Challenge 2 - LED Fade

In this tutorial, you'll extend the previous tutorial to include the ability to 'fade' the LED, i.e. change the brightness. This will demonstrate the difference between digitalWrite and analogWrite and introduce the concept of Pulse Width Modulation (PWM).

Resources Needed

Hardware:

- Arduino Uno (or equivalent),
- LED,
- 220Ω Resistor,
- Jumper Wires

Software:

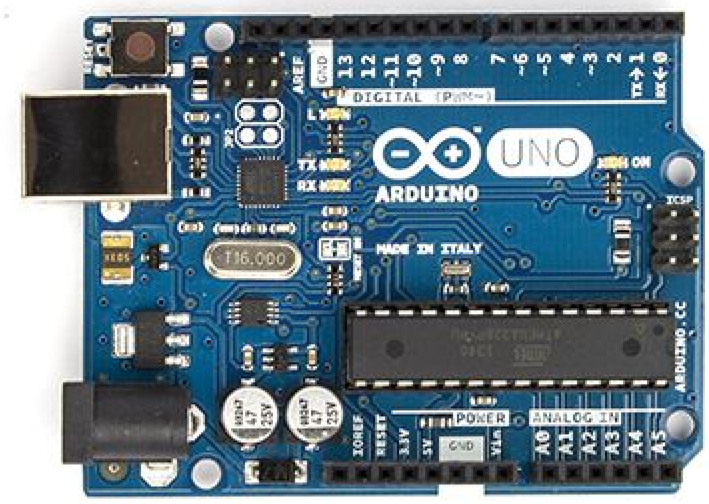
- Arduino IDE
- Fritzing

Theory

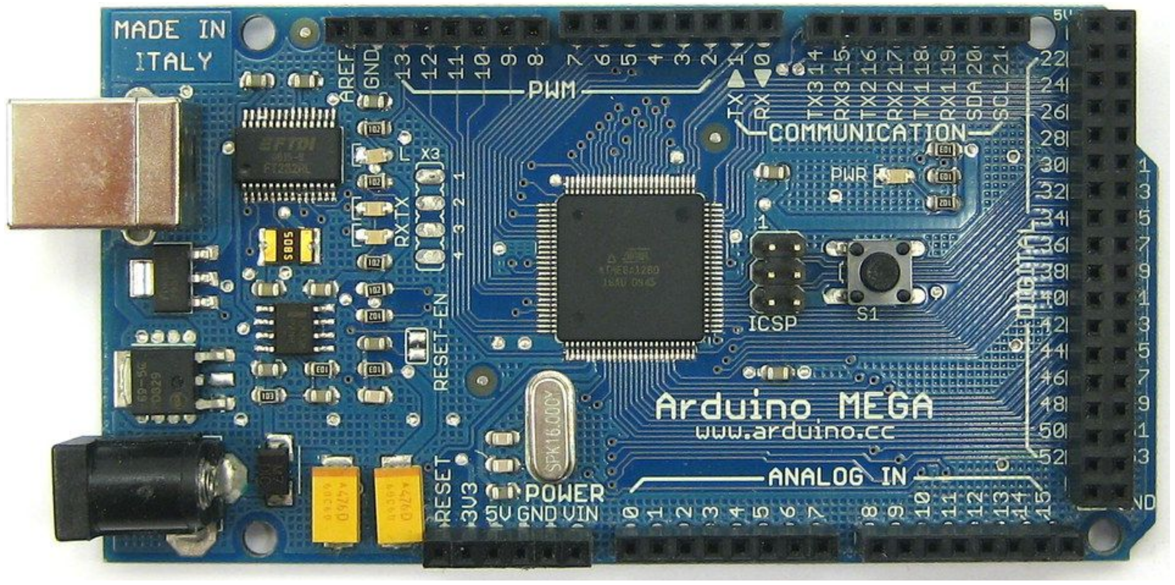
Pulse Width Modulation (PWM)

Pulse Width Modulation is the ability to simulate analog output using digital signals. The output pins on Arduino (and compatible) boards are digital.

The Arduino Uno boards have the PWM pins indicated with a tilde (~) symbol next to the pin.



The Arduino Mega has more PWM pins, but still not all digital out pins are PWM:



You can use this site to see the PWM compatible pins on all Arduino boards:

https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/

This is an excellent resource to learn the concepts of PWM:
https://learn.sparkfun.com/tutorials/pulse-width-modulation

Variables

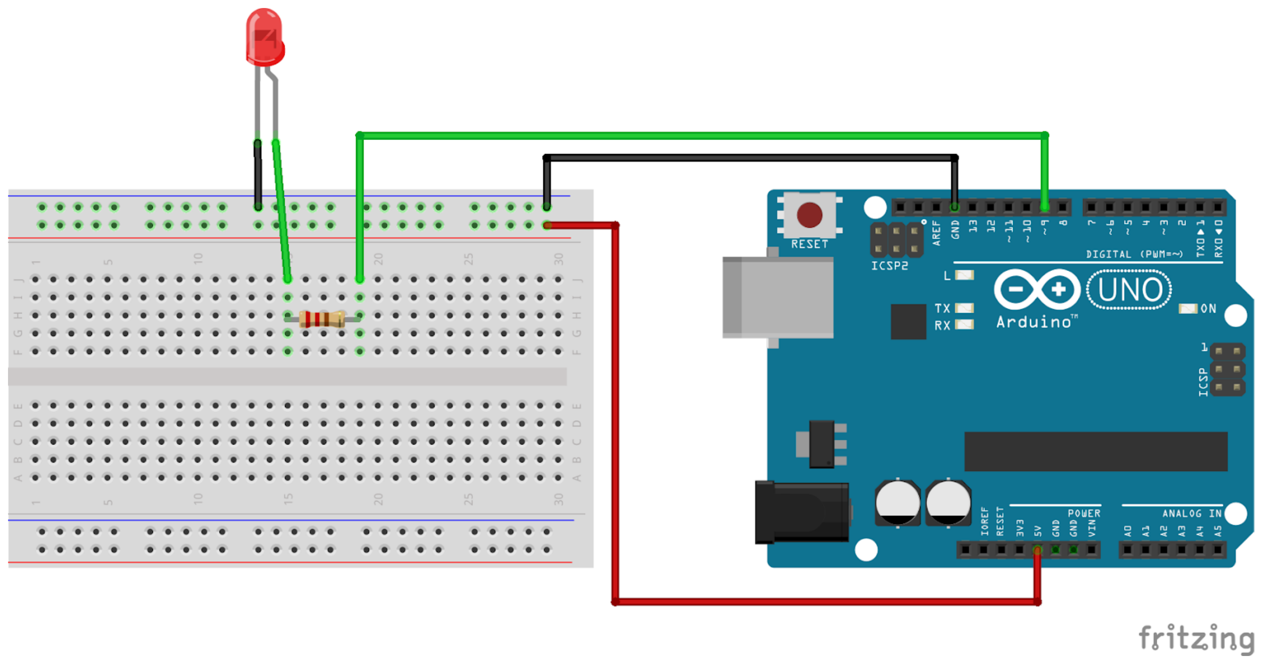
TODO

Version 1

This code and circuit demonstrates the use of PWM when outputting to compatible devices such as LEDs. PWM allows you to output a range of values between LOW and HIGH. Instead of the LED being turned on and off, PWM allows the LED to be powered at 20%, 25%, 73%, etc.

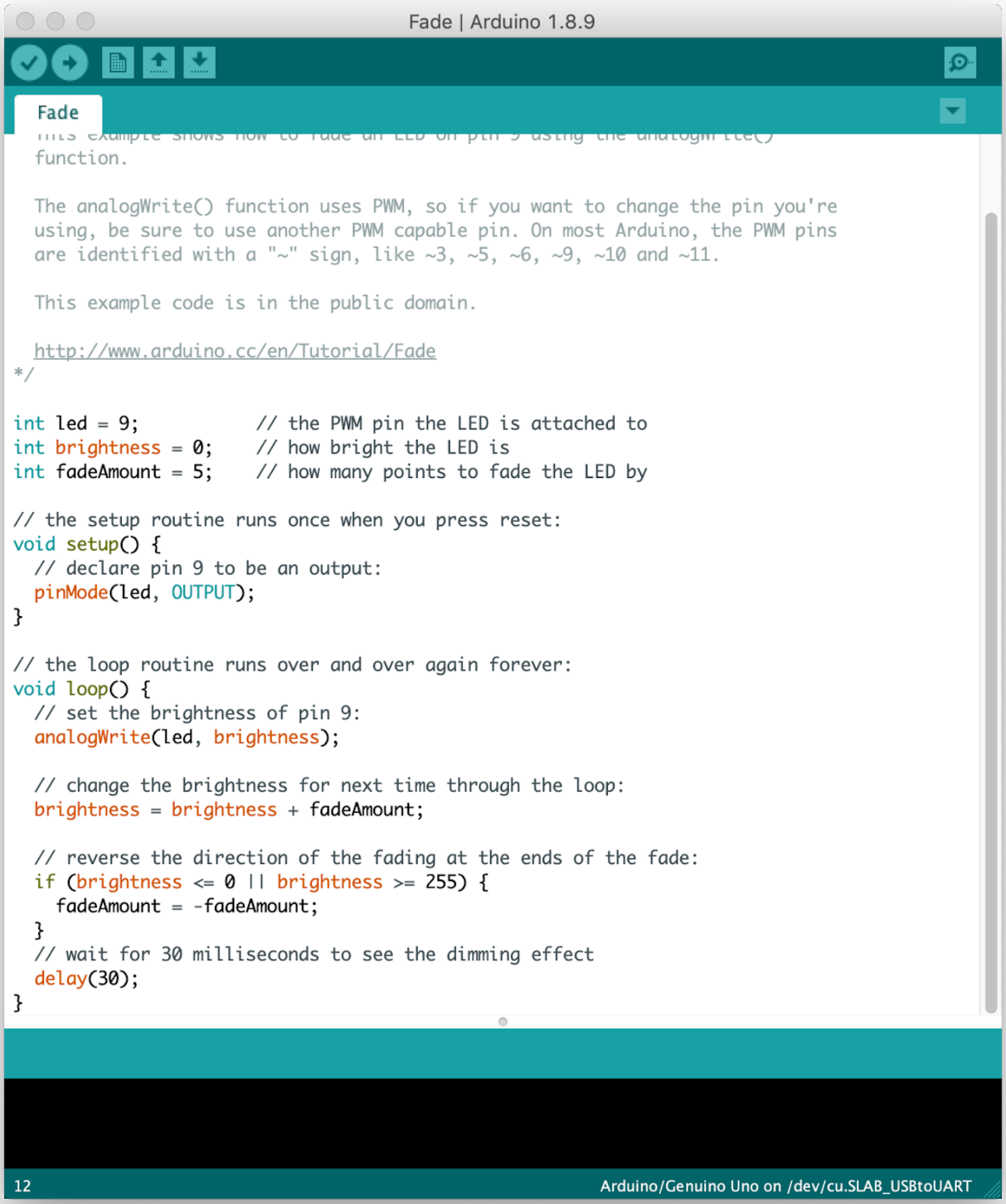
Circuit

The circuit here is the same as in the previous challenge, except the output pin for the LED has moved from Pin 13 to Pin 9.



Code

This **Fade** code example can be found in the Arduino IDE under File -> Examples -> 01.Basics



```
  function.

  The analogWrite() function uses PWM, so if you want to change the pin you're
  using, be sure to use another PWM capable pin. On most Arduino, the PWM pins
  are identified with a "~" sign, like ~3, ~5, ~6, ~9, ~10 and ~11.

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Fade
*/

int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

12   Arduino/Genuino Uno on /dev/cu.SLAB_USBtoUART

```
/*
```

```
  Fade

  This example shows how to fade an LED on pin 9 using the analogWrite()
  function.

  The analogWrite() function uses PWM, so if you want to change the pin you're
  using, be sure to use another PWM capable pin. On most Arduino, the PWM pins
  are identified with a "~" sign, like ~3, ~5, ~6, ~9, ~10 and ~11.

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Fade
*/

int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Code Explanation

As you can see in the code above, the setup function sets the LED pin (currently set to 9) to an output pin. This is a necessary step for arduino code otherwise any digital or analog write command will not function correctly.

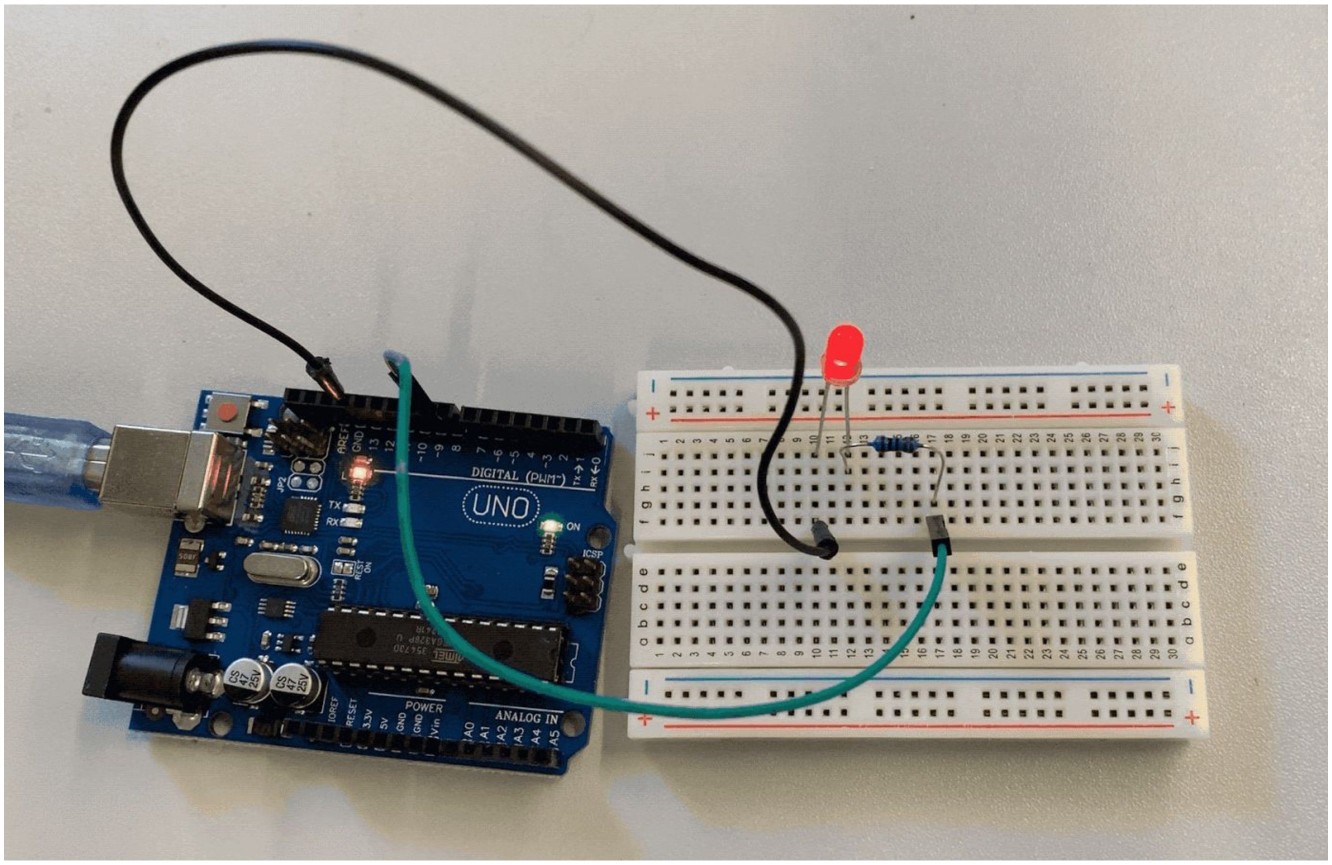
The main loop() function firstly writes the current value of brightness to the led. Initially that is set to 0 (LOW or OFF). The next line of code when adds fadeAmount (set to 5) to brightness, meaning that the next value that will be written to the LED using analogWrite will be 5 instead of 0 in the next iteration of the loop.

The if statement is testing whether brightness has reached a level of being equal or greater than 255 or equal to or less than 0. 255 is the maximum value for PWM and 0 is the lowest. If these outer bounds are reached, then fadeAmount is made negative. This means that if fadeAmount is 5, then it will become -5, and if it is -5, then it will become 5.

So, if brightness is 255, then fadeAmount will be set to -5, meaning the brightness will start decreasing back to 0, and then will increase again once 0 is reached.

Process

Publish the code to the Arduino and watch the LED fade on and off.



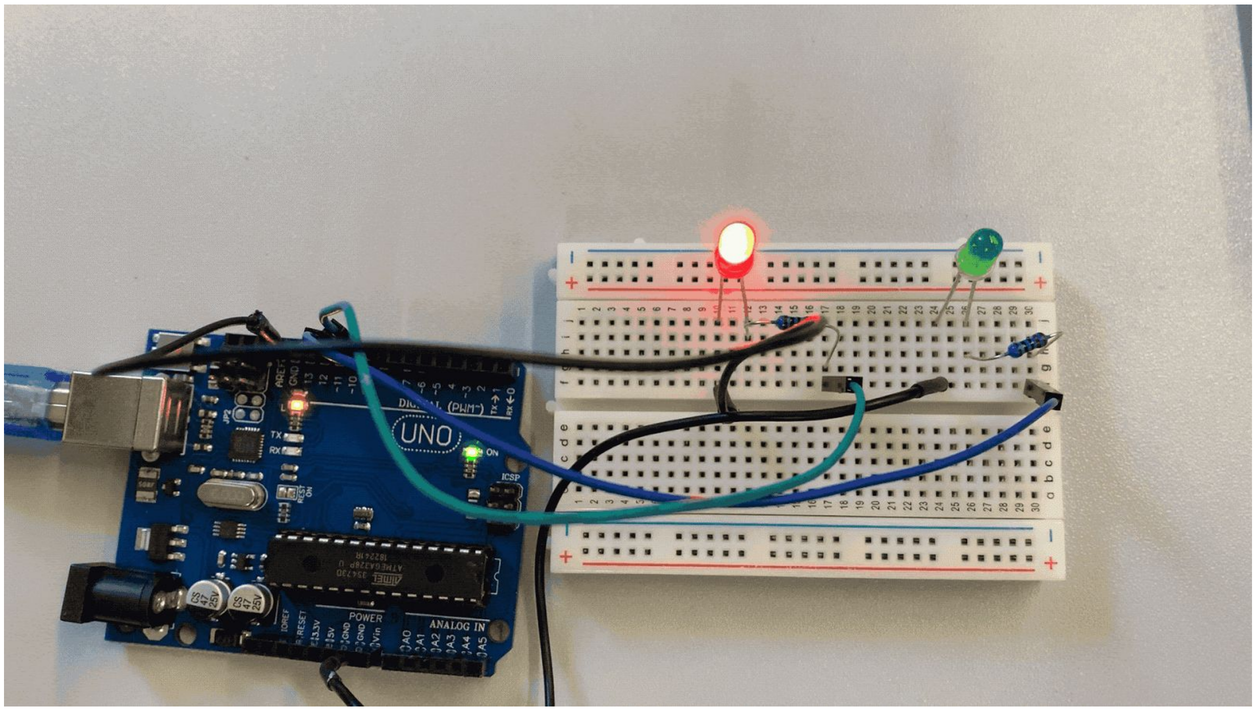
Challenge Extension

Extend your circuit and code to have 2 LEDs fade from 0 to 255 opposite to each other - I.e. when one LED is set to 0, then the other is set to 255. These LEDs will need to be wired to different PWM pins.

Start by wiring the extra LED on another PWM pin (check which pins on your board are compatible), then modify the code to write the same brightness value to both pins to ensure that it's working correctly.

Once you have confirmed that the wiring is correct and the above test works, then expand your code by adding another variable to be used on the second LED. You could call this variable "brightnessTwo" for example.

When brightness is updated by increasing or decreasing it, brightnessTwo will need to be increased or decreased (the opposite to brightness) by fadeAmount as well.



Finally, clean up your code by updating the comments to match the functionality and then ensure that your variables match the accepted Arduino Style Guide.

Update the fritzing diagram to match your final circuit.