Belajar Arduino Dalam 15 Langkah

DISUSUN OLEH:
DESSYANTO BOEDI P, ST., MT.
HIDAYATULAH HIMAWAN, ST., M.ENG.
MANGARAS YANU F, ST., MT.

PENERBIT LPPM UPN "VETERAN" YOGYAKARTA

ISBN 978-602-5534-75-1
9 786025 534751

Belajar Arduino
Dalam 15 Langkah

Disusun Oleh:
Dessyanto Boedi P, ST., MT.
Hidayatulah Himawan, ST., M.Eng.
Mangaras Yanu F, ST., M.Eng.

PENERBIT LPPM UPN "VETERAN" YOGYAKARTA

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang senantiasa memberikan rahmat, hidayah serta ridhonya sehingga penulis dapat menyelesaikan buku yang berjudul “Belajar Arduino Dalam 15 Langkah”. Salawat serta salam diberikan kepada jujungan Nabi Muhammad SAW yang telah membawa risalah yang benar sebagai pedoman menuju jalan yang lurus yang di ridhoi Allah SWT. Buku ini merupakan salah satu instrument output penelitian. Terlepas dari semua itu, penulis menyadari sepenuhnya bahwa masih ada kekurangan baik dari segi susunan kalimat maupun tata bahasanya. Oleh karena itu dengan tangan terbuka penulis menerima segala saran dan kritik dari pembaca agar penulis dapat memperbaiki makalah ilmiah ini. Akhir kata penulis berharap semoga buku ini dapat memberikan manfaat maupun inpirasi terhadap pembaca. Semoga Allah SWT selalu meridhoi semua umatnya, Amiin Ya Robbal Alamin.

Yogyakarta, September 2019

Penulis

DAFTAR ISI

Pendahuluan

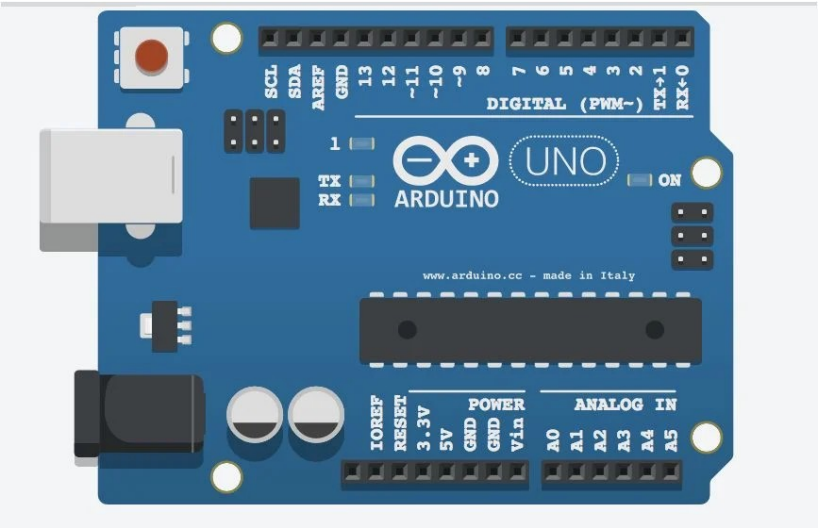
Arduino adalah komputer berukuran saku (juga disebut "mikrokontroler") yang dapat Anda program dan gunakan untuk mengontrol sirkuit. Berinteraksi dengan kata luar melalui sensor, led, motor, speaker ... bahkan internet; ini membuatnya menjadi platform yang fleksibel untuk banyak proyek kreatif. Beberapa kegunaan populer termasuk:

- Layar cahaya yang dapat diprogram yang merespons musik atau interaksi manusia
- robot yang menggunakan informasi dari sensor untuk menavigasi atau melakukan tugas lain
- pengontrol dan antarmuka unik dan dapat disesuaikan untuk musik, game, dan lainnya
- Menghubungkan benda-benda dunia nyata ke internet (twitter sangat populer)
- apapun yang interaktif
- mengotomatisasi dan membuat prototipe

Ada beberapa mikrokontroler di pasaran saat ini, tetapi Arduino terpisah dari yang lain karena komunitas online aktif di sekitarnya. Jika Anda mencari di google atau youtube, Anda akan menemukan banyak ide dan informasi proyek hebat untuk membantu Anda memulai. Meskipun Anda mungkin tidak memiliki pengalaman pemrograman atau bekerja dengan mikrokontroler, Arduino mudah untuk dibangun dan dijalankan, dan ini

adalah cara yang menyenangkan untuk belajar tentang elektronik melalui eksperimen.

Langkah 1: Apa itu Arduino?



Pertama kita akan melihat semua bagian dari Arduino. Arduino pada dasarnya adalah komputer kecil yang dapat terhubung ke sirkuit listrik. The Arduino Uno ditenagai oleh chip Atmega 328P, itu adalah chip terbesar di papan (lihat catatan gambar pada gambar di atas). Chip ini mampu menjalankan program yang disimpan dalam memori (sangat terbatas).

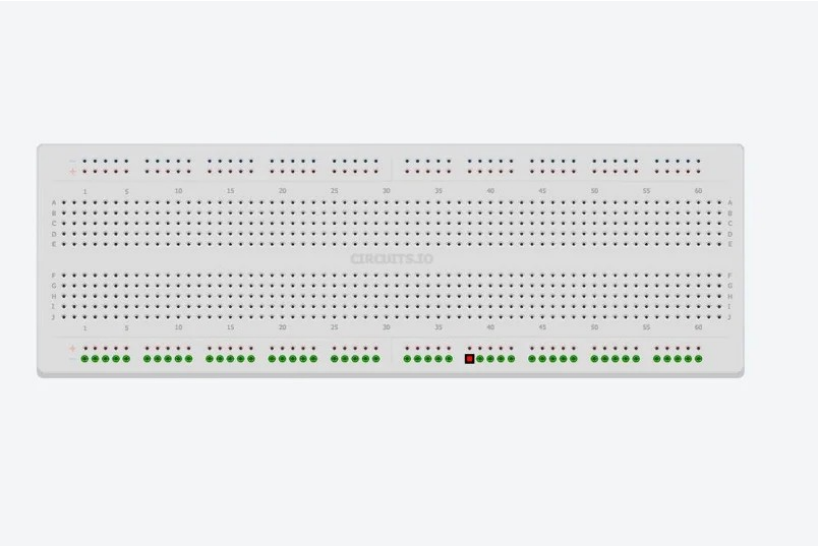
Kami dapat memuat program ke dalam chip melalui USB menggunakan Arduino IDE (unduh ini jika Anda belum melakukannya). Port USB juga menyediakan daya ke Arduino. Atau, kita dapat menyalakan papan yang diprogram menggunakan colokan listrik, dalam hal ini kita tidak memerlukan koneksi USB.

Arduino memiliki beberapa baris pin yang dapat digunakan untuk memasang kabel. Pin daya diberi label pada gambar di atas. Arduino memiliki suplai 3.3V atau 5V; di kelas ini kita akan menggunakan suplai 5V, tetapi Anda mungkin menemukan beberapa chip atau komponen yang membutuhkan 3.3V untuk dijalankan, dalam hal ini suplai 3.3V akan bermanfaat. Anda juga akan menemukan beberapa pin berlabel "GND" di Arduino, ini adalah pin ground (ground adalah hal yang sama dengan 0V). Arus listrik selalu mengalir dari beberapa tegangan positif ke ground, sehingga pin ini berguna untuk menyelesaikan rangkaian, kami akan sering menggunakannya.

Arduino memiliki 14 pin digital, berlabel 0-14, yang terhubung ke sirkuit untuk menghidupkan atau mematikannya, atau untuk mengukur tombol dan sirkuit 2-negara lainnya (tombol adalah dua keadaan karena baik ditekan atau tidak ditekan, sebagai lawan ke dial, yang memiliki berbagai kemungkinan status). Pin ini dapat bertindak sebagai input atau output, yang berarti mereka dapat mengontrol sirkuit atau mengukurnya.

Di samping koneksi daya adalah pin input Analog, berlabel A0-A5. Pin ini digunakan untuk membuat pengukuran analog sensor atau komponen lainnya. Input analog sangat baik untuk mengukur sesuatu dengan kisaran nilai yang mungkin. Misalnya, pin input analog akan memungkinkan kita mengukur jumlah kelenturan sensor kelenturan, atau jumlah pemutar yang diputar. Anda dapat menggunakan input analog untuk mengukur komponen digital (seperti tombol) atau bahkan bertindak seperti output digital, mereka pada dasarnya adalah pin digital dengan kekuatan ekstra.

Langkah 2: Cara Menggunakan Bread Board



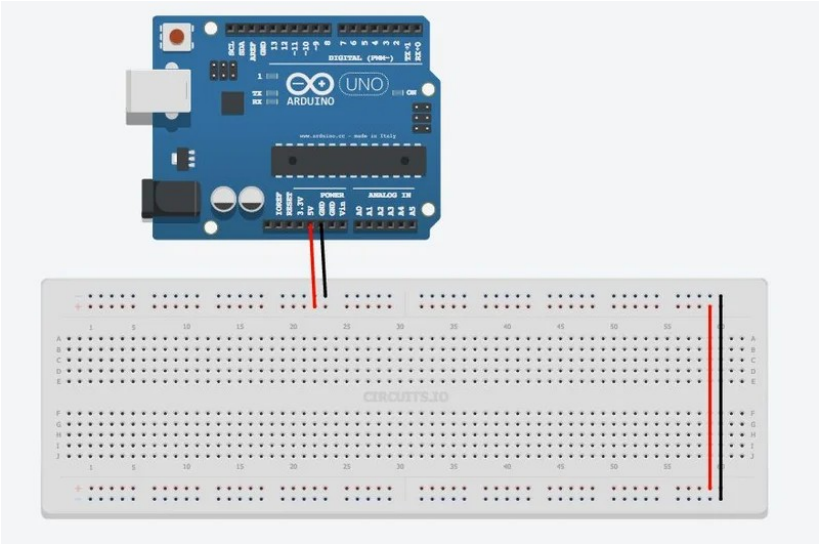
Breadboard memungkinkan kita membuat sambungan listrik sementara antar komponen sehingga kita dapat menguji sirkuit sebelum kita menyatukannya secara permanen. Seluruh kelas ini akan dilakukan pada papan tempat memotong roti sehingga kita dapat menggunakan kembali komponen dan membuat perubahan cepat ke sirkuit.

Papan tempat memotong roti memiliki deretan lubang tempat Anda dapat mencolokkan kabel atau komponen listrik lainnya. Beberapa lubang ini terhubung secara elektrik satu sama lain melalui strip logam di bagian bawah papan tempat memotong roti. Begini cara kerja koneksi:

Di setiap sisi papan tempat memotong roti, dua baris lubang terhubung di seluruh panjang papan (gambar 1 dan 2 di atas). Secara umum, Anda akan menghubungkan "rel" panjang ini ke 0V (juga disebut "ground") dan tegangan apa pun yang Anda gunakan untuk daya (di kelas ini kami akan menggunakan 5V dari Arduino), sehingga koneksi tersebut tersedia di mana saja di papan tulis. . Dalam hal ini, hal pertama yang ingin Anda lakukan adalah memasang koneksi ini ke Arduino Anda seperti yang ditunjukkan pada gambar 4, perhatikan bagaimana saya menghubungkan ground ke baris berlabel "-" dan 5V ke baris berlabel "+", papan tempat memotong roti Anda mungkin atau mungkin tidak diberi label. Catatan: terkadang strip samping ini hanya akan memanjang setengah melewati papan tempat memotong roti yang lebih panjang, gunakan kabel untuk menyelesaikan koneksi (gambar 5).

Sisa lubang di papan tempat memotong roti dikelompokkan menjadi lima baris di tengah papan tempat memotong roti (gambar 3). Di sinilah Anda akan menghubungkan komponen listrik satu sama lain untuk membentuk sirkuit.

Langkah 3: Menyalakan LED Menggunakan Listrik 5V



Seperti yang saya jelaskan sebelumnya, arus listrik mengalir dari tegangan tinggi ke tegangan rendah. Di kelas ini kita akan mematikan semuanya 5V dari Arduino, jadi arus akan mengalir dari 5V keluar dari Arduino, melalui sirkuit kita, dan kembali ke pin "ground" Arduino. Hal pertama yang akan kami nyalakan adalah LED.

Rangkaian yang menyalakan LED melibatkan dua komponen: resistor dan LED. Representasi skematis dari sirkuit ditunjukkan pada gambar 4 di atas. Resistor diwakili oleh kotak persegi panjang (Anda mungkin juga melihatnya diwakili oleh garis zigzag). LED diwakili oleh segitiga dengan

garis, dan biasanya beberapa panah mengarah ke luar yang mewakili cahaya yang keluar dari komponen.

Jadi mengapa kita membutuhkan resistor di sirkuit ini? Resistor ini disebut resistor pembatas arus, ini berarti resistor membatasi jumlah arus listrik yang mengalir melalui LED. Setiap LED dinilai untuk jumlah arus tertentu, jika Anda melebihi jumlah itu, Anda mungkin akan merusak LED. Dengan menggunakan Hukum Ohm, kita dapat menghitung nilai resistor pembatas arus yang harus kita gunakan dengan LED kita.

Hukum Ohm sangat sederhana, ia mengatakan bahwa ada hubungan linear antara arus dan tegangan dalam resistor: meningkatkan tegangan melintasi resistor akan meningkatkan arus yang mengalir melaluinya. Secara khusus dikatakan:

V = I * R
dimana
V = tegangan melintasi resistor
I = arus melalui resistor
R = resistensi - ini yang ingin kita hitung
jadi jika kita mengetahui nilai-nilai V dan I, kita dapat menghitung R yang benar untuk rangkaian kita

Pertama-tama kita perlu menghitung tegangan melintasi resistor. Di sirkuit yang ditunjukkan pada gambar 4, total 5V diterapkan ke sirkuit. Sebagian besar LED 3mm atau 5mm yang akan Anda gunakan

memerlukan 3V untuk menyala, sehingga sisa 2V (5V - 3V = 2V) diterapkan pada resistor.

Selanjutnya kita menghitung arus yang melalui resistor. Kebanyakan LED 3mm atau 5mm beroperasi pada kecerahan penuh sekitar 20mA saat ini; Jika ini terjadi, ini bisa merusak LED, dan ini akan membuat dimmer cahaya LED (tetapi tidak membahayakan). Dengan asumsi kita ingin menjalankan LED pada 20mA, kita tahu bahwa jumlah arus yang sama harus dijalankan melalui resistor karena komponen-komponen tersebut dihubungkan bersama secara seri. Ini membuat kita dengan:

2V = 20mA * R
2V = 0,02A * R
R = 100 Ohm

Jadi 100 Ohm adalah resistansi minimum absolut yang kami butuhkan untuk memastikan bahwa kami tidak merusak LED. Agar aman, adalah ide yang baik untuk menggunakan sesuatu yang sedikit lebih tinggi, kalau-kalau LED Anda memiliki peringkat yang sedikit berbeda dari apa yang saya gunakan di sini. Saya suka menggunakan 220Ohms karena saya sepertinya selalu punya banyak orang di sekitar. Jika Anda mengetahui peringkat LED Anda (Anda dapat menemukannya di lembar data LED) dan Anda ingin melakukan perhitungan ini sendiri, Anda juga dapat mencoba menggunakan kalkulator online.

Selanjutnya kita akan memasang LED di papan tempat memotong roti. Colokkan resistor dan LED ke bagian tengah papan tempat memotong

roti sehingga timah LED yang lebih panjang dihubungkan secara elektrik ke salah satu kabel resistor (gambar 3). Kemudian hubungkan ujung resistor yang tersisa ke 5V dan ujung LED yang tersisa ke ground. Anda akan melihat LED menyala.

Langkah 4: Anatomi Arduino

Program dalam bahasa Arduino disebut "sketsa". Sketsa Arduino terdiri dari dua bagian utama: fungsi pengaturan dan fungsi loop.

setup () - fungsi setup () secara otomatis dieksekusi pada awal program Arduino. Di dalam fungsi ini Anda akan menginisialisasi variabel, pin, dan perpustakaan apa pun yang mungkin Anda gunakan dalam sketsa Anda. Fungsi setup () hanya dijalankan sekali selama sketsa Arduino, tepat saat papan dinyalakan atau diatur ulang.

loop () - loop () adalah tempat sebagian besar program Anda akan hidup. Fungsi ini dieksekusi setelah setup () selesai. Arduino akan menjalankan perintah di dalam loop berulang-ulang sampai papan dimatikan.

Mulai sekarang, halaman referensi Arduino akan sangat berguna untuk dokumentasi tentang bahasa Arduino dan lingkungan pemrograman.

Langkah 5: Arduino LED Blink

Dalam contoh ini kita akan memasang rangkaian LED kita ke salah satu pin digital Arduino dan menghidupkan dan mematikan LED dengan kode. Contoh ini memperkenalkan beberapa fungsi berguna yang dibangun ke dalam bahasa Arduino, yaitu:

pinMode (pinNumber, mode) - pinMode digunakan selama pengaturan () bagian sketsa untuk menginisialisasi setiap pin yang kita gunakan sebagai input atau output. Kami tidak dapat membaca atau menulis ke pin sebelum pinMode telah ditetapkan. pinMode () membutuhkan dua argumen - nomor pin (masing-masing pin Arduino dilabeli dengan angka) dan mode yang kita inginkan pin (baik "INPUT" atau "OUTPUT"). Dalam kasus berkedip LED, kami mengirim data dari Arduino untuk mengontrol keadaan LED, jadi kami menggunakan "OUTPUT" sebagai argumen kedua.

digitalWrite (pinNumber, state) - digitalWrite adalah perintah yang memungkinkan kita mengatur voltase pin menjadi 5V atau ground (ingat "ground" identik dengan 0 Volts). Pada contoh terakhir kita menghubungkan LED ke suplai 5V dan melihatnya menyala, jika kita menghubungkan LED ke salah satu pin digital Arduino, kita bisa menyalakan LED dengan mengatur pin ke 5V dan mematikan dengan mengatur pin ke tanah. digitalWrite () juga membutuhkan dua argumen - nomor pin dan status pin ("HIGH" untuk 5V dan "LOW" untuk ground).

delay (timeInMs) - delay menjeda program untuk jumlah waktu tertentu. Misalnya, penundaan (2000) akan menghentikan sementara program

selama 2000 milidetik (2000 milidetik = 2 detik), penundaan (100) akan menghentikan sementara program selama 100 milidetik (1/10 detik), dan seterusnya ...

Di bawah ini adalah kode LED Blink, jalankan kode ini di Arduino Anda.

```
//LED Blink

int ledPin = 7;//the Arduino pin that is connected to the LED

void setup() {
  pinMode(ledPin, OUTPUT);// initialize the pin as an output
}

void loop() {
  digitalWrite(ledPin, HIGH);//turn LED on
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(ledPin, LOW);//turn LED off
  delay(1000);//wait one second
}
```

Langkah 6: Mengendalikan Beberapa LED Menggunakan Arduino

Dalam contoh ini kita akan memasang tiga LED lagi seperti yang kita lakukan pada contoh terakhir, dan mengendalikannya dengan beberapa pin digital. Pertama pasang tiga LED lagi dan resistor pembatas arus seperti yang ditunjukkan di bawah ini:

```
//Multi LED Blink

int led1Pin = 4;
int led2Pin = 5;
int led3Pin = 6;
int led4Pin = 7;

void setup() {
  //initialize the led pins as an outputs
  pinMode(led1Pin, OUTPUT);
  pinMode(led2Pin, OUTPUT);
  pinMode(led3Pin, OUTPUT);
  pinMode(led4Pin, OUTPUT);
}

void loop() {
  digitalWrite(led1Pin, HIGH);//turn LED on
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(led1Pin, LOW);//turn LED off
  delay(1000);//wait one second

  //do the same for the other 3 LEDs
  digitalWrite(led2Pin, HIGH);//turn LED on
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(led2Pin, LOW);//turn LED off
  delay(1000);//wait one second

  digitalWrite(led3Pin, HIGH);//turn LED on
```

```
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(led3Pin, LOW);//turn LED off
  delay(1000);//wait one second

  digitalWrite(led4Pin, HIGH);//turn LED on
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(led4Pin, LOW);//turn LED off
  delay(1000);//wait one second
}
```

Ini berfungsi, dan kita bisa membiarkannya dan semuanya akan bekerja dengan baik, tetapi itu bukan cara yang paling efisien untuk menulis kode kita. Sebagai gantinya, kami akan menggunakan struktur yang disebut for loop untuk berputar melalui LED. Untuk loop berguna untuk mengulangi sepotong kode berulang-ulang. Dalam kasus di atas kami mengulangi baris:

digitalWrite(led4Pin, HIGH);
delay(1000);
digitalWrite(led4Pin, LOW);
delay(1000);

inilah cara kami akan menulis loop for:

for (int ledPin=4;ledPin<8;ledPin++){

digitalWrite(ledPin, HIGH);
delay(1000);
digitalWrite(ledPin, LOW);

delay(1000);

}

Pada baris pertama kita menginisialisasi variabel "ledPin" sebagai 4 dan memberitahu Arduino bahwa kita ingin menggilir nilai-nilai variabel mulai dari 4, hingga 7 (ledPin <8). The ledPin ++ memberitahu Arduino untuk meningkatkan nilai ledPin sebanyak 1 setiap kali kita mengulangi loop. Kemudian kita jalankan baris di dalam loop menggunakan variabel ledPin. Jadi ledPin pertama kali = 4, dan pin 4 dihidupkan kemudian dimatikan, kemudian ledPin dinaikkan menjadi 5 dan untuk loop dimulai lagi, kali ini menghidupkan pin 5 pada saat mati, dan seterusnya ... Hasilnya persis sama dengan sketsa lebih verbose di atas, di mana kami mengulangi perintah digitalWrite dan menunda berkali-kali. Berikut ini sketsa lengkapnya:

```
//Multi LED Blink

int led1Pin = 4;
int led2Pin = 5;
int led3Pin = 6;
int led4Pin = 7;

void setup() {
  //initialize the led pins as an outputs
  pinMode(led1Pin, OUTPUT);
  pinMode(led2Pin, OUTPUT);
  pinMode(led3Pin, OUTPUT);
  pinMode(led4Pin, OUTPUT);
}
```

```
void loop() {
  for (int ledPin=4;ledPin<8;ledPin++){//for pins 4-7
    digitalWrite(ledPin, HIGH);//turn LED on
    delay(1000);// wait for 1000 milliseconds (one second)
    digitalWrite(ledPin, LOW);//turn LED off
    delay(1000);//wait one second
  }
}
```

Langkah 7: Fade LED menggunakan AnalogWrite

Terkadang kita ingin mengontrol kecerahan LED, dalam hal ini kita dapat menggunakan perintah yang disebut analogWrite (). analogWrite bekerja dengan menyalakan dan mematikan LED dengan sangat cepat, begitu cepat sehingga mata kita tidak melihat layar berkedip. Jika sebuah LED menghabiskan separuh waktu dan separuh waktunya, maka akan tampak setengah lebih terang. Teknik ini disebut modulasi lebar pulsa (PWM), ini digunakan berulang kali dalam elektronik karena memungkinkan kita untuk mengontrol komponen dengan cara "analog" menggunakan pin digital. Tidak semua pin digital pada Arduino dapat melakukan PWM, jika Anda melihat dari dekat pada Arduino Anda, Anda akan melihat bahwa beberapa pin memiliki tanda "~" di sebelahnya (pin 3, 5, 6, 9, 10, 11) , ini adalah pin yang diaktifkan PWM.

Hubungkan salah satu LED Anda ke pin berkemampuan PWM, saya menggunakan pin 9. Coba jalankan sketsa kedip dari sebelumnya, tetapi gunakan analogWrite alih-alih digitalWrite untuk menghidupkan LED (lihat sketsa di bawah). analogWrite () membutuhkan dua argumen: nomor pin dan tingkat kecerahan (antara 0 dan 255).

```
//LED Blink (half brightness)

int ledPin = 9;//the Arduino pin that is connected to the LED

void setup() {
  pinMode(ledPin, OUTPUT);// initialize the pin as an output
}
```

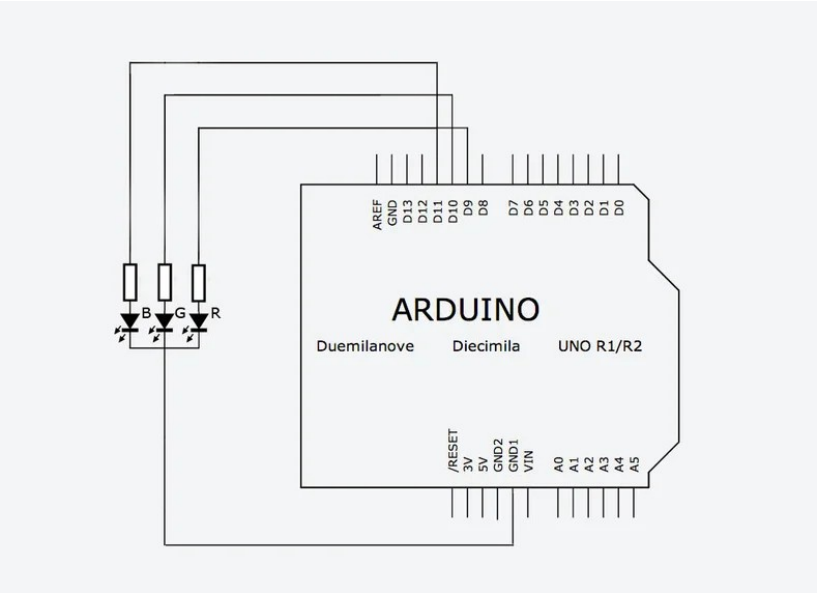
```
void loop() {
  analogWrite(ledPin, 255);//turn LED on at full brightness (255/255 =
1)
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(ledPin, LOW);//turn LED off
  delay(1000);//wait one second

  analogWrite(ledPin, 191);//turn LED on at 3/4 brightness (191/255 ~=
0.75)
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(ledPin, LOW);//turn LED off
  delay(1000);//wait one second

   analogWrite(ledPin, 127);//turn LED on at half brightness (127/255
~= 0.5)
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(ledPin, LOW);//turn LED off
  delay(1000);//wait one second

   analogWrite(ledPin, 63);//turn LED on at one quarter brightness
(63/255 ~= 0.25)
  delay(1000);// wait for 1000 milliseconds (one second)
  digitalWrite(ledPin, LOW);//turn LED off
  delay(1000);//wait one second
}
```

Langkah 8: RGB LED dan Arduino



LED RGB benar-benar menyenangkan, seperti yang Anda lihat pada gambar pertama di atas, setiap LED RGB sebenarnya terdiri dari tiga LED: satu merah, satu hijau, dan satu biru. Jika Anda menyalakan beberapa LED sekaligus mereka akan berbaur untuk membentuk warna baru.

RGB LED yang kami gunakan di kelas ini adalah katoda umum, artinya ketiga LED berbagi pin ground yang sama (beberapa LED RGB, disebut common anode, berbagi pin pasokan bersama dan memiliki alasan terpisah). Kami akan memasang sirkuit kami seperti gambar pertama di atas, setiap LED dalam LED RGB memiliki satu resistor 220Ohm secara seri

dengan itu yang dihubungkan ke pin Arduino yang diaktifkan PWM (saya menggunakan pin 9-11). Dengan cara ini, kita dapat secara selektif menyalakan dan mematikan setiap LED di RGB secara individual.

Lihat gambar kedua di atas untuk mencari tahu yang mengarah dari LED RGB yang sesuai dengan merah, hijau, biru, dan tanah (mereka diberi nomor 1-4).

Sketsa pertama ini akan menggilir setiap warna dalam LED:

```
//RGB LED - test

//pin connections
int red = 9;
int green = 10;
int blue = 11;

void setup(){
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}

void loop(){
  //turn red led on
  digitalWrite(red, HIGH);
  delay(500);
  digitalWrite(red, LOW);
  delay(500);

  //turn green led on
  digitalWrite(green, HIGH);
  delay(500);
  digitalWrite(green, LOW);
```

```
  delay(500);

  //turn blue led on
  digitalWrite(blue, HIGH);
  delay(500);
  digitalWrite(blue, LOW);
  delay(500);
}
```

Kemudian gunakan analogWrite () dan random () untuk mengatur tingkat kecerahan acak untuk masing-masing warna dalam LED. Tiga warna akan bercampur dalam proporsi yang berbeda (tergantung pada kecerahannya) untuk membuat berbagai warna (255 ^ 3 = 16.581.375 warna yang mungkin).

```
//RGB LED - random colors

//pin connections
int red = 9;
int green = 10;
int blue = 11;
void setup(){
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop(){
  //pick a random color
  analogWrite(red, random(256));
  analogWrite(blue, random(256));
  analogWrite(green, random(256));
  delay(1000);//wait one second
}
```

Langkah 9: Fungsi Arduino

Sketsa berikut memudar LED dari merah ke hijau ke biru ke merah ke hijau dan seterusnya.

```
//RGB LED - fading between colors
//pin connections
int red = 9;
int green = 10;
int blue = 11;
void setup(){
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop(){
  for (int brightness=0;brightness<256;brightness++){
    analogWrite(red, 255-brightness);
    analogWrite(green, brightness);
    delay(10);
  }
  for (int brightness=0;brightness<256;brightness++){
    analogWrite(green, 255-brightness);
    analogWrite(blue, brightness);
    delay(10);
  }
  for (int brightness=0;brightness<256;brightness++){
    analogWrite(blue, 255-brightness);
    analogWrite(red, brightness);
    delay(10);
  }
}
```

Sketsa di atas berfungsi, tetapi ada banyak kode berulang. Kita dapat menyederhanakan dengan menulis fungsi pembantu kita sendiri yang memudar dari satu warna ke warna lain. Seperti apa fungsinya nanti:

```
void fader (int color1, int color2) {

untuk (kecerahan int = 0; kecerahan <256; kecerahan ++) {
analogWrite (color1, 255-brightness);
analogWrite (color2, brightness);
keterlambatan (10);
}

}
```

Mari kita periksa definisi fungsi ini sepotong demi sepotong. Fungsinya disebut "fader" dan dibutuhkan dua argumen. Setiap argumen dipisahkan oleh koma dan memiliki tipe yang dideklarasikan di baris pertama dari definisi fungsi:

```
void fader (int color1, int color2) {
```

Kita dapat melihat bahwa kedua argumen yang diterima fader adalah int, dan kami menggunakan nama "color1" dan "color2" sebagai variabel dummy untuk definisi fungsi kami. "Void" mengacu pada tipe data yang dikembalikan fungsi, karena fungsi kami tidak mengembalikan apa pun (hanya mengeksekusi perintah), kami menetapkan tipe kembali ke batal. Jika kami membuat fungsi yang mengalikan dua angka dan mengembalikan produk, kami mungkin mendefinisikannya seperti ini:

```
pengali int (int number1, int number2) {
```

produk int = number1 * number2;
produk kembali;

}

Perhatikan bagaimana kami menyatakan int sebagai tipe pengembalian di sini, bukan batal.

Nyali fungsi adalah hal-hal yang telah kita lihat sebelumnya. Itu sama untuk loop yang kami ulangi di sketsa terakhir kami, tetapi nomor pin telah diganti dengan variabel color1 dan color2. Jika kami memanggil:

fader (merah, hijau);

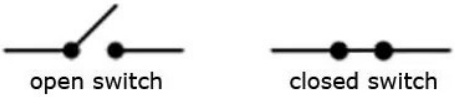
dari loop Arduino (), Arduino mengevaluasi fungsi fader dengan color1 = merah dan color2 = hijau.

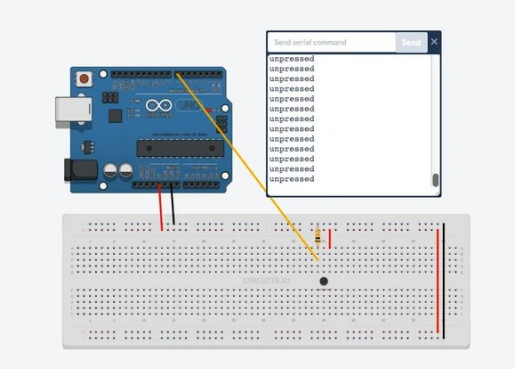
Dengan menggabungkan semua ini, kita dapat menulis ulang sketsa menggunakan fungsi ini sebagai berikut, ini akan bekerja persis sama dengan sketsa di bagian atas langkah ini.

```
//RGB LED - fading between colors
//pin connections
int red = 9;
int green = 10;
int blue = 11;
void setup(){
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
```

```
  pinMode(green, OUTPUT);
}
void loop(){
  fader(red,green);
  fader(green,blue);
  fader(blue, red);
}
void fader(int color1, int color2){
    for (int brightness=0;brightness<256;brightness++){
        analogWrite(color1, 255-brightness);
        analogWrite(color2, brightness);
        delay(10);
    }
}
```

Langkah 10 : Tombol





Saatnya untuk jenis sirkuit baru, sekarang kita akan melihat cara menggunakan tombol push dengan Arduino. Tombol adalah jenis sakelar, jenis tombol yang kami gunakan disebut "sakelar sesaat yang biasanya terbuka". "Biasanya terbuka" artinya ketika tombol tidak ditekan, tidak ada arus yang akan mengalir melalui tombol karena kedua belah pihak tidak terhubung - itu terbentuk sebagai sirkuit terbuka (lihat gambar pertama di atas). "Sesaat" mengacu pada kenyataan bahwa saklar ini

hanya ditutup selama Anda menekannya dengan jari Anda; ini membuatnya sangat berbeda dari sakelar sakelar, yang berganti-ganti antara keadaan terbuka dan tertutup setiap kali Anda menekannya.

Rangkaian tombol yang akan kami gunakan terbuat dari dua komponen - tombol tekan dan resistor. Tidak seperti rangkaian LED, kita tidak peduli tentang jumlah arus yang melewati tombol (pada tingkat arus yang sangat tinggi kita mungkin harus khawatir tentang peleburan tombol, tetapi Arduino tidak sekuat itu), sehingga resistor tidak bekerja seperti resistor pembatas arus di sirkuit LED. Sebaliknya resistor ini bertindak sebagai resistor pull-down. Sebuah resistor pull-down mengikat tombol ke ground, sehingga mengukur tegangan di persimpangan antara tombol dan resistor akan selalu 0V (membumi) ketika tombol tidak ditekan (dan sirkuit terbuka). Di sirkuit ini, nilai resistor pull-down tidak terlalu penting, saya suka menggunakan sesuatu sekitar 10kOhms.

```
//Button Press Detection
int buttonPin = 7;
void setup(){
  pinMode(buttonPin, INPUT);//this time we will set the pin as INPUT
  Serial.begin(9600);//initialize Serial connection
}
void loop(){
  if (digitalRead(buttonPin)==HIGH){//if button pressed
    Serial.println("pressed");
  } else {
    Serial.println("unpressed");
  }
}
```

Sketsa tombol memperkenalkan beberapa ide baru:

digitalRead (pinNumber) - mirip dengan digitalWrite (), tetapi digunakan untuk mengukur nilai TINGGI atau RENDAH di sirkuit kami. digitalRead () mengambil satu argumen - nomor pin yang kita baca. Kami juga harus memastikan untuk menginisialisasi pin input dengan benar:

pinMode (buttonPin, INPUT);

Serial Communication - Serial communication memungkinkan Arduino mengirim pesan ke komputer Anda saat program sedang berjalan, ini berguna untuk debugging, mengirim pesan ke perangkat atau aplikasi lain, atau hanya mendapatkan pemahaman yang lebih baik tentang apa yang terjadi di sirkuit Anda. Untuk mengaktifkan komunikasi serial di sketsa Anda, Anda harus menginisialisasi koneksi serial di fungsi setup () Arduino dengan perintah Serial.begin (). Serial.begin () mengambil satu argumen, baud rate, yaitu laju transfer data antara Arduino dan komputer Anda, 9600 adalah baud rate yang bagus untuk saat ini. Dalam sketsa berikut, kami akan menggunakan Serial.println () untuk mencetak pesan di Arduino IDE (Tools >> Serial Monitor).

if / else - If / else statement memberi kita kendali lebih besar atas perintah mana yang dieksekusi ketika. Pada sketsa tombol saya menggunakan pernyataan if / else berikut:

if (digitalRead (buttonPin) == HIGH) {

```
Serial.println ("ditekan");

} lain {

Serial.println ("tidak tertekan");

}
```

jika hasil digitalRead (buttonPin) mengembalikan HIGH maka Arduino mencetak kata "pressed", jika digitalRead (buttonPin) mengembalikan sesuatu selain HIGH (seperti RENDAH), Arduino mencetak kata "unpressed". Jika pernyataan dapat memeriksa == ("sama dengan"),! = ("Tidak sama dengan"),>, <,> =, dan <=. Coba jalankan pernyataan berikut jika dalam loop Arduino ():

```
if (4> 3) {

Serial.println ("true");

} lain {

Serial.println ("false");

}
```

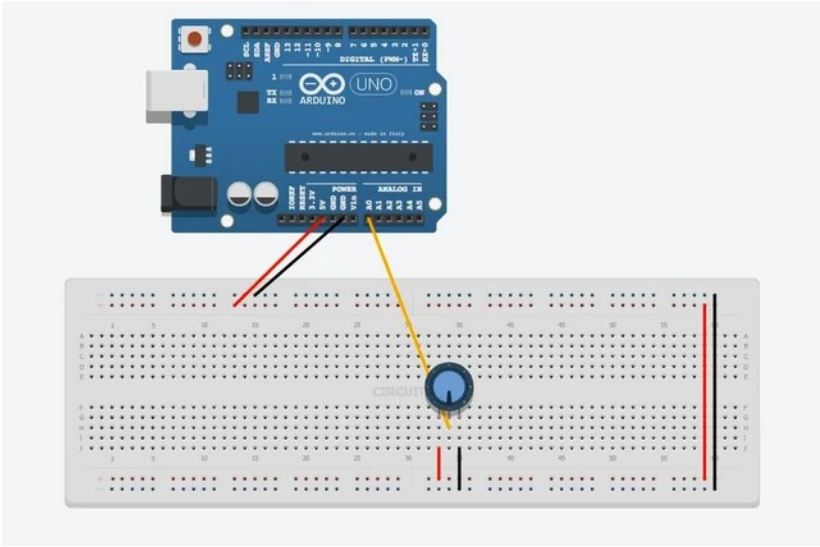
Coba ubah pernyataan if untuk mengevaluasi hal-hal lain.

Langkah 11: Arduino Digital Inputs dan Outputs

Sekarang kita dapat menggunakan data dari tombol tekan untuk menyalakan dan mematikan LED. Ubah sketsa dari langkah terakhir untuk menyalakan LED yang terhubung ke pin 8:

```
//button press detection with LED output
int buttonPin = 7;
int ledPin = 8;
void setup(){
  pinMode(buttonPin, INPUT);//this time we will set button pin as INPUT
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}
void loop(){
  if (digitalRead(buttonPin)==HIGH){
    digitalWrite(ledPin,HIGH);
    Serial.println("pressed");
  } else {
    digitalWrite(ledPin,LOW);
    Serial.println("unpressed");
  }
}
```

Step 12: Arduino Analog Input



analogRead (pinNumber) - analogRead () membaca dari salah satu pin analog Arduino dan menampilkan nilai antara 0 (tegangan pada pin = 0V) dan 1023 (tegangan pada pin = 5V), jika tegangan pin analog 2,5V, maka akan dicetak:

2.5 / 5 * 1023 = 512

analogRead () membutuhkan satu argumen - nama pin analog (A0-A5) untuk dibaca.

Potensiometer adalah sebuah resistor dengan pin di tengah yang terhubung ke beberapa titik sepanjang panjang resistor. Saat Anda memutar potensiometer, Anda memindahkan pin tengah di sepanjang resistor dan mengubah rasio bahan resistif di kedua sisi pin. Ini memungkinkan potensiometer untuk bertindak sebagai pembagi tegangan variabel.

Hubungkan potensiometer sehingga pin luar terhubung ke 5V dan ground (orientasi tidak masalah), dan pin tengah terhubung ke pin A0 pada Arduino. Jalankan kode berikut dan perhatikan hasilnya dari Serial Monitor.

```
//analog input

int potPin = A0;//center pin of the potentiometer is attached to pin A0

void setup(){
  //analog pins are initilized as INPUT by default, no need for pinMode() command
  Serial.begin(9600);
}

void loop(){
  int potVal = analogRead(potPin);//potVal is a number between 0 and 1023
  Serial.println(potVal);
}
```

Sekarang putar pot dan lihat bagaimana nilai cetakan perubahan potVal. Anda harus melihat arduino print 1023 ketika Anda memutar pot sampai ke sisi yang terhubung ke 5V, dan 0 ketika Anda memutar pot sampai ke sisi yang lain. Anda juga harus melihat rentang nilai yang dicetak di antara kedua ekstrem itu.

Langkah 13: Bekerja Dengan Analog Input Data

Sebelum menggunakan data analog untuk mengontrol hal-hal lain dalam program Anda, Anda mungkin perlu mengatur atau membatasi antara beberapa min dan maks. Misalnya, bayangkan Anda ingin menggunakan bacaan dari input analog Anda untuk mengontrol kecerahan LED dengan analogWrite (). analogRead () mengembalikan angka antara 0 dan 1023, tetapi analogWrite () hanya menerima angka antara 0 dan 255. Dalam hal ini Anda dapat menggunakan peta () untuk skala rentang nilai yang keluar dari analogRead () ke sesuatu yang sesuai untuk analogWrite () ;

map (value, fromLow, fromHigh, toLow, toHigh) - skala satu rentang ke yang lain. map () menerima empat input: nilai yang kami coba skala, min rentang kami scaling dari, max rentang kami scaling dari, min dari rentang kami scaling ke, dan maks dari rentang yang kami scaling.

Berikut ini sebuah contoh:

```
//analog input with map

int potPin = A0;
int ledPin = 9;

void setup(){
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop(){
  int analogVal = analogRead(potPin);//analogVal is between 0 and 1023
   int scaledVal  = map(analogVal, 0, 1023, 0, 255);//scaled val is between 0 and
255
```

```
  Serial.print("analogVal = ");
  Serial.print(analogVal);
  Serial.print("   scaledVal = ");
  Serial.print(scaledVal);
  analogWrite(ledPin, scaledVal);
}
```

Langkah 14: Latihan Dengan Arduino Input dan Output

Contoh berikut ini menggabungkan sketsa deteksi tombol dengan sketsa kontrol LED analog. Pasang sebuah tombol untuk menyematkan 7, seperti yang ditunjukkan dalam skema dari langkah 10, dan sambungkan pot ke A0 dan LED ke pin 9, seperti yang ditunjukkan pada langkah 13. Kemudian unggah kode berikut:

```
//button press detection with LED output and variable intensity
int buttonPin = 7;
int ledPin = 9;
int potPin = A0;
void setup(){
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}
void loop(){
  if (digitalRead(buttonPin)==HIGH){//if button pressed
    int analogVal = analogRead(potPin);
    int scaledVal  = map(analogVal, 0, 1023, 0, 255);
    analogWrite(ledPin, scaledVal);//turn on led with intensity set by pot
    Serial.println("pressed");
  } else {
    digitalWrite(ledPin, LOW);//turn off if button is not pressed
    Serial.println("unpressed");
  }
}
```

Sketsa ini mematikan dan menghidupkan LED tergantung pada keadaan tombol (ditekan / tidak ditekan), dan pada saat yang sama menggunakan potensiometer untuk mengontrol kecerahan LED ketika sedang dalam keadaan "on".

Langkah 15: Tombol Toggle

Terkadang Anda akan tertarik pada saat yang tepat tombol ditekan atau dilepaskan, sehingga Anda dapat memicu suatu peristiwa dalam sketsa Anda. Dalam hal ini Anda perlu menyimpan Status saat ini dari tombol dan membandingkannya dengan keadaan yang terakhir direkam. Jika status saat ini adalah TINGGI dan status terakhir adalah RENDAH, maka Anda tahu tombolnya baru saja ditekan. Lihatlah kode di bawah ini:

```
//Button Press Detection - single message

int buttonPin = 7;
boolean currentState = LOW;//stroage for current button state
boolean lastState = LOW;//storage for last button state

void setup(){
  pinMode(buttonPin, INPUT);//this time we will set the pin as INPUT
  Serial.begin(9600);//initialize Serial connection
}

void loop(){
  currentState = digitalRead(buttonPin);
  if (currentState == HIGH && lastState == LOW){//if button has just been pressed
    Serial.println("pressed");
    delay(1);//crude form of button debouncing
  } else if(currentState == LOW && lastState == HIGH){
    Serial.println("released");
    delay(1);//crude form of button debouncing
  }
  lastState = currentState;
}
```

Saya menggunakan sesuatu yang baru dalam pernyataan if saya:

if (currentState == HIGH && lastState == LOW)

ini berbunyi "jika currentState adalah TINGGI dan lastState adalah RENDAH", && memungkinkan kita untuk memeriksa kebenaran banyak hal dalam pernyataan if yang sama. Anda juga dapat menggunakan || ("atau") untuk menguji adalah satu hal atau yang lain benar. Baca lebih lanjut di sini.

Anda juga akan melihat baris berikut muncul dua kali dalam kode di atas:

keterlambatan (1);

Penundaan ini dilakukan di sana untuk memberikan waktu tombol untuk menyelesaikan tegangan stabil sebelum kita mulai mengukurnya lagi, ini disebut tombol debouncing; itu mencegah kita menghitung satu pers sebagai dua penekanan karena obrolan tombol. Menggunakan penundaan untuk melakukan pelepasan tombol tidak masalah untuk contoh sederhana ini, tetapi jika Anda mengukur banyak tombol, penundaan akan bertambah dan membuat kode Anda dieksekusi sangat lambat. Ini mungkin pada akhirnya memberikan hardware Anda perasaan tertinggal. Saya akan membahas beberapa teknik yang lebih baik untuk melakukan debouncing di kelas ini nanti.

Kode ini juga memperkenalkan tipe data baru: boolean. Boolean digunakan untuk menyimpan 1 bit informasi, hal-hal seperti true / false, on / off, 1/0, dari HIGH / LOW. Dalam kode saya, saya menggunakannya

untuk menyimpan keadaan tombol saat ini dan terakhir (TINGGI atau RENDAH).

Inilah cara kami dapat menggunakan ini untuk menyalakan dan mematikan LED setiap kali tombol ditekan:

```
//Button Toggle LED

int ledPin = 9;
int buttonPin = 7;
boolean currentState = LOW;//stroage for current button state
boolean lastState = LOW;//storage for last button state
boolean ledState = LOW;//storage for the current state of the LED (off/on)

void setup(){
  pinMode(buttonPin, INPUT);//this time we will set the pin as INPUT
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);//initialize Serial connection
}

void loop(){
  currentState = digitalRead(buttonPin);
  if (currentState == HIGH && lastState == LOW){//if button has just been pressed
    Serial.println("pressed");
    delay(1);//crude form of button debouncing

    //toggle the state of the LED
    if (ledState == HIGH){
      digitalWrite(ledPin, LOW);
      ledState = LOW;
    } else {
      digitalWrite(ledPin, HIGH);
      ledState = HIGH;
    }
  }
```

```
  lastState = currentState;
}
```

Dalam kode di atas saya mengatur variabel yang disebut "ledState" untuk menyimpan keadaan LED saat ini, lalu setiap kali tombol ditekan, saya menggunakan digitalWrite untuk mengatur LED ke keadaan sebaliknya dan menyimpan ledState baru.

Lebih jauh lagi, Anda dapat menggunakan tombol sakelar kode dengan kode fader dari contoh LED RGB sebagai berikut:

```
//Button Press Detection - single message

//pin connections
int red = 9;
int green = 10;
int blue = 11;
int buttonPin = 7;

boolean currentState = LOW;//stroage for current button state
boolean lastState = LOW;//storage for last button state
int currentColor = red;//storage for current color

void setup(){
  pinMode(buttonPin, INPUT);//this time we will set the pin as INPUT
  pinMode(red, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
  Serial.begin(9600);//initialize Serial connection
     digitalWrite(currentColor,  HIGH);//initialize  with  currentColor  on  (full
brightness)
}

void loop(){
```

```
  currentState = digitalRead(buttonPin);
  if (currentState == HIGH && lastState == LOW){//if button has just been pressed
    Serial.println("pressed");
    delay(1);//crude form of button debouncing

    int nextColor = getNextColor(currentColor);
    fader(currentColor, nextColor);
    currentColor = nextColor;

  }

  lastState = currentState;

}

int getNextColor(int color){//helper function that gives us the next color to fade
to
  if (color == red) return green;
  if (color == green) return blue;
  if (color == blue) return red;
}

void fader(int color1, int color2){
    for (int brightness=0;brightness<256;brightness++){
        analogWrite(color1, 255-brightness);
        analogWrite(color2, brightness);
        delay(2);
    }
}
```

Saya menambahkan fungsi pembantu tambahan dalam kode di atas untuk membantu memilih warna berikutnya yang akan pudar:

int getNextColor (int color) {

jika (warna == merah) kembali hijau;
jika (warna == hijau) kembali biru;
jika (warna == biru) kembali merah;

}

Saya mendeklarasikan fungsi dengan int untuk memberi tahu Arduino bahwa ia seharusnya mengharapkan fungsi mengembalikan nomor (dalam hal ini, jumlah pin Arduino yang terhubung ke salah satu pin LED RGB. Pernyataan if terlihat sedikit berbeda dengan apa yang yang telah kita lihat sebelumnya, saya bisa menulis fungsi seperti ini:

```
int getNextColor(int color){
  if (color == red) {
    return green;
  }
  if (color == green) {
    return blue;
  }
  if (color == blue) {
    return red;
  }
}
```