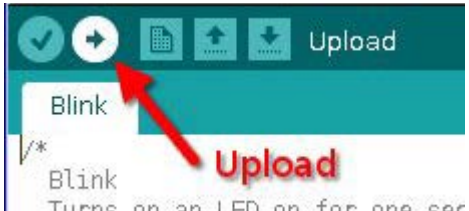
Cheerlights – Part 1

Using a Tri-Color LED in Arduino

Running an Arduino Sketch

- Open your Arduino environment by opening the **arduino.exe** file that you saved during setup.
- Upload the **blink** sketch to your Galileo by navigating to **File > Examples > 01.Basics > Blink** to load the sketch file
- Make sure that the **Board** in the Tools menu is Galileo Gen 2, and the **Serial Port** matches the one indicated by your Device Manager
- Click the **Upload** button



What's in the Blink Sketch

- The **setup()** function runs once at the start
  - Here, it sets **pinMode** for the LED to OUTPUT. It's necessary to set pinMode for all digital pins used
- The **loop()** function that runs over and over
  - It uses **digitalWrite** to write either HIGH (on) or LOW (off) voltage to the LED pin, waiting 1000 milliseconds in between each command

Common Arduino functions

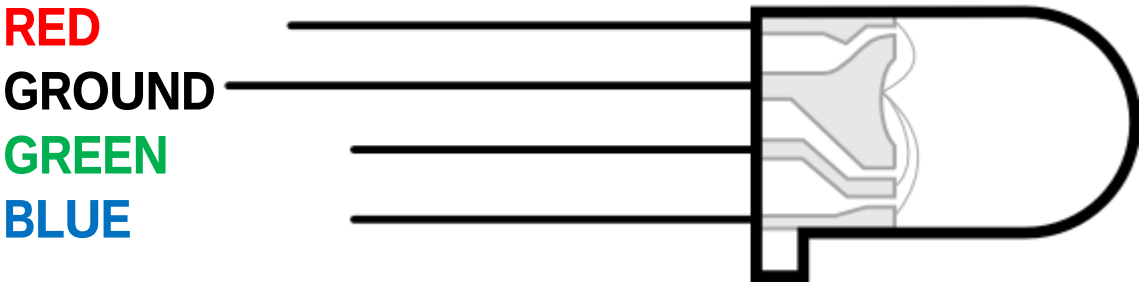
- **`digitalRead()`** will read input from a digital pin (either 1 or 0)
- **`digitalWrite()`** will write a digital value to a digital pin (either 1 or 0, HIGH or LOW)
- **`analogWrite()`** will write an analog value to a pin (given as an integer between 0 and 255)
- **`analogRead()`** will read an analog voltage value from an analog pin and present it as an integer from 0 to 1023

Other Notes about Arduino Code

- In C language, all variables have to be declared by type (int, float, double, long, String, etc)
  - `int i = 0;`
- Similarly, functions must be declared by the type of their return value
  - `void setup()`
  - `int get_value()`
- All lines must end in a semicolon (`;`)
- Functions, loops, and if statements are *blocked* and denoted by curly braces: `{ }`

Your Tri-Color LED

- Four pins:



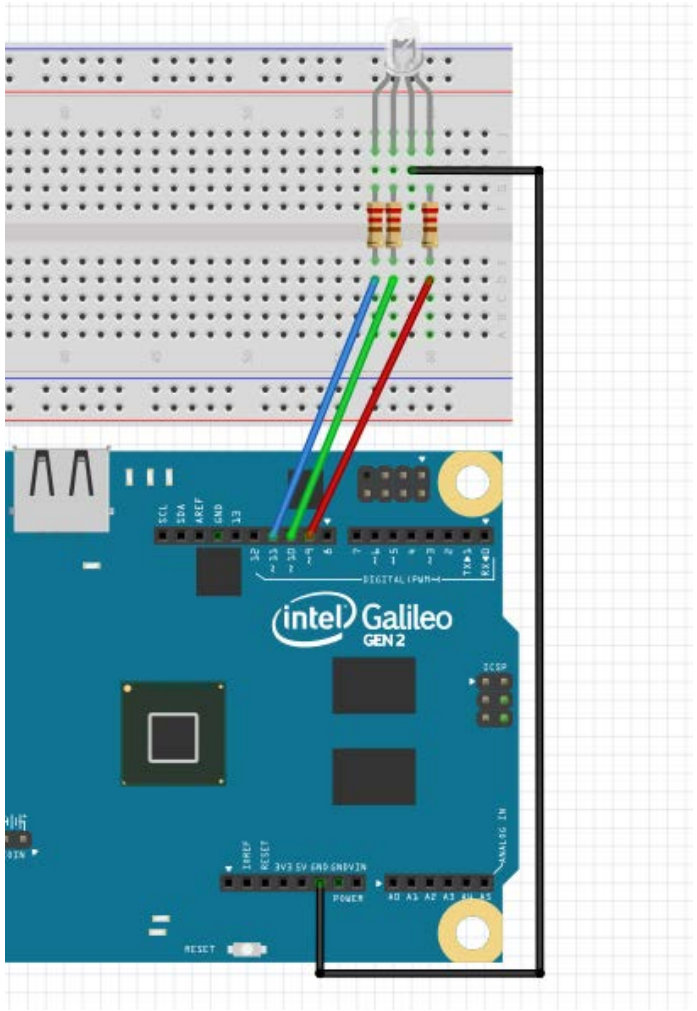
- Each color pin will be connected to its own pin on the Galileo to be written individually

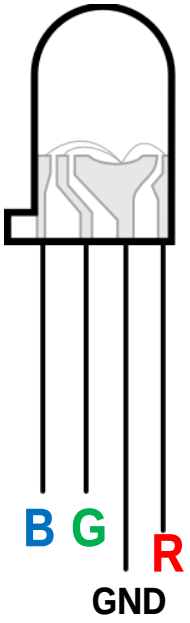
Understanding the Tri-Color LED

- It has 3 LEDs in it (red, green, and blue) that can be given values from 0-255 to create colors
- Colors are expressed in RGB as a *tuple*, or list of three numbers (R,G,B)
- Red is (255,0,0), green is (0,255,0), blue is (0,0,255), and white is (255,255,255)
- Look online for the RGB codes for other common colors

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





Arduino Code

- Open the *tricolor_led.ino* sketch from your Arduino folder (remember that you copied all the Arduino files into your Documents/Arduino folder)
- It will cycle through some of the basic colors

Code Parts: Define Pins

```
int RED_PIN = 9;
int GREEN_PIN = 10;
int BLUE_PIN = 11;
```

- This piece of code defines variables hold the pin numbers for each color pin

Code Parts: Setup Function

```
void setup()
{
  pinMode(RED_PIN, OUTPUT);
  pinMode(GREEN_PIN, OUTPUT);
  pinMode(BLUE_PIN, OUTPUT);
}
```

- This piece of code sets each pin as an OUTPUT pin, meaning we will tell the Galileo how much voltage to send to each pin

Code Parts: Loop

```
void loop()
{
  set_color(255, 0, 0);  // red
  delay(1000);
  set_color(0, 255, 0);  // green
  delay(1000);
...
```

- The loop function sets each color for one second apiece by called a function we've created called **set_color**

Code Parts: set_color function

```
void set_color(int red, int green, int blue)
{
  analogWrite(RED_PIN, red);
  analogWrite(GREEN_PIN, green);
  analogWrite(BLUE_PIN, blue);
}
```

- This function is **void** (it does not return anything)
- It takes the RGB color values are *arguments* and uses the **`analogWrite`** function to send that voltage value to each color pin.

Extending the Tri-Color Code

- Change the **loop** function so that the LED fades smoothly from red to green to blue

Hint: Use a **for loop** to increment one pin value as you decrease the other. The syntax for a **for** loop in C is:

```
for(i=0; i<NUMBER; i++)
{
  // code here
}
```

See the solution in the file *tricolor_led_fader.ino*