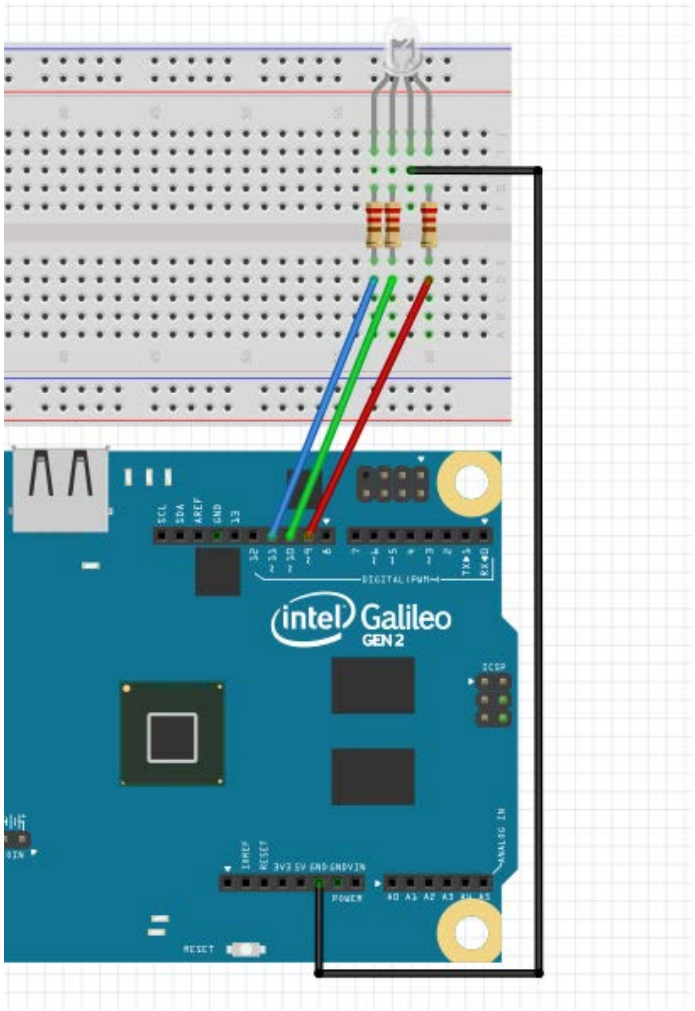
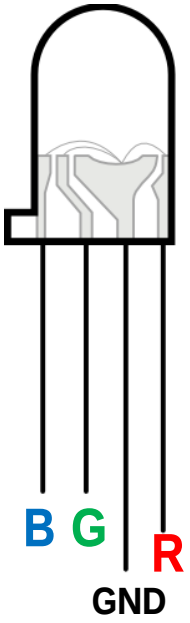
Cheerlights – Part 3

Integrating Python and Arduino to make your LED respond

Setting Up the Circuit

- Tri-Color LED
- GND pin to Ground
- Resistors between each color pin and the Galileo socket
- In this sketch I have:
  - Red – pin 9
  - Green – pin 10
  - Blue – pin 11





Connecting to Linux from Arduino

- We want to use our Python script to get the color from the internet, and then use our Arduino code to change the LED color accordingly
- First, we need to change the Python output so that Arduino can read it
- Right now, it prints a color as a string, infinitely. Arduino needs RGB tuples in order to set the pin voltages, though

The New Python Script

- We can set up a dictionary at the beginning of the Python code to match each color with its RGB value:

```
colors = {'red': '100,0,0',
          'green': '0,100,0',
          'blue': '0,0,100',
          'cyan': '0,100,100',
          'white': '100,100,100',
          'warmwhite': '99,96,90',
          'purple': '50,0,50',
          'magenta': '100,0,100',
          'yellow': '100,100,0',
          'orange': '100,65,0',
          'pink': '100,75,80',
          'oldlace': '99,96,90'}
```

The New Python Script

- Next, we convert out **last_color** variable from its string to its corresponding set of RGB values
- We also get rid of the while loop (we only need the value once) and the extra words it prints:

```
last_color = urllib2.urlopen(url).read()
print colors[last_color]
```

- The full new code is available in the Github as *cheerlights_receiver.py*
- Transfer the new Python script to the Galileo via **pscp**

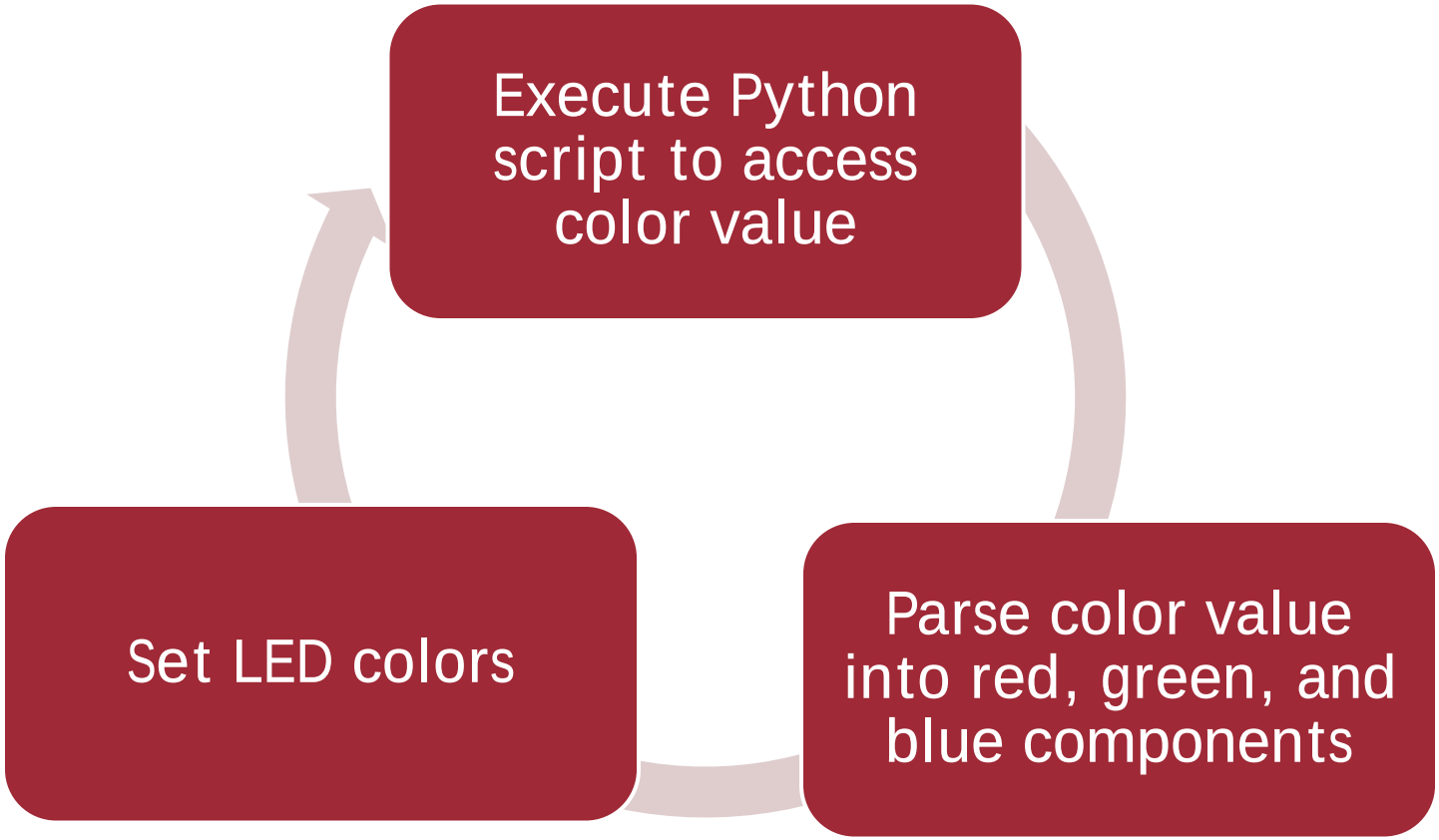
Integrating with Arduino

- The Arduino needs to be able to call the Python script, so we'll put it in the top-level of the SD card so we can use Arduino's SD library
- This directory is located at `/media/realroot`
- Move the Python script there by typing

  `mv cheerlights_receiver.py /media/realroot`

  into the Linux shell

The Arduino Code: Overview



Running the Python Code from Arduino

- View the code in the file *tricolor_led_cheerlights.ino*
- We can't directly access what Python prints to command line, but we can save what it prints into a file and access the file contents
- The command **system("…")** in Arduino executes the given command in Linux

```
system("python /media/realroot/cheerlights_receiver.py > /media/realroot/colors");
```

- This tells Linux to execute our script and the "**>**" tells it to write the output of that command into a file we've named "colors"

Accessing the colors File

- Using Arduino's SD library, we can access the file we saved to the card
- We imported this library using the `#include <SD.h>` line at the top of the program
- We then use the SD.open() function to save it as a File variable

```
File colors_file = SD.open("colors");
```

Reading the File variable

```
while (colors_file.peek() != '\n') {
  r = colors_file.parseInt();
  g = colors_file.parseInt();
  b = colors_file.parseInt();
}
```

- The `peek()` function checks what the next character in the file is. This loop will read characters until we hit a newline, which means we've read all the contents of that line
- The `parseInt()` function is a way to access the next integer while skipping the commas in the file

Finishing the Function

```
colors_file.close(); // close the colors file
system("rm /media/realroot/colors"); // remove
```

- Now that we have captured the information we need for the current color, we close the file and remove it from the system so that we can create a new one during the next read

Using the variables (loop function)

```
if (millis() > prev_millis + update_rate)
  {
    prev_millis = millis();
    update_colors();
    analogWrite(RED_PIN, r);
    analogWrite(BLUE_PIN, b);
    analogWrite(GREEN_PIN, g);
  }
```

- This code calls the `update_colors()` function to read the new color variables and changes the pin values accordingly
- It does this every ten seconds according to the `update_rate` variable

Seeing the Output

- Upload the code to watch your LED change in real time with according to the current cheerlights color
- If you have a twitter, you can tweet @cheerlights with a new color to see everyone's LED change

Allowed colors: red, green, blue, cyan, white, warmwhite, purple, magenta, yellow, orange, pink, oldlace