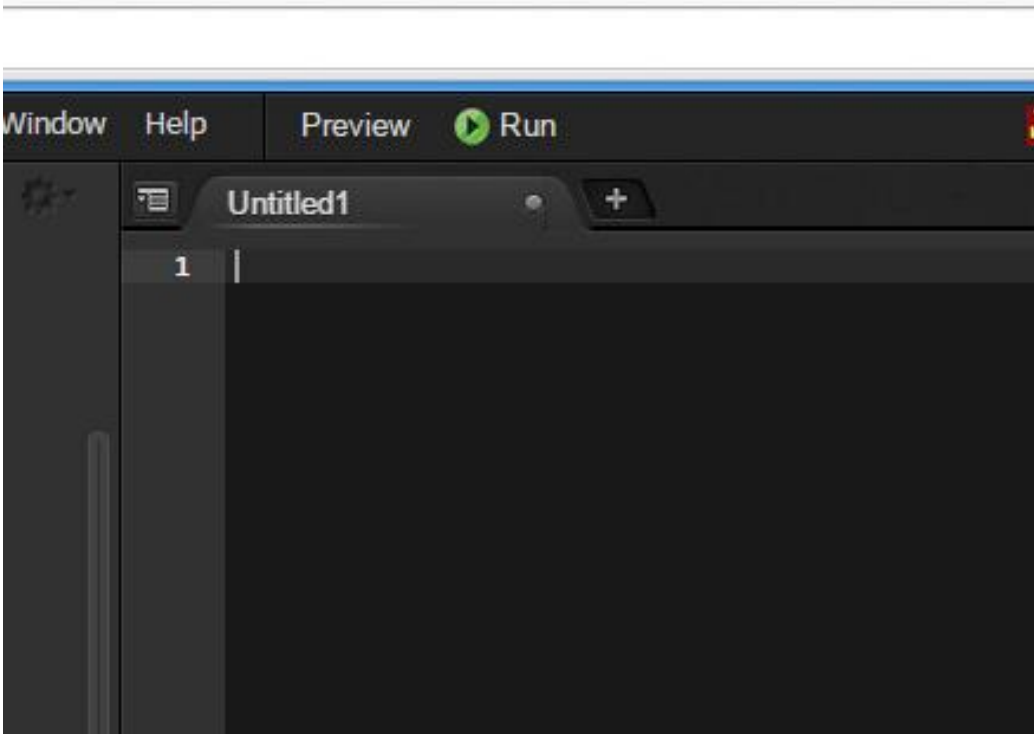
As a simple exercise to become familiar with Cloud9 IDE, creating a simple application to blink one of the 4 user programmable LEDs on the BeagleBone is a good start.

If this is your first time to use Cloud9 IDE, please follow this [link](#) [/BeagleBone_Green].

Step1: Set the Grove - UART socket as a Grove - GPIO Socket, just follow this [link](https://www.seeedstudio.com/recipe/362-how-to-use-the-grove-uart-port-as-a-gpio-on-bbg.html) [https://www.seeedstudio.com/recipe/362-how-to-use-the-grove-uart-port-as-a-gpio-on-bbg.html].

Step2: Click the "+" in the top-right to create a new file.





Step3: Copy and paste the following code into the new tab

```
1  import time
2  import Adafruit_BBIO.GPIO as GPIO
3
4  CLK_PIN = "P9_22"
5  DATA_PIN = "P9_21"
6  NUMBER_OF_LEDS = 1
7
8  class ChainableLED():
9      def __init__(self, clk_pin, data_pin, number_of_led
10         self.__clk_pin = clk_pin
11         self.__data_pin = data_pin
12         self.__number_of_leds = number_of_leds
13
14         GPIO.setup(self.__clk_pin, GPIO.OUT)
15         GPIO.setup(self.__data_pin, GPIO.OUT)
16
17         for i in range(self.__number_of_leds):
18             self.setColorRGB(i, 0, 0, 0)
19
20     def clk(self):
```

```
21     GPIO.output(self.__clk_pin, GPIO.LOW)
22     time.sleep(0.00002)
23     GPIO.output(self.__clk_pin, GPIO.HIGH)
24     time.sleep(0.00002)
25
26     def sendByte(self, b):
27         "Send one bit at a time, starting with the MSB"
28         for i in range(8):
29             # If MSB is 1, write one and clock it, else
30             if (b & 0x80) != 0:
31                 GPIO.output(self.__data_pin, GPIO.HIGH)
32             else:
33                 GPIO.output(self.__data_pin, GPIO.LOW)
34             self.clk()
35
36             # Advance to the next bit to send
37             b = b << 1
38
39     def sendColor(self, red, green, blue):
40         "Start by sending a byte with the format '1 1 /
41         #prefix = B11000000
42         prefix = 0xC0
43         if (blue & 0x80) == 0:
44             #prefix |= B00100000
45             prefix |= 0x20
46         if (blue & 0x40) == 0:
47             #prefix |= B00010000
48             prefix |= 0x10
49         if (green & 0x80) == 0:
50             #prefix |= B00001000
51             prefix |= 0x08
52         if (green & 0x40) == 0:
53             #prefix |= B00000100
54             prefix |= 0x04
55         if (red & 0x80) == 0:
56             #prefix |= B00000010
57             prefix |= 0x02
58         if (red & 0x40) == 0:
59             #prefix |= B00000001
60             prefix |= 0x01
61         self.sendByte(prefix)
```

```
62
63     # Now must send the 3 colors
64     self.sendByte(blue)
65     self.sendByte(green)
66     self.sendByte(red)
67
68     def setColorRGB(self, led, red, green, blue):
69         # Send data frame prefix (32x '0')
70         self.sendByte(0x00)
71         self.sendByte(0x00)
72         self.sendByte(0x00)
73         self.sendByte(0x00)
74
75         # Send color data for each one of the leds
76         for i in range(self.__number_of_leds):
77             '''
78             if i == led:
79                 _led_state[i*3 + _CL_RED] = red;
80                 _led_state[i*3 + _CL_GREEN] = green;
81                 _led_state[i*3 + _CL_BLUE] = blue;
82                 sendColor(_led_state[i*3 + _CL_RED],
83                           _led_state[i*3 + _CL_GREEN],
84                           _led_state[i*3 + _CL_BLUE]);
85             '''
86             self.sendColor(red, green, blue)
87
88         # Terminate data frame (32x "0")
89         self.sendByte(0x00)
90         self.sendByte(0x00)
91         self.sendByte(0x00)
92         self.sendByte(0x00)
93
94
95     # Note: Use P9_22(UART2_RXD) and P9_21(UART2_TXD) as GP.
96     # Connect the Grove - Chainable RGB LED to UART Grove p
97     if __name__ == "__main__":
98         rgb_led = ChainableLED(CLK_PIN, DATA_PIN, NUMBER_OF_
99
100         while True:
101             # The first parameter: NUMBER_OF_LEDS - 1; Othe
102             rgb_led.setColorRGB(0, 255, 0, 0)
```

```
103     time.sleep(2)
104     rgb_led.setColorRGB(0, 0, 255, 0)
105     time.sleep(2)
106     rgb_led.setColorRGB(0, 0, 0, 255)
107     time.sleep(2)
108     rgb_led.setColorRGB(0, 0, 255, 255)
109     time.sleep(2)
110     rgb_led.setColorRGB(0, 255, 0, 255)
111     time.sleep(2)
112     rgb_led.setColorRGB(0, 255, 255, 0)
113     time.sleep(2)
114     rgb_led.setColorRGB(0, 255, 255, 255)
115     time.sleep(2)
```

Step4: Save the file by clicking the disk icon and giving the file a name with the .py extension.

Step5: Connect Grove Chainable RGB LED to Grove UART socket on BBG.

Step6: Run the code. You'll find the RGB LED is changing color every 2 seconds.

Chainable RGB LED eagle file V1



Chainable RGB LED eagle file V2



Resources

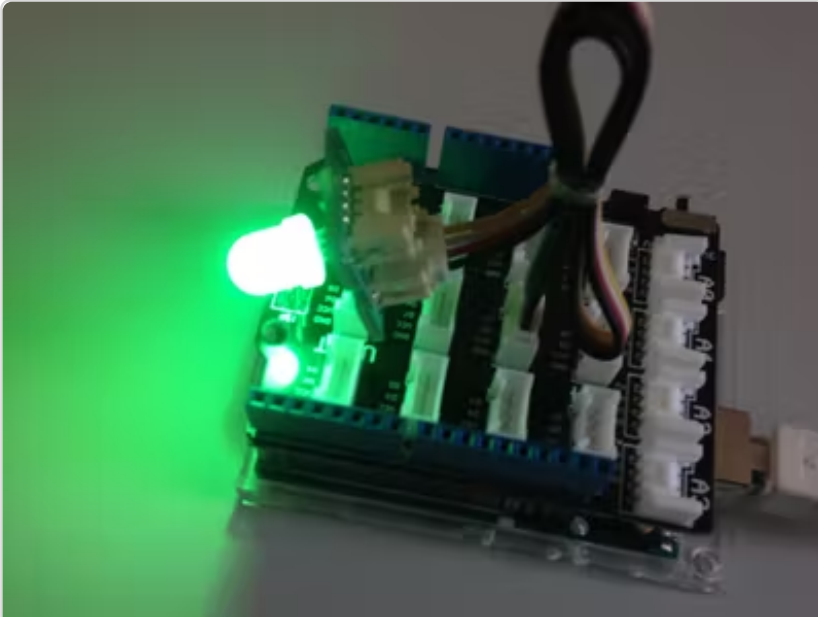
- **[Library]** [Chainable RGB LED Library for the P9813](https://github.com/pjpmarques/ChainableLED)
[https://github.com/pjpmarques/ChainableLED]
- **[Library]** [Github repository for Chainable RGB LED Library \(new\)](https://github.com/Seeed-Studio/Grove_Chainable_RGB_LED)
[https://github.com/Seeed-Studio/Grove_Chainable_RGB_LED]
- **[Library]** [CodeCraft Code](https://files.seeedstudio.com/wiki/Grove-)
[https://files.seeedstudio.com/wiki/Grove-

[Chainable_RGB_LED/res/Chainable%20RGB%20LED.zip](#)]

- **[Eagle]**[Chainable RGB LED eagle file V1](#)
[https://files.seeedstudio.com/wiki/Grove-Chainable_RGB_LED/res/Chainable_RGB_LED_eagle_file%20V1.zip]
- **[Eagle]**[Chainable RGB LED eagle file V2](#)
[https://files.seeedstudio.com/wiki/Grove-Chainable_RGB_LED/res/Grove%20-%20Chainable%20RGB%20LED%20v2.0.zip]
- **[Datasheet]**[P9813 Datasheet](#)
[https://files.seeedstudio.com/wiki/Grove-Chainable_RGB_LED/res/P9813_datasheet.pdf]

Projects

Grove - Introduction to Chainable LED: This project shows how to connect a chainable LED to Grove.



(<https://www.hackster.io/ingo-lohs/grove-introduction-to-chainable-led-d668b7>)

DIY a device for explaining RGB color model



(<https://www.hackster.io/kevin-lee2/diy-a-device-for-explaining-rgb-color-model-496cbc>)

Security Access Using Seeduino Lotus When you knock on the door or close to the door, the door will open automatically.



(<https://www.hackster.io/limanchen/security-access-using-seeduino-lotus-7eb90f>)

Security Access Using Seeduino Lotus

(<https://www.hackster.io/limanchen/security>)

Tech Support

Please submit any technical issue into our [forum](https://forum.seeedstudio.com/)
[<https://forum.seeedstudio.com/>].



[<https://www.seeedstudio.com/act-4.html?>

utm_source=wiki&utm_medium=wikibanner&utm_campaign=newpr
oducts]