Minicurso de Arduino

Retos para aprender y enseñar con Arduino

Adaptado a uso de Shield Edubasica (de @leobotmanuel)

Autor : *Pedro Ruiz Fernández* (@pedroruizf)

Fuentes utilizadas:

- Adaptación y modificación de Curso de Arduino de *Prudencio Luna Belizón* (@plunax)
- Ejercicios de Arduino resueltos (de @pedroruizf)
- Taller de Edubasica de *Manuel Hidalgo* (@leobotmanuel)

Versión: 21/04/2019

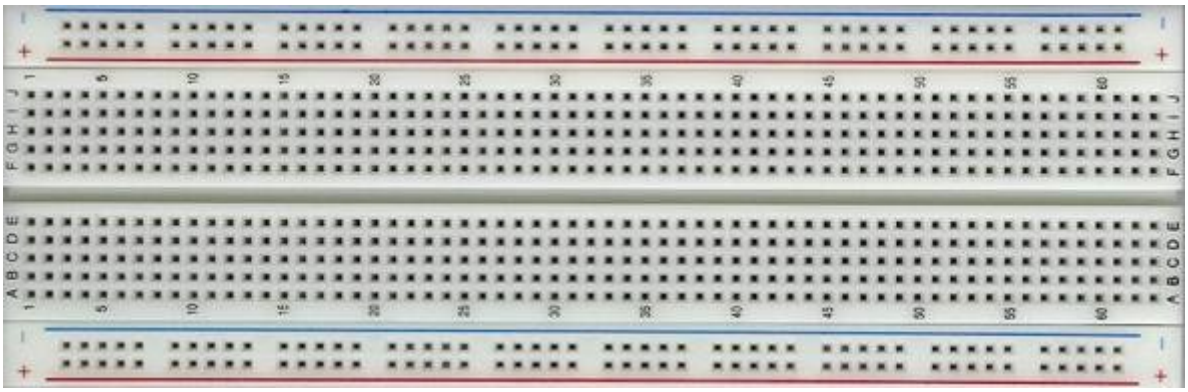
Reto 0: La protoboard (placa de pruebas)

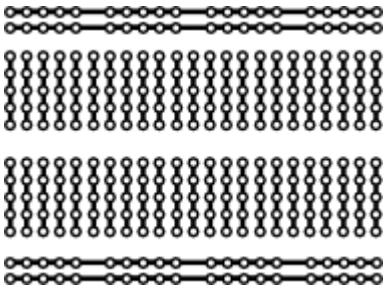
En este caso el reto consiste en iluminar un led, alimentándolo directamente con arduino, por tanto no lo programaremos. Para ello utilizaremos una placa de pruebas, una resistencia, un led y los terminales (5V y GND) de la placa de Arduino.

El objetivo es conocer la forma de conexionado en la placa de pruebas.

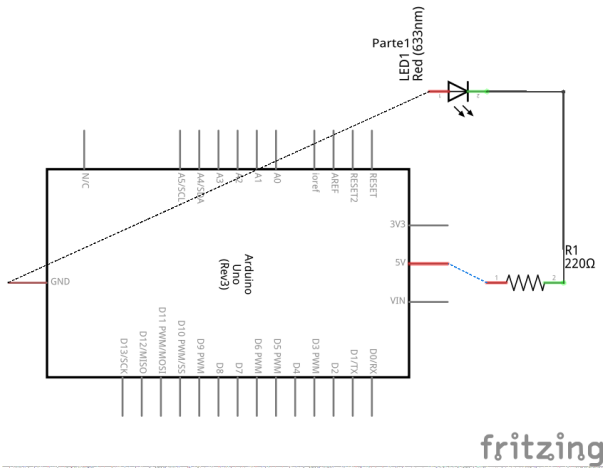
¿Qué es una protoboard?

Es un tablero con orificios que se encuentran conectados eléctricamente entre sí.

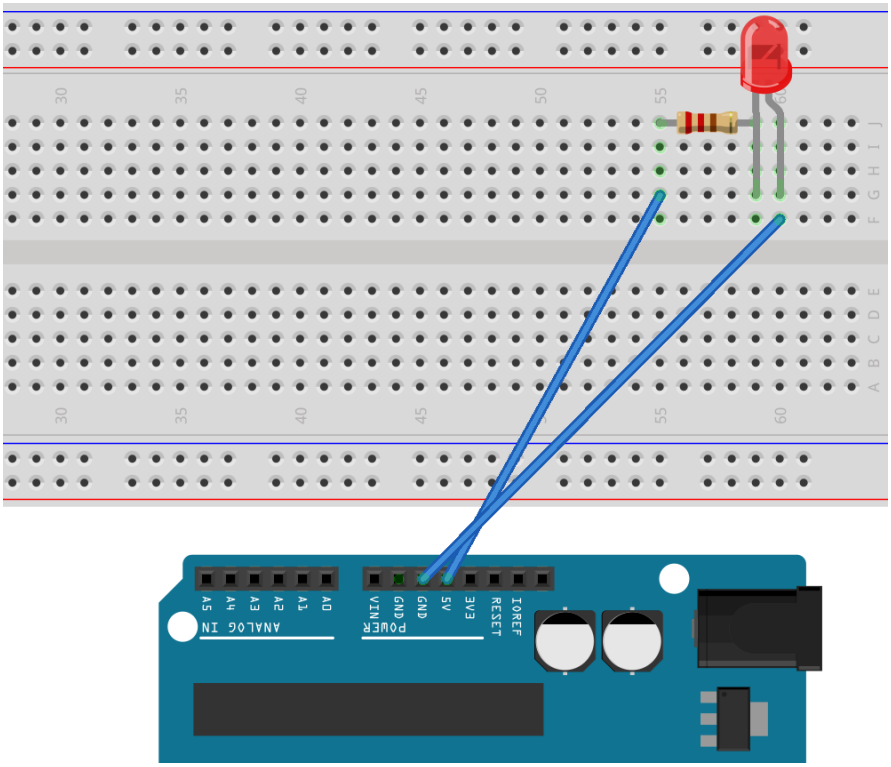




El esquema del circuito eléctrico a montar será el siguiente:



La forma de conexión será la que se muestra en la figura.


30
35
40
45
50
55
60
F G H I J
A B C D E
ANALOG IN
A5 A4 A3 A2 A1 A0
POWER
VIN GND GND 5V 3V3 RESET IOREF

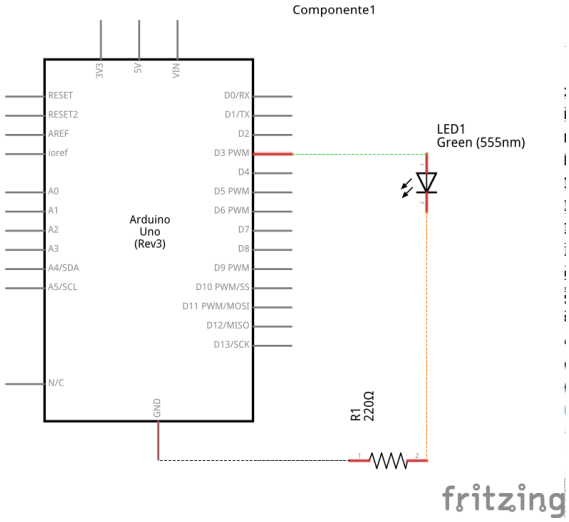
Reto 1: LED parpadeante

El reto consiste en iluminar de manera intermitente un led utilizando cualquier pin digital de Arduino, en nuestro caso vamos a usar el 3, que edubasica lo provee de un led verde. El led estará encendido durante 500ms y que se apague 100ms y así de forma ininterrumpida. Si se utiliza el pin 13 no hace falta resistencia en otros casos sí.

Objetivos:

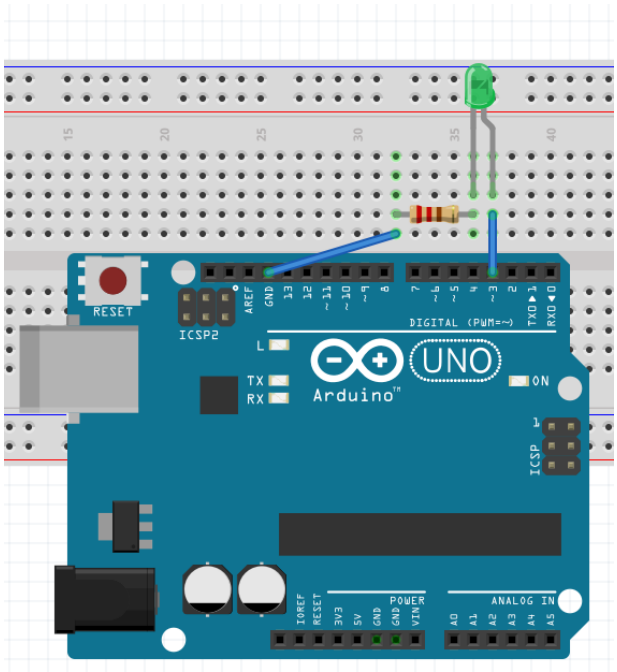
- Familiarizarse con el entorno de programación.
- Reconocer las partes de un programa de Arduino.
- Conocer órdenes como: pinMode, digitalWrite y delay.

Esquema



Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto01 §

```
void setup() {    //comienza la configuracion
  pinMode(3, OUTPUT);//configura el pin 3 o el que queramos como salida
}           //termina la configuracion

void loop() {         //comienza el bucle principal del programa
  digitalWrite(3, HIGH);  //envia 5V al pin (salida) 3
  delay (500);      //espera 500 ms pin 3 con 5V
  digitalWrite(3, LOW); //envia 0V al pin (salida) 3
  delay (100);      //espera 100 ms pin 3 con 0V
}
```

Reto 2: Secuencia de leds

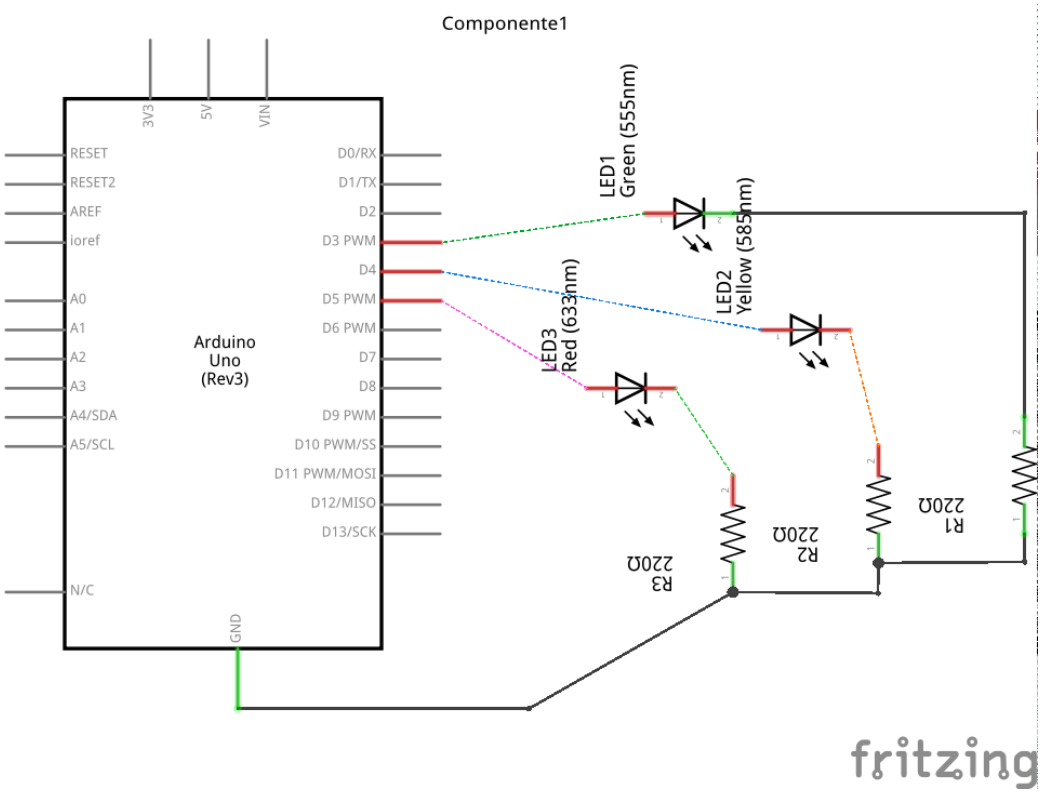
Se trata de encender y apagar 3 leds secuencialmente. Los leds deben estar conectados a los pines 3, 4 y 5 (los pines de los leds de edubasica). Tanto el tiempo de encendido y apagado será de 200 milisegundos.

Nota: en una segunda solución la secuencia principal del programa debe estar reproducida en una función a la que llamará el programa principal.

Objetivos:

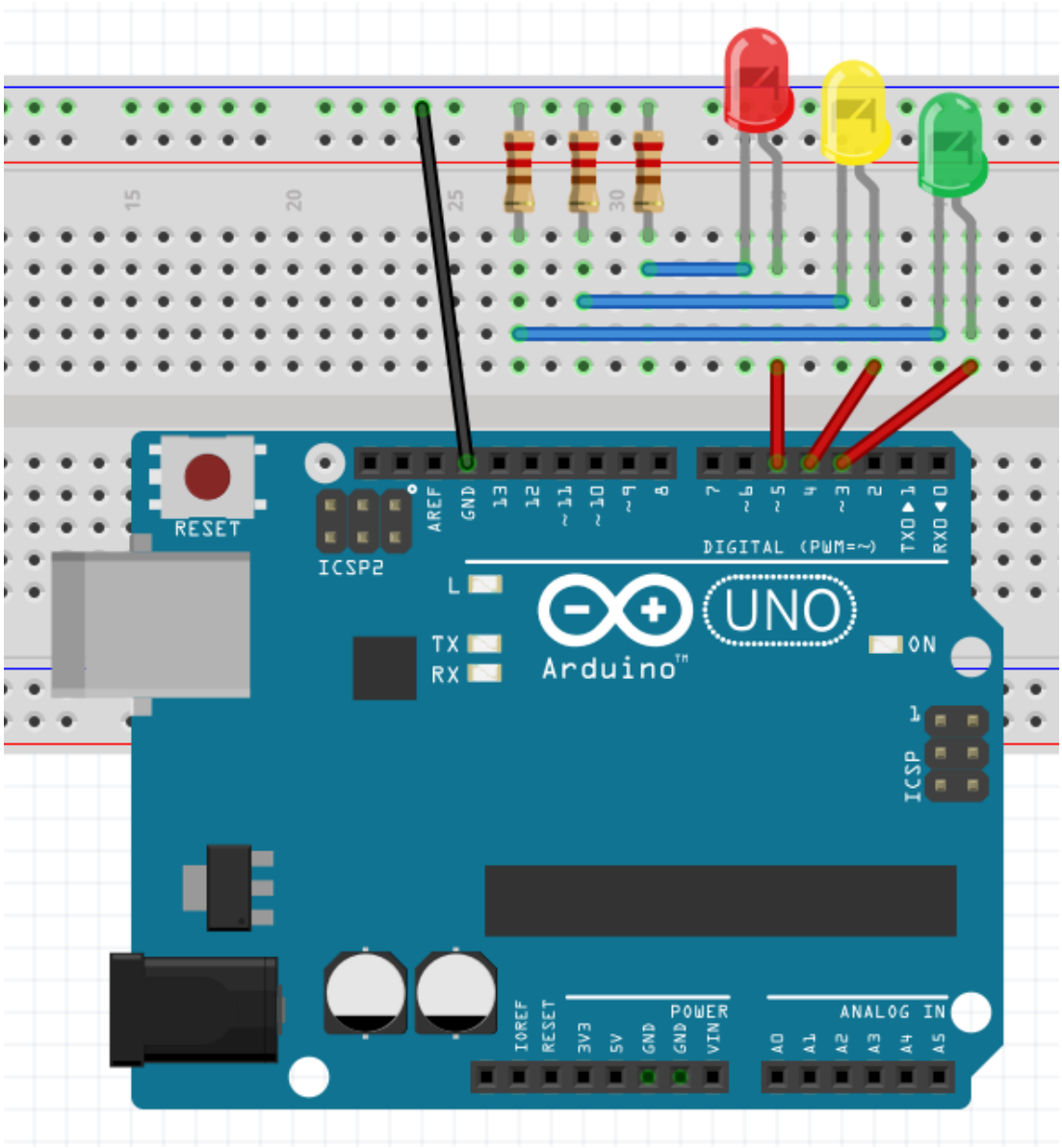
- Familiarizarse con el entorno de programación.
- Estructura de control for.
- Escribir funciones.

Esquema



Conexionado

La forma de conexión será la que se muestra en la figura:



Código fuente

reto02_a §

```
int tiempo = 200; //declara una variable como entero y de valor 200

void setup() { //comienza la configuracion
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
}

void loop() { //comienza el bucle principal del programa
  digitalWrite(3, HIGH);
  delay(tiempo);
  digitalWrite(3, LOW);
  delay(tiempo);
  digitalWrite(4, HIGH);
  delay(tiempo);
  digitalWrite(4, LOW);
  delay(tiempo);
  digitalWrite(5, HIGH);
  delay(tiempo);
  digitalWrite(5, LOW);
}
```

Código fuente 2

reto02_b §

```
int tiempo = 200;

void setup() {
  for (n = 3; n < 6; n++) {

    pinMode (n, OUTPUT);
  }


}

void loop() {

  for (n = 3; n < 6; n++) {
    digitalWrite (n, HIGH);
    delay (tiempo);
    digitalWrite (n, LOW);
    delay (tiempo);

  }

}
```

Código fuente 3

reto02_c §

```
int tiempo = 200;
int n;

void setup() { //comienza la configuracion
  for (n = 3; n < 6; n++) {
    pinMode (n, OUTPUT);
  }
}

void secuencia() {
  for (n = 3; n < 6; n++) {
    digitalWrite (n, HIGH);
    delay (tiempo);
    digitalWrite (n, LOW);
    delay (tiempo);
  }
}

void loop() {
  secuencia();
}
```

Reto 3: SOS con zumbador

Se trata de un zumbador que en código morse (pitidos largos/cortos) especifica una palabra, en nuestro caso SOS. Para el que no lo sepa, la S son tres señales acústicas de corta duración y la O tres señales acústica de larga duración.

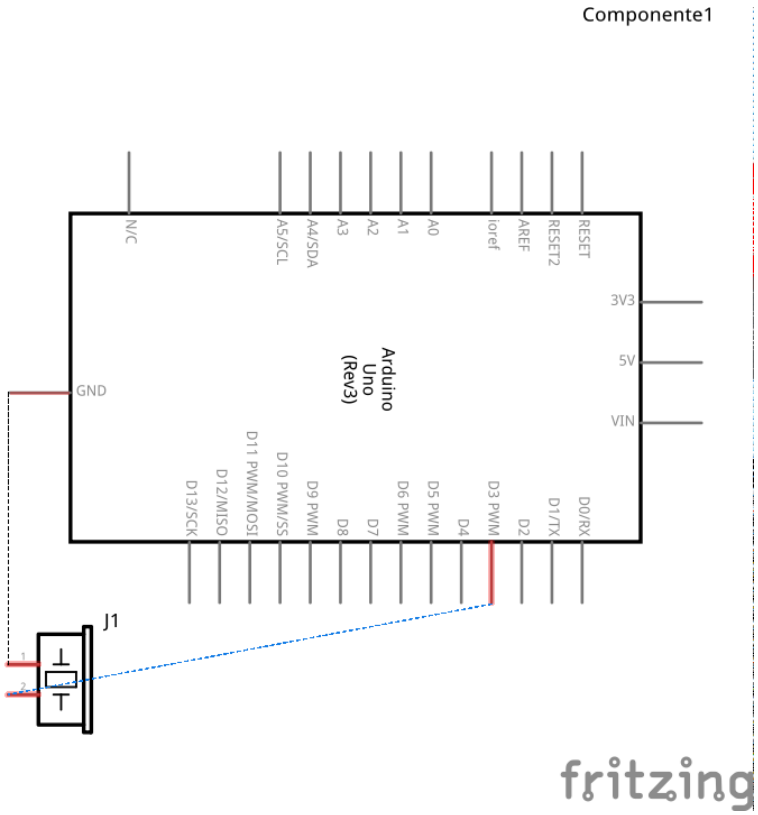
El zumbador debe estar conectado al pin 3, aunque para no ser ruidosos se puede hacer con el led asociado al pin 3 de edubasica (verde), los pitidos o destellos cortos tendrán una duración de 100 ms y los largos 300 ms. Entre letra y letra debe pasar un tiempo de 300 ms y entre SOS debe haber un tiempo de 1000 ms.

Nota: Debes usar variables para guardar los tiempos que vas a usar.

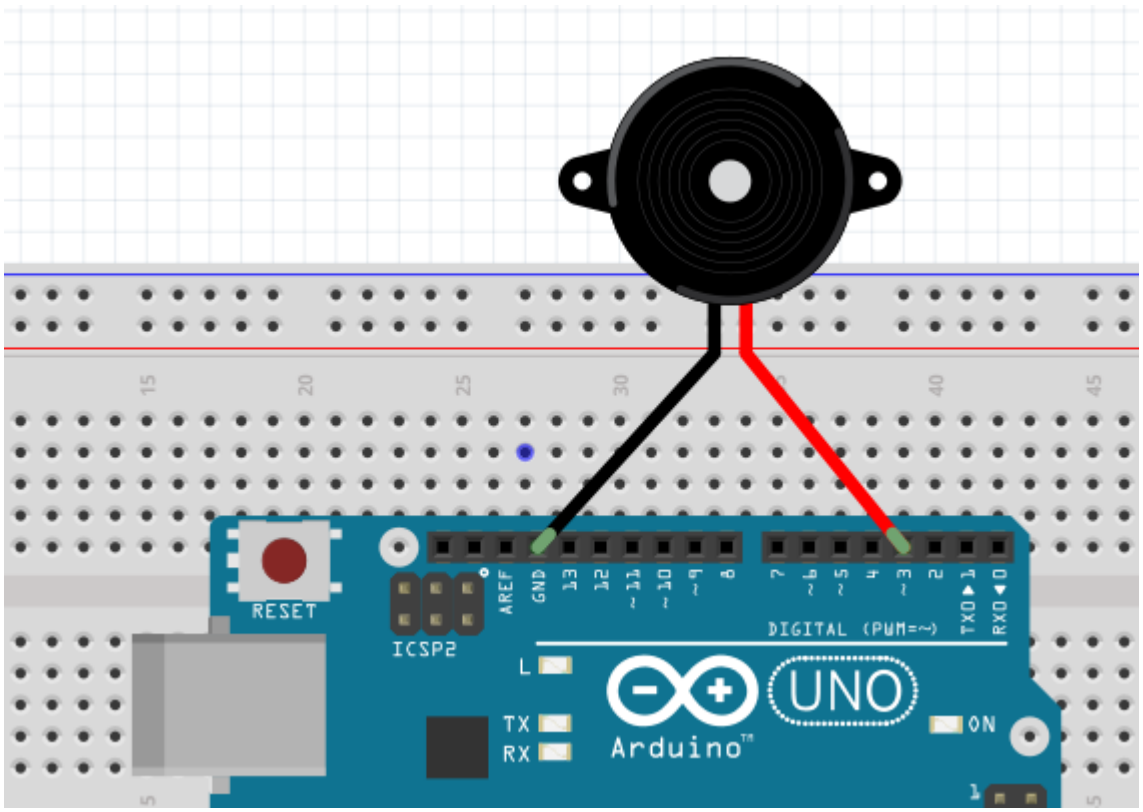
Objetivos:

- Repaso a blucle for.
- Repaso a funciones.
- Introducción del zumbador.
- Resolución y comprobación de reto.

Esquema



Conexionado



Código fuente

Solución 1:

reto04_a §

```
int corto = 100; //Declara la variable de argumento entero "corto" y la inicializa con el valor 100 (letra S)
int pausa = 300; //tiempo entre letra y letra
int largo = 300; //variable de argumento entero "largo" y la inicializa con el valor 300 (letra O)
int espera = 1000; //variable argumento entero "espera" y la inicializa con el valor 1000 (tiempo entre SOS - SOS)
int n = 0;
int zumb = 3; //PIN digital al que conectamos el zumbador

void setup() { //comienza la configuracion
  pinMode(zumb, OUTPUT);
}

void loop() {
  for (n = 0; n < 3; n++) { //Iteracion en la que la variable n comienza con el valor 0
    digitalWrite(zumb, HIGH); // y va aumentando en 1 en cada ciclo hasta que toma el valor 2,
    delay(corto); // con lo que las instrucciones comprendidas entre los corchetes
    digitalWrite(zumb, LOW); // se repiten 3 veces
    delay(corto);
  }

  delay(pausa); //Tiempo entre letras

  for (n = 0; n < 3; n++) { //Aqui esta la O
    digitalWrite(zumb, HIGH);
    delay(largo);
    digitalWrite(zumb, LOW);
    delay(largo);
  }

  delay(pausa);

  for (n = 0; n < 3; n++) {
    digitalWrite(zumb, HIGH);
    delay(corto);
    digitalWrite(zumb, LOW);
    delay(corto);
  }

  delay(espera); //Tiempo hasta repetir SOS de nuevo
}
```

Solución 2:

reto04_b

```
int tcorto = 100;
int tlargo = 300;
int pausa = 300;
int espera = 1000;
int n = 0;

void setup() { //comienza la configuracion
  pinMode(3, OUTPUT);
}
void s() { //comienza el bucle para la letra S
  for (n = 0; n < 3; n++) {
    digitalWrite (3, HIGH);
    delay (tcorto);
    digitalWrite (3, LOW);
    delay (tcorto);
  }
}
void o() { //comienza el bucle para la letra O
  for (n = 0; n < 3; n++) {
    digitalWrite (3, HIGH);
    delay (tlargo);
    digitalWrite (3, LOW);
    delay (tlargo);
  }
}
void loop() { //se ejecuta el bucle principal en el orden siguiente
  s();
  delay(pausa);
  o();
  delay(pausa);
  s();
  delay(espera);
}
```

Reto 4: Coche fantástico

Se trata de encender y apagar 3 leds secuencialmente. Los leds deben estar conectados a los pines 3, 4, y 5, que son los leds que provee edubasica.

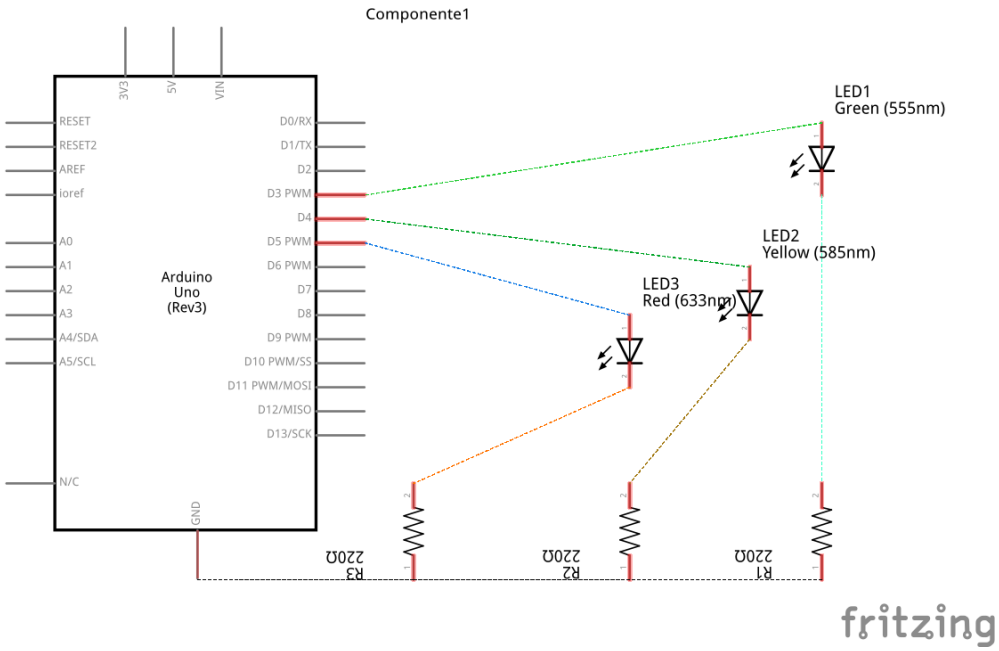
Se deben encender y apagar los leds desde el pin 3 al 5, con un tiempo de encendido y apagado de 50 ms, más tarde se deben encender y apagar los leds desde el pin 5 al 3, con un tiempo de encendido y apagado de 50 ms. La secuencia se debe repetir indefinidamente.

El efecto del programa es el de las luces delanteras de nuestro querido "Coche fantástico".

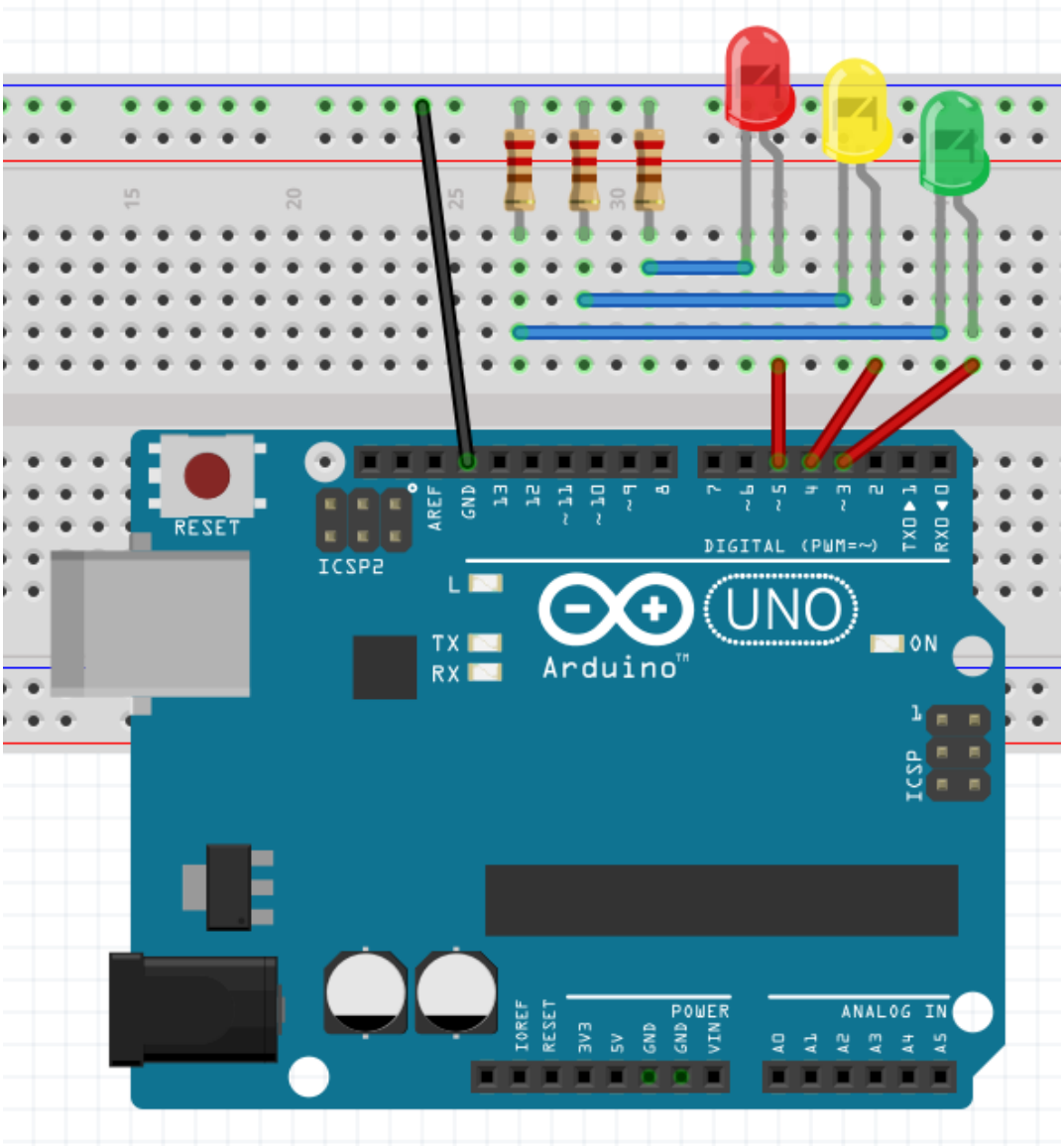
Objetivos:

- Familiarizarse con el entorno de programación.
- Repasar declaración de variables tipo lista de valores.
- Repasar órdenes de control de programa como: for.

Esquema



Conexionado



Código fuente

Solución 1:

reto05_b

```
int n = 0;
int tiempo = 50;

void setup() { //comienza la configuración
  for (n = 3; n < 6; n++) {
    pinMode(n, OUTPUT);
  }
}

void loop() {
  for (n = 3; n < 6; n++) {
    digitalWrite (n, HIGH);
    delay(tiempo);
    digitalWrite (n, LOW);
    delay(tiempo);
  }
  for (n = 5; n >= 3; n--) {
    digitalWrite (n, HIGH);
    delay(tiempo);
    digitalWrite (n, LOW);
    delay(tiempo);
  }
}
```

Solución 2 (con array)):

reto05_a

```
int leds[] = {3, 4, 5};
int n = 0;
int tiempo = 50;

void setup() { //comienza la configuración
  for (n = 0; n < 3; n++) {
    pinMode(leds[n], OUTPUT);
  }
}
void loop() {
  for (n = 0; n < 3; n++) {
    digitalWrite (leds[n], HIGH);
    delay(tiempo);
    digitalWrite (leds[n], LOW);
    delay(tiempo);
  }
  for (n = 2; n >= 0; n--) {
    digitalWrite (leds[n], HIGH);
    delay(tiempo);
    digitalWrite (leds[n], LOW);
    delay(tiempo);
  }
}
```

Solución 3 (mejorando el efecto visual):

reto05_c

```
int leds[] = {3, 4, 5};
int n = 0;
int tiempo = 30;

void setup() { //comienza la configuración
  for (n = 0; n < 3; n++) {
    pinMode(leds[n], OUTPUT);
  }
}

void loop() {
  for (n = 0; n < 3; n++) {
    digitalWrite (leds[n], HIGH);
    delay(tiempo);
    digitalWrite(leds[n + 1], HIGH);
    delay(tiempo);
    digitalWrite (leds[n], LOW);
    delay(tiempo * 2);
  }
  for (n = 2; n >= 0; n--) {
    digitalWrite (leds[n], HIGH);
    delay(tiempo);
    digitalWrite(leds[n - 1], HIGH);
    delay(tiempo);
    digitalWrite (leds[n], LOW);
    delay(tiempo * 2);
  }
}
```

Reto 5: Secuencia de leds con pulsador

Se trata de encender y apagar 3 leds secuencialmente al accionar un pulsador. El pulsador debe estar conectado al pin 2 (al que está conectado el pulsador de edubasica), y los leds a los pines 3, 4 y 5 (los de edubasica).

Se deben encender y posteriormente apagar los leds desde el pin 3 al 5, con un tiempo de duración de encendido y apagado de 200 milisegundos.

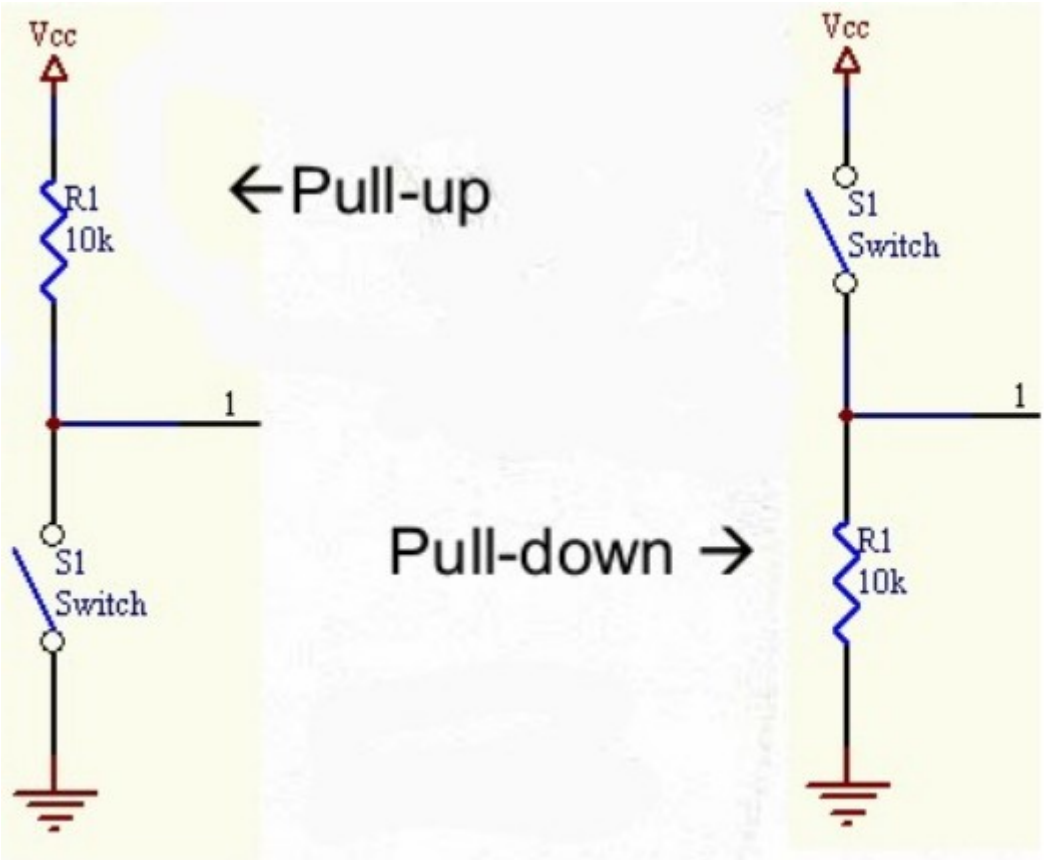
Nota: la secuencia principal del programa debe estar reproducida en una función a la que llamará el programa principal.

Objetivos:

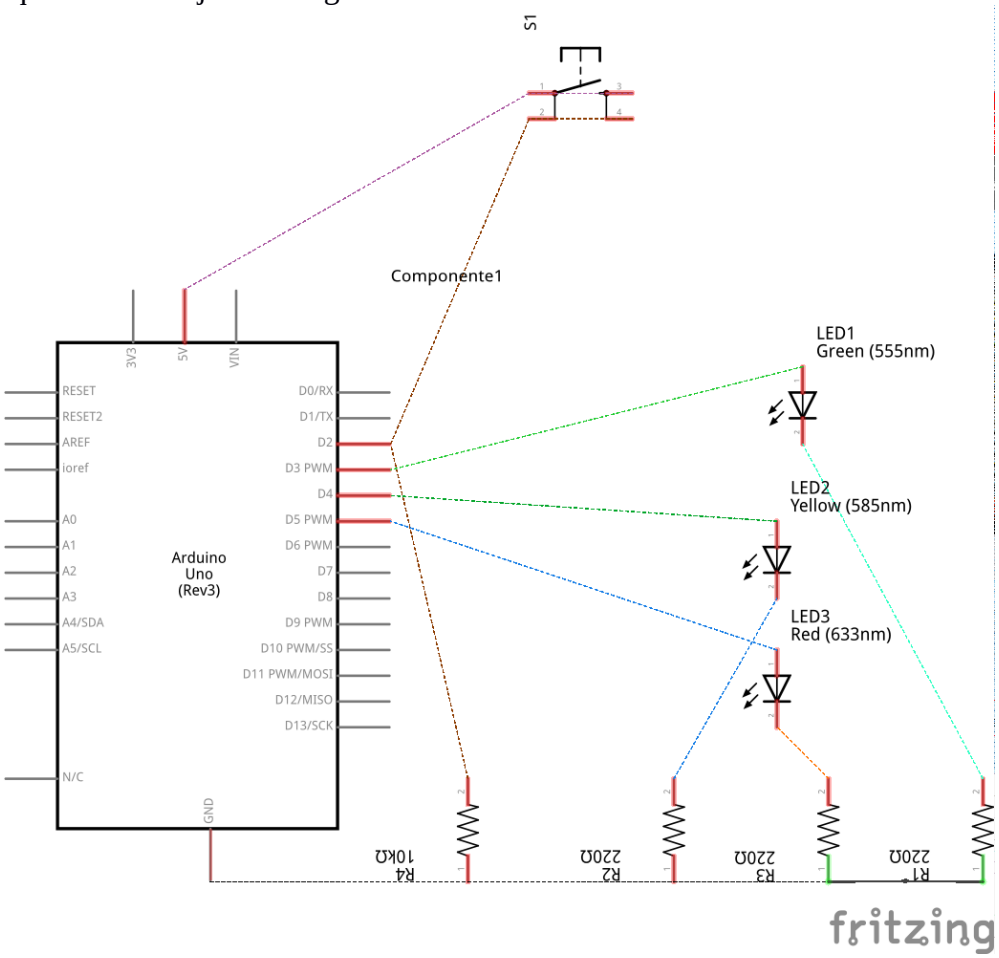
- Aprender a conectar una entrada digital a arduino (pulsador). Divisor de tensión
- Conexiones Pull-down y Pull-up.
- Conocer órdenes como: digitalRead.
- Conocer órdenes de control de programa como: If.
- Aprender a enviar y visualizar datos por puerto serie.

Esquemas

El esquema eléctrico del pulsador será:

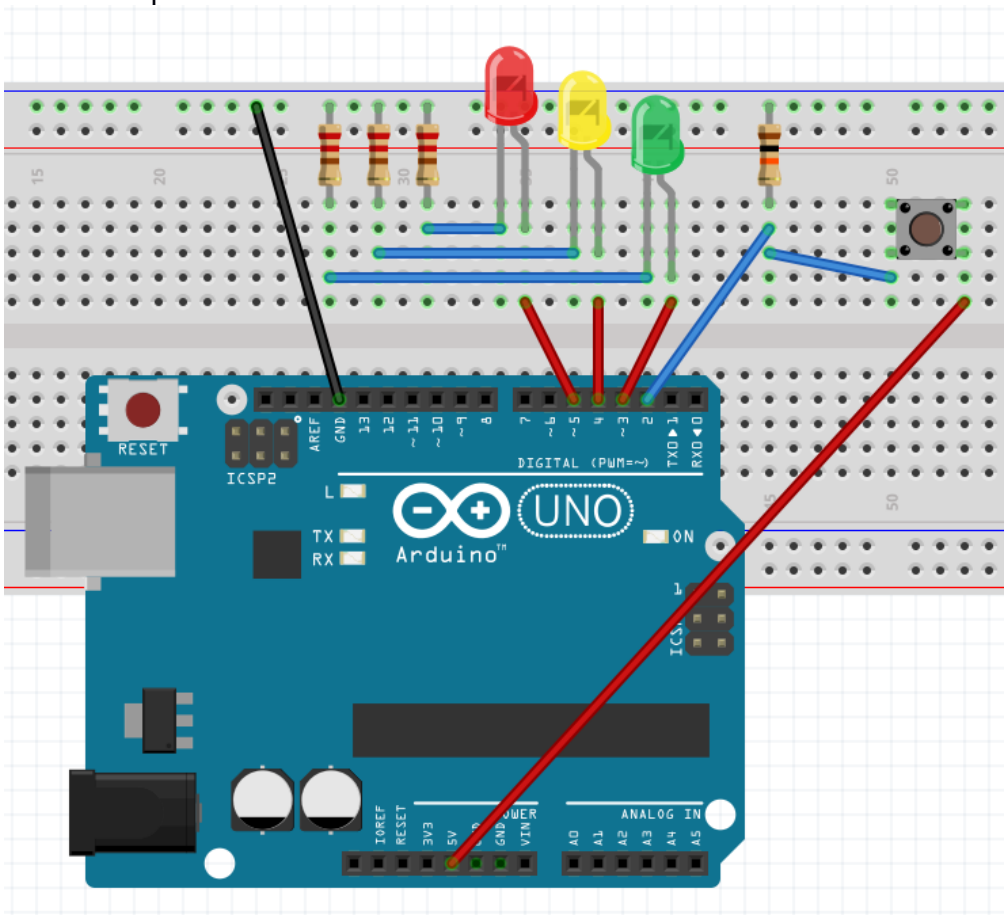


Siendo el esquema del conjunto el siguiente:

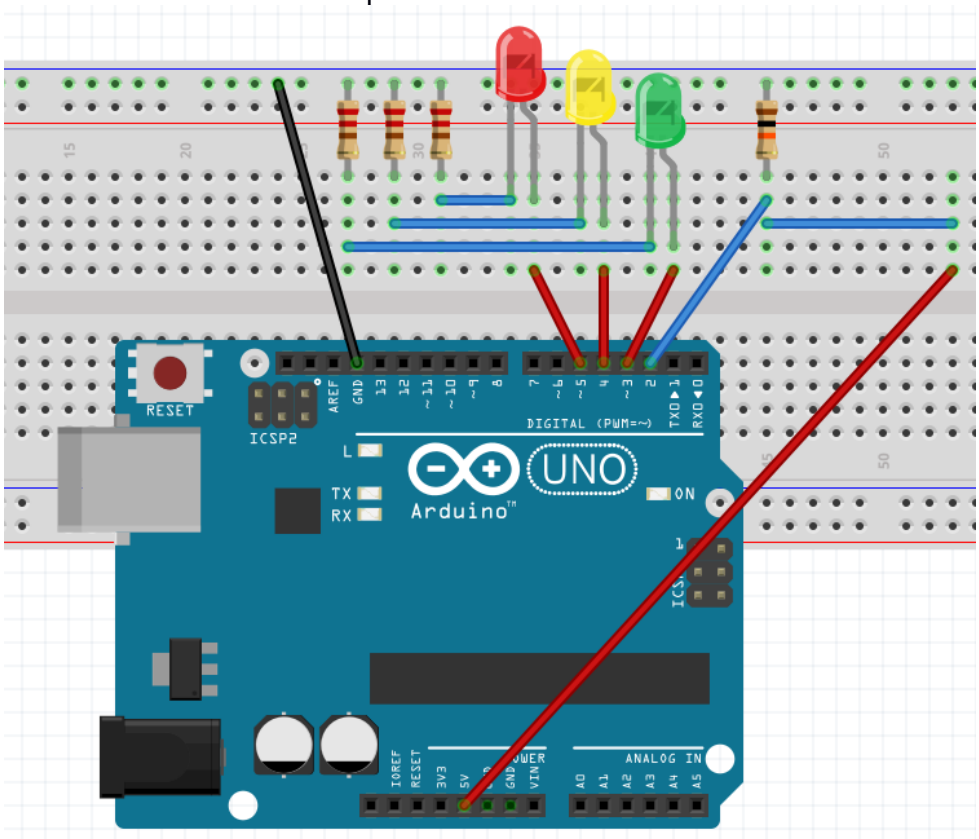


Conexionado

Conexionado real con pulsador:



Conexionado real con cables a modo de pulsador:



Código fuente

Solución 1

reto06_a §

```
int pulsador = 2;
int tiempo = 200;
int n;

void setup() {
  for (n = 3; n < 6; n++) {

    pinMode (n, OUTPUT);
  }
  pinMode (pulsador, INPUT);

}

void flash() {

  for (n = 3; n < 6; n++) {
    digitalWrite (n, HIGH);
    delay(tiempo);
    digitalWrite (n, LOW);
    delay(tiempo);

  }

}

void loop() {

  if (digitalRead (pulsador) == HIGH) {

    flash();
  }

}
```

Solución 2 (con variable tipo lista)

reto06_b

```
int cadenaleds[] = {3, 4, 5};
int pulsador = 2;
int tiempo = 200;
int n = 0;

void setup() {
  for (n = 0; n < 3; n++) {
    pinMode (cadenaleds[n], OUTPUT);
  }
  pinMode (pulsador, INPUT);
}

void flash() {
  for (n = 0; n < 3; n++) {
    digitalWrite (cadenaleds[n], HIGH);
    delay (tiempo);
    digitalWrite (cadenaleds[n], LOW);
    delay (tiempo);
  }
}

void loop() {
  if (digitalRead(pulsador) == HIGH) {
    flash ();
  }
}
```

Solución 3 (con visualización de datos por puerto serie)

reto06_c

```
int leds[] = {3, 4, 5};
int tiempo = 200;
int pulsador = 2;
int n = 0;
int valorpulsador = 0;

void setup() {
  for (n = 0; n < 3; n++) {
    pinMode(leds[n], OUTPUT);
  }
  pinMode(pulsador, INPUT);
  Serial.begin(9600);
}
void monitoriza() {
  Serial.print("El valor del pulsador es ...");
  Serial.println(valorpulsador);
  delay(1000);
}
void secuencia() {
  for (n = 0; n < 3; n++) {
    digitalWrite(leds[n], HIGH);
    delay(tiempo);
    digitalWrite(leds[n], LOW);
    delay(tiempo);
  }
}
void loop(){
  valorpulsador = digitalRead(pulsador);
  monitoriza();
  if (valorpulsador == 1) {
    secuencia();
  }
}
```

Reto 6: Ruleta de la fortuna

Se trata de tres leds que se van encendiendo y apagando formando una secuencia, el jugador debe dar al pulsador cuando el led intermedio se enciende, si acierta funciona un zumbador y la velocidad de la secuencia aumenta.

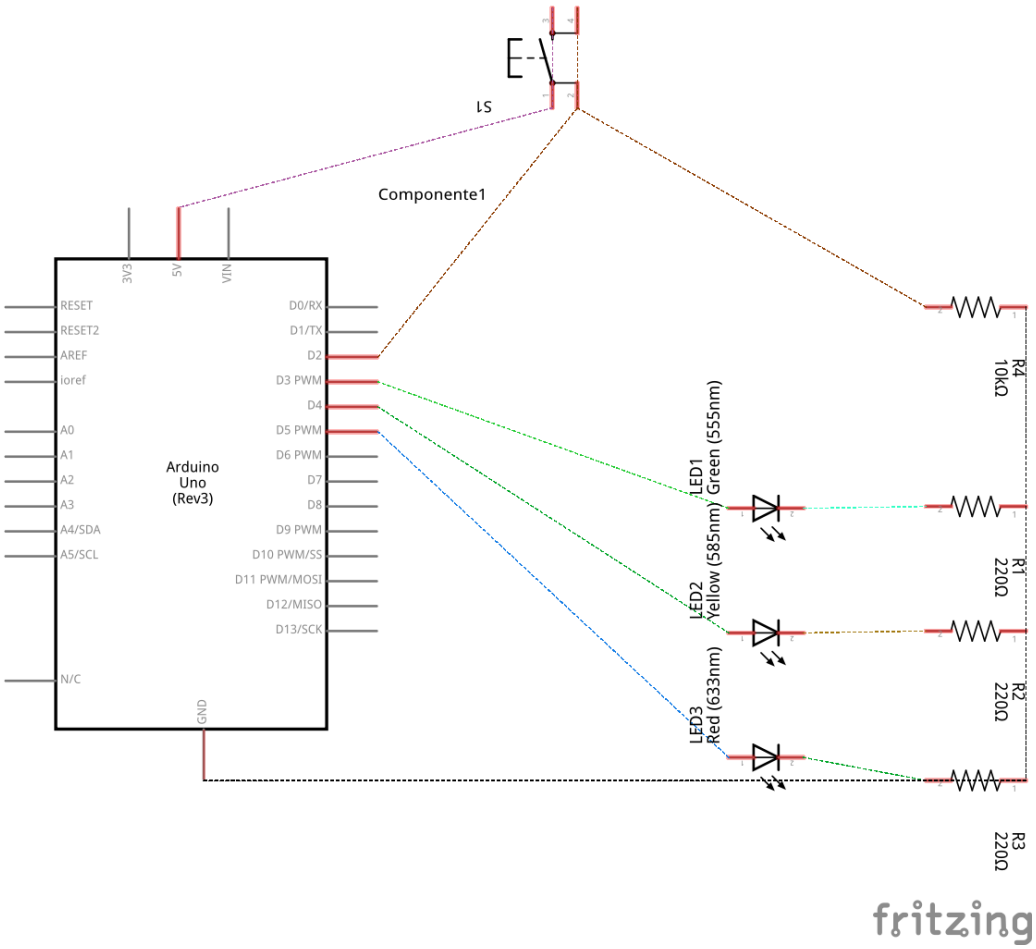
Los leds deben estar conectados de los pines 3 a 5 (los de edubasica), el zumbador al pin 7, el pulsador al pin 2 (el de edubasica).

El tiempo inicial entre encendido y encendido de leds debe ser 200 ms, si se acierta el tiempo disminuye en 20 ms, si el tiempo entre encendidos llegase a 10 ms, se devuelve el tiempo a 200 ms.

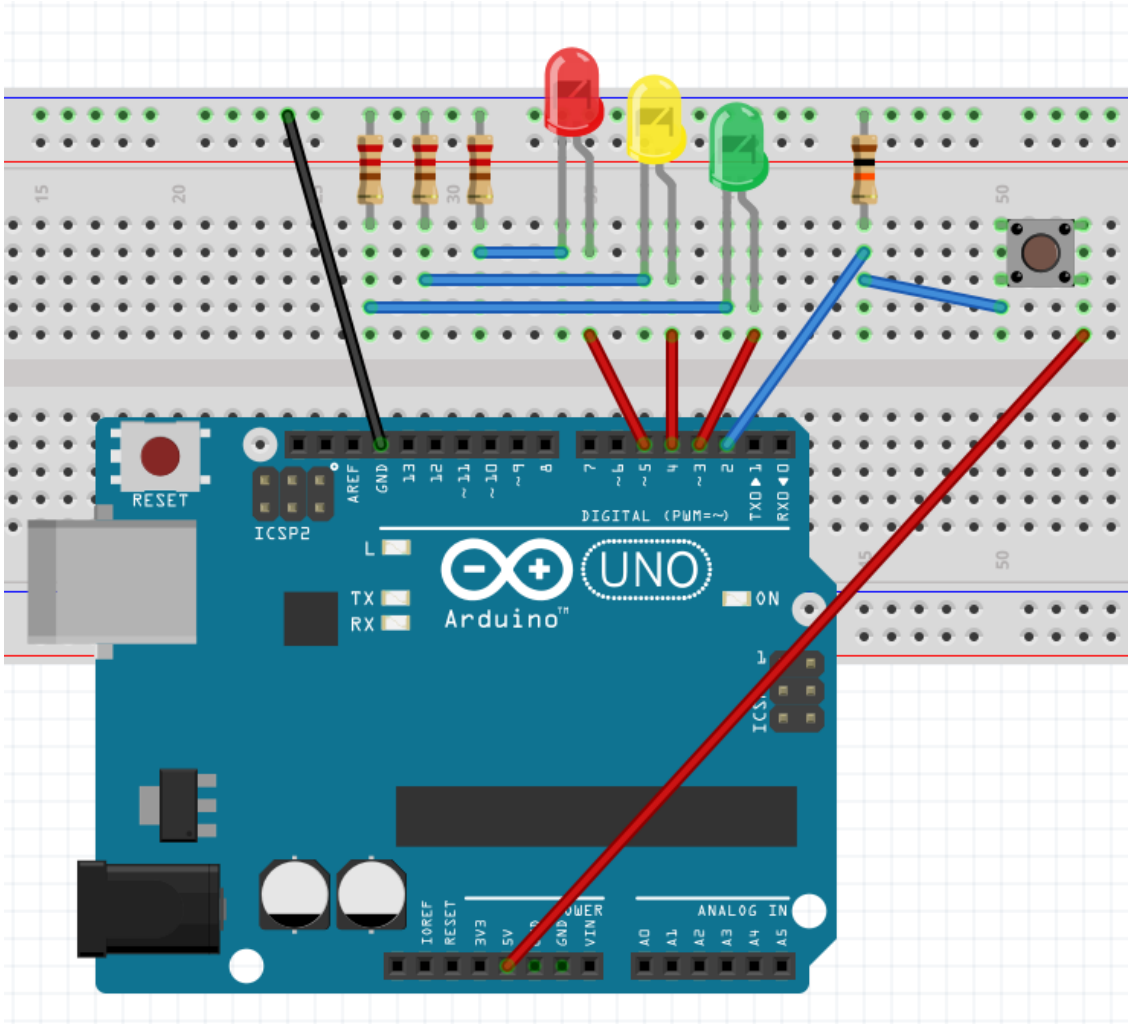
Objetivos:

- Conocer órdenes: if, &&, ||
- Repaso a uso de funciones
- Repaso a variables tipo lista
- Repaso a entradas digitales

Esquema



Conexionado



Código fuente

Código 1

reto_06_a

```
int n = 0;
int tiempo = 200;
int zumbador = 7;
int pulsador = 2;

void setup () {
  for (n = 3; n < 6; n++) {
    pinMode(n, OUTPUT);
  }
  pinMode(zumbador, OUTPUT);
  pinMode(pulsador, INPUT);
}
void compruebaacierto() {
  if (digitalRead(pulsador) == HIGH && n == 4) {
    digitalWrite(zumbador, HIGH);
    delay (1000);
    digitalWrite(zumbador, LOW);
    tiempo = tiempo - 20;
    if (tiempo < 10) {
      tiempo = 200;
    }
  }
}
void loop () {
  for (n = 3; n < 6; n++) {
    digitalWrite(n, HIGH);
    delay(tiempo);
    compruebaacierto();
    digitalWrite(n, LOW);
    delay(tiempo);
  }
}
```

Código 2 (con variable tipo lista)

reto_06_b

```
int leds[] = {3, 4, 5};
int n = 0;
int tiempo = 200;
int zumbador = 7;
int pulsador = 2;

void setup () {
  for (n = 0; n < 3; n++) {
    pinMode(leds[n], OUTPUT);
  }
  pinMode(zumbador, OUTPUT);
  pinMode(pulsador, INPUT);
}
void compruebaacierto() {
  if (digitalRead(pulsador) == HIGH && n == 1) {
    digitalWrite(zumbador, HIGH);
    delay (1000);
    digitalWrite(zumbador, LOW);
    tiempo = tiempo - 20;
    if (tiempo < 10) {
      tiempo = 200;
    }
  }
}
void loop () {
  for (n = 0; n < 3; n++) {
    digitalWrite(leds[n], HIGH);
    delay(tiempo);
    compruebaacierto();
    digitalWrite(leds[n], LOW);
    delay(tiempo);
  }
}
```

Código 3 (evita acertar con el pulsador permanentemente pulsado)

reto_06_c

```
int leds[] = {3, 4, 5};
int n = 0;
int tiempo = 200;
int zumbador = 7;
int pulsador = 2;

void setup () {
  for (n = 0; n < 3; n++) {
    pinMode(leds[n], OUTPUT);
  }
  pinMode(zumbador, OUTPUT);
  pinMode(pulsador, INPUT);
}
void compruebaacierto() {
  if (digitalRead(pulsador) == HIGH && n == 1) {
    digitalWrite(zumbador, HIGH);
    delay (1000);
    digitalWrite(zumbador, LOW);
    tiempo = tiempo - 20;
    if (tiempo < 10) {
      tiempo = 200;
    }
  }
  if (digitalRead(pulsador) == HIGH && n != 1) {
    for (n = 0; n < 3; n++) {
      digitalWrite(leds[n], HIGH);
    }
    delay(100);
    for (n = 0; n < 3; n++) {
      digitalWrite(leds[n], LOW);
    }
    delay (100);
    tiempo = 200;

  }
  void loop () {
    for (n = 0; n < 3; n++) {
      digitalWrite(leds[n], HIGH);
      delay(tiempo);
      compruebaacierto();
      digitalWrite(leds[n], LOW);
      delay(tiempo);
    }
  }
```

Reto 7: Detector de oscuridad

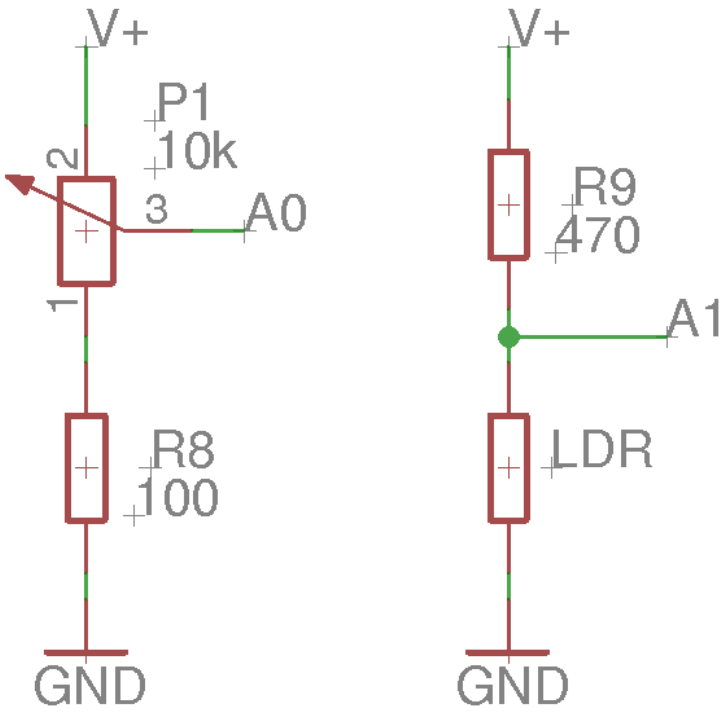
Se trata de un dispositivo que haga funcionar un led cuando la luminosidad baja de cierto umbral. Para ello conectaremos una ldr a la entrada analógica 1 (edubasica ya la trae conectada) y un led al pin 3 (uno de los edubasica). Cuando la luz llegue a cierto umbral de voltaje (entre 0 y 1023) que nosotros decidamos, se encenderá el diodo led. Además se deberá visionar el valor de voltaje en la entrada analógica (valor entre 0 y 1024) en una consola en el PC.

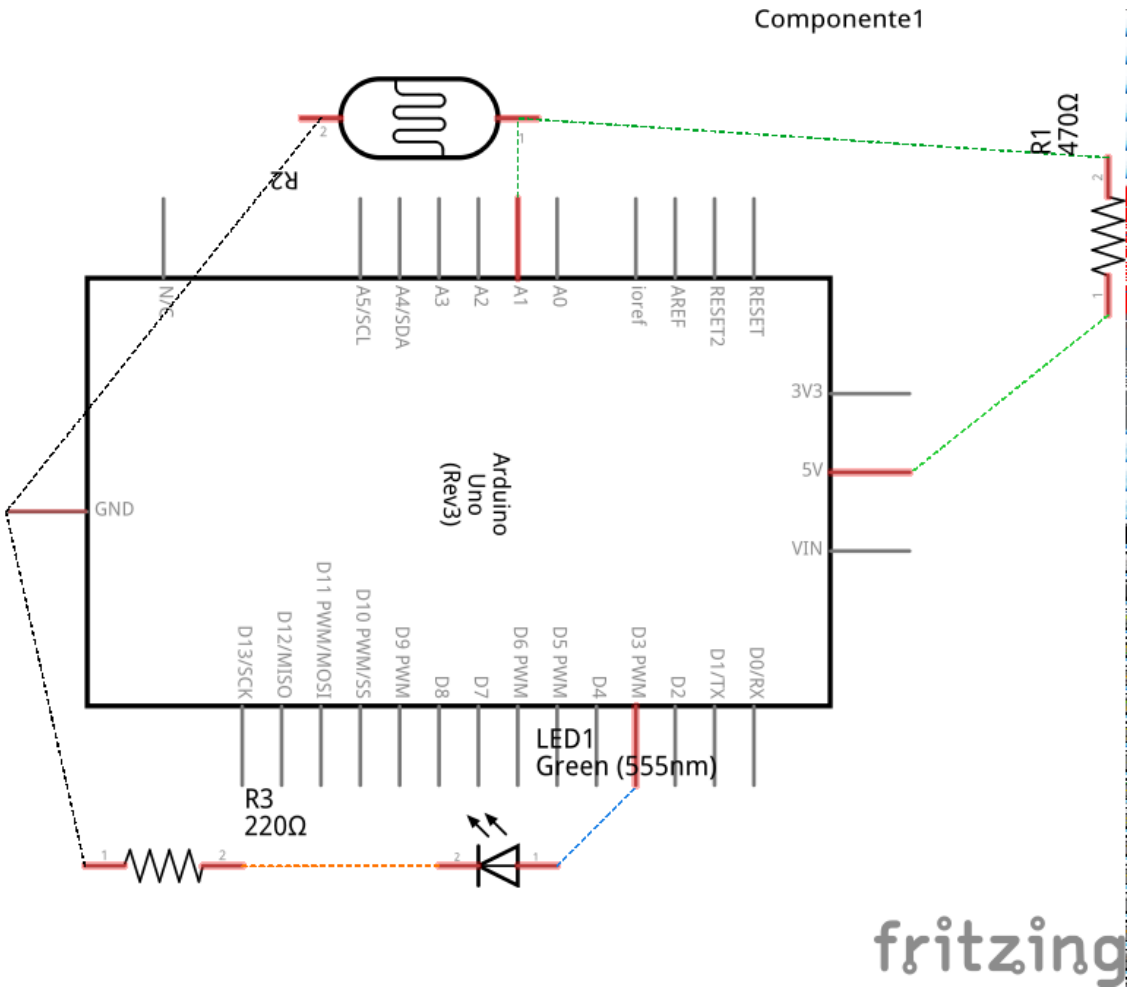
Objetivos:

- ¿Qué son las entradas analógicas?
- Conexión de entrada analógica a arduino (ldr). Repaso de divisor de tensión.
- Órdenes como: analogRead.
- Visualizar datos en consola de puerto serie, con órdenes como: Serial.begin,Serial.print.
- Órdenes de control de programa como: If else.

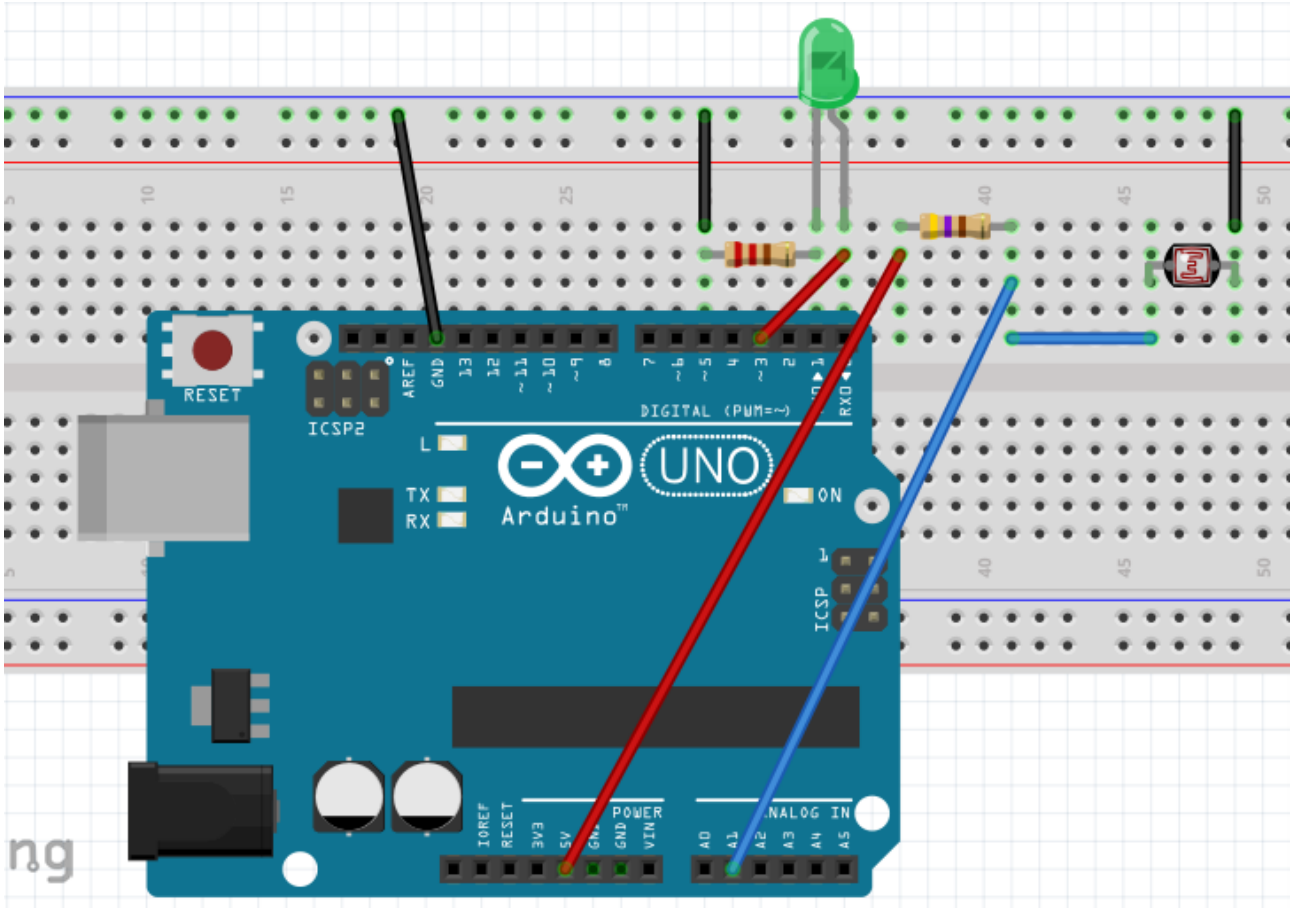
Esquemas

Esquema eléctrico.




Componente1
R2
R1
470Ω
A5/SCL
A4/SDA
A3
A2
A1
A0
ioref
AREF
RESET2
RESET
3V3
5V
VIN
GND
Arduino
Uno
(Rev3)
D13/SCK
D12/MISO
D11 PWM/MOSI
D10 PWM/SS
D9 PWM
D8
D7
D6 PWM
D5 PWM
D4
D3 PWM
D2
D1/TX
D0/RX
LED1
Green (555nm)
R3
220Ω
fritzing

Conexionado



Código fuente

reto08

```
int led = 3;
int ldr = 1;
int medida = 0;
int nivel = 900; //variable que guarda el límite de luz al que se activa el led

void setup() {
  pinMode(led, OUTPUT);
  Serial.begin(9600);
}
void monitoriza() { //procedimiento que envía al puerto serie, para ser leído en el monitor,
  Serial.print("La medida es ...");
  Serial.println(medida);
  delay(1000); //para evitar saturar el puerto serie
}
void loop() {
  medida = analogRead(ldr);
  monitoriza();
  if (medida > nivel) { //si la señal del sensor supera el nivel marcado:
    digitalWrite(led, HIGH); //se enciende un aviso luminoso
  }
  else { // si la señal está por debajo del nivel marcado
    digitalWrite(led, LOW);
  }
}
```

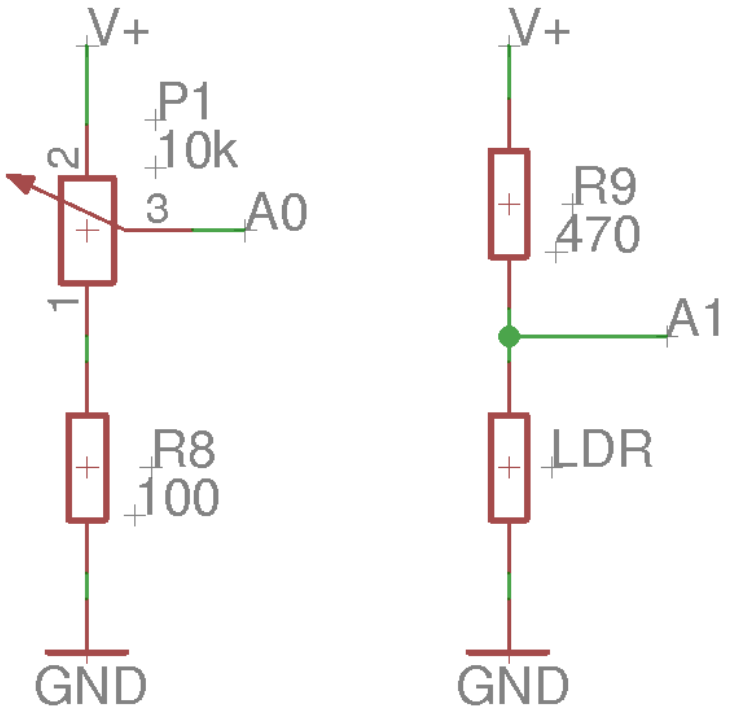
Reto 8: Potenciómetro que enciende un led rojo o uno verde en función de un valor de referencia

El reto consiste en iluminar un led rojo o un led verde en función de un valor de referencia que obtendremos de un divisor de tensión formado por un potenciómetro y una resistencia. El led rojo estará conectado al pin 5 y el verde al pin 3. El divisor de tensión de conectará a la entrada analógica A0.

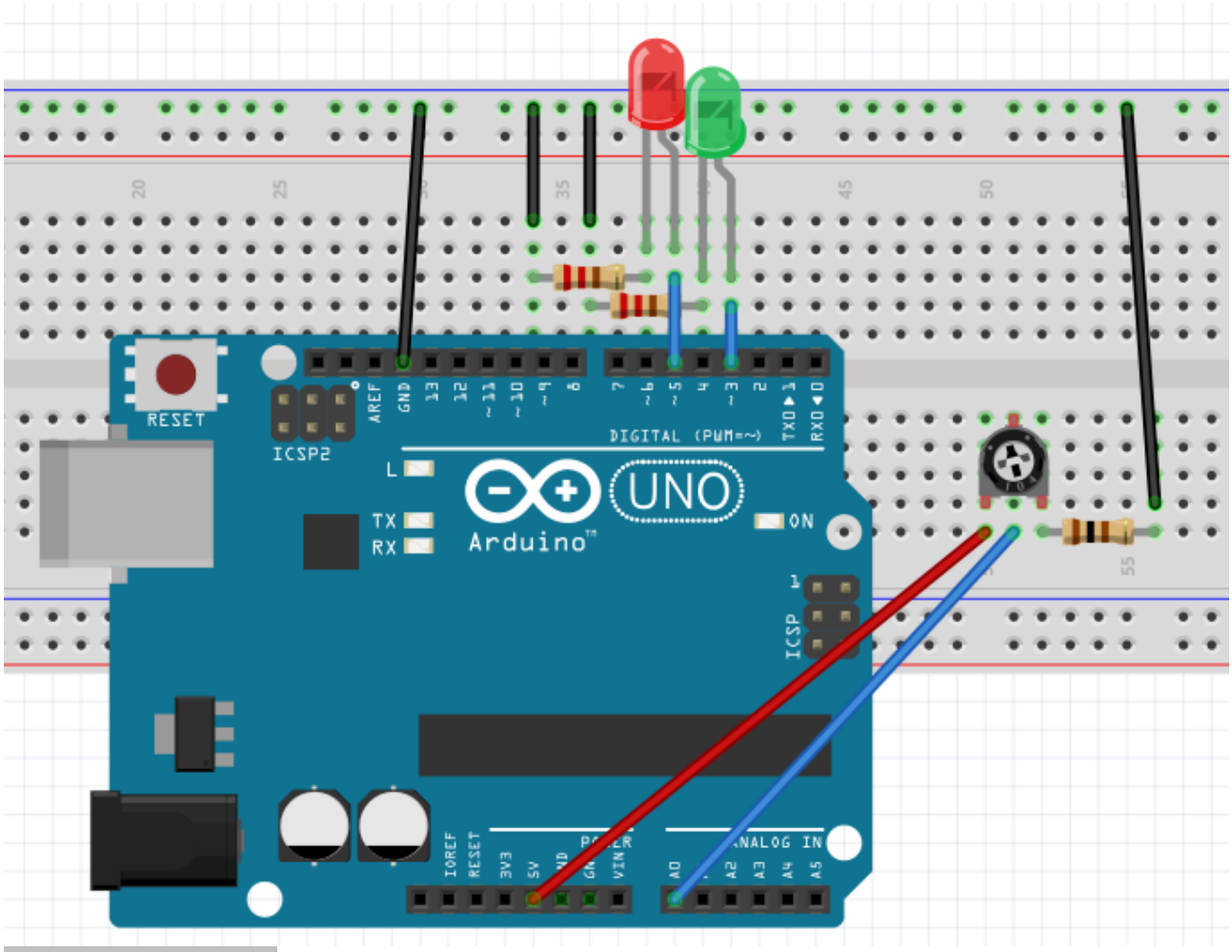
Objetivos:

- Repaso a entradas analógicas.
- Representación de valores utilizando la comunicación serie
- Repaso a órdenes de control If, else.

Esquema



Conexionado



Código fuente

reto_08

```
//Pines
int ledRojo = 5;
int ledVerde = 3;
int analogPin = 0;

int val = 0;
int referencia = 512;

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

void loop() {
  val = analogRead(analogPin);// lee el pin entrada
  Serial.println(val); //muestra el valor

  if (val <= referencia) {
    digitalWrite(ledVerde, HIGH);
    digitalWrite(ledRojo, LOW);
  }
  else {
    digitalWrite(ledRojo, HIGH);
    digitalWrite(ledVerde, LOW);
  }
}
```

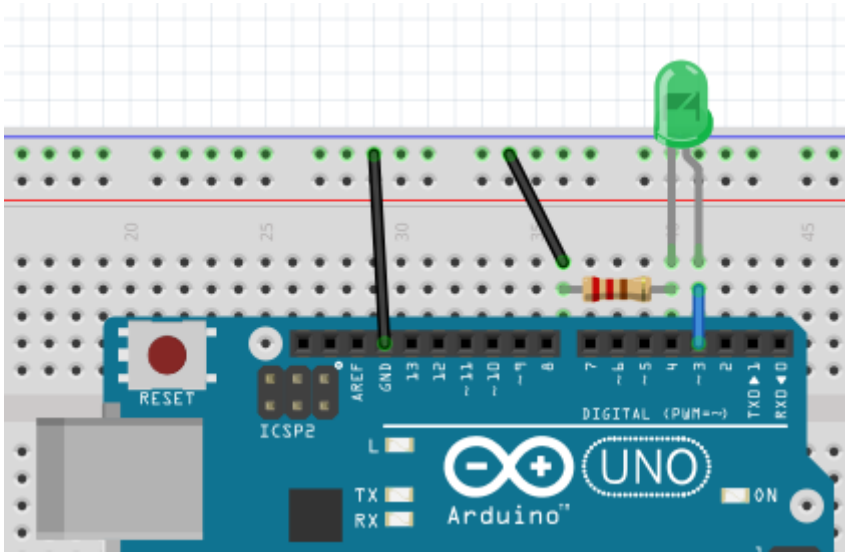
Reto 9: Aumentar y disminuir intensidad luminosa de led (fading)

Se trata aumentar y disminuir la luminosidad de un led usando la capacidad de ofrecer una tensión variable que da una salida analógica. Para ello se conecta un led al pin 3 (led verde en edubasica) y se provoca que su luminosidad pase de mínima a máxima, para luego ir de máxima a mínima. Los valores de salidas analógicas van del mínimo 0, al máximo 255.

Objetivos:

- Conexionado de salidas analógicas (power with module pwm).
- Conocer órdenes como analogWrite.

Conexionado



Código fuente

reto10

```
int luminosidad = 0; // variable para asignar la luminosidad al led
int led = 3; // pin del led
void setup() {
  // en el setup no hay que configurar nada
}
void loop(){
  for (luminosidad = 0 ; luminosidad <= 255; luminosidad = luminosidad + 3) { // fade in (from min to max)
    analogWrite(led, luminosidad); // ilumina el led con el valor asignado a luminosidad (entre 0 y 255)
    delay(30); // espera 30 ms para que se vea el efecto
  }
  for (luminosidad = 255; luminosidad >= 0; luminosidad = luminosidad - 3) { // fade out (from max to min)
    analogWrite(led, luminosidad);
    delay(30);
  }
}
```

Reto 10: Aumentar luminosidad de led con pulsador (fading)

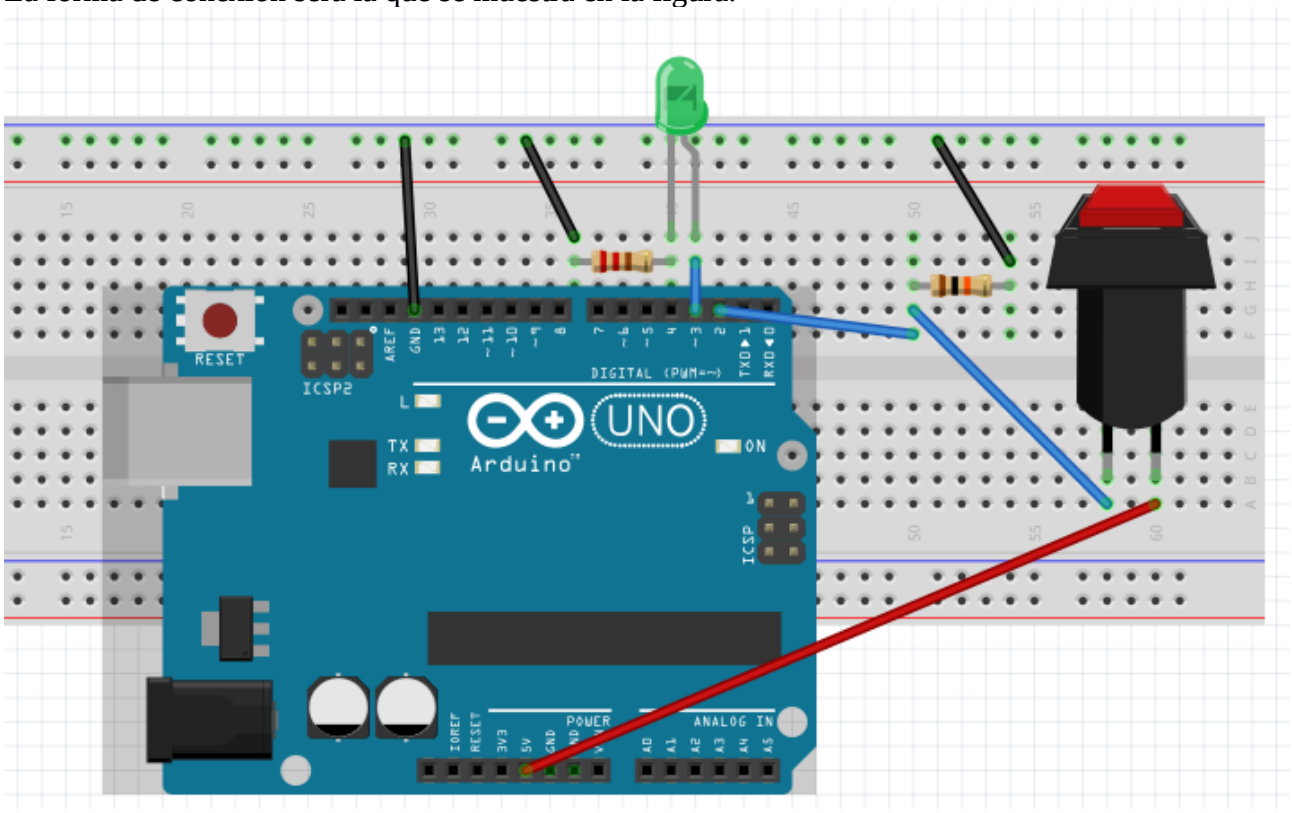
Se trata de aumentar la luminosidad de un diodo led conectado al pin 3 (verde de edubasica) a través de la activación de un pulsador conectado al pin 2 (el de edubasica). Si el pulsador se activa aumenta la luminosidad del led pudiendo llegar hasta su valor máximo (255), si el pulsador no se activa se mantendrá su luminosidad, si nos pasamos del valor máximo de luminosidad (255) pulsando nuevas veces, la luminosidad pasará a valor nulo (0).

Los objetivos:

- Repaso de conexionado de entradas digitales.
- Repaso de orden digitalRead.
- Repaso de conexionado de salidas analógicas.
- Repaso de orden analogWrite.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto11

```
int led = 3; // elegimos el pin del led
int pulsador = 2; // elegimos el pin del pulsador
int x = 0; // configuramos la variable para incrementar el valor de luminosidad

void setup() {
  pinMode(led, OUTPUT); // declaramos led como salida
  pinMode(pulsador, INPUT); // declaramos pulsador como entrada
}
void loop() {
  while (digitalRead(pulsador) == HIGH && x <= 255) { // chequea si el pulsador está pulsado y x es menor de 255
    analogWrite(led, x); // aumenta la luminosidad del led en función del tiempo de activación de pulsador
    delay(20);
    x = x + 3;
  }
  if (x > 255) {
    x = 0; // asigna el valor 0 a x
    analogWrite(led, 0); // apaga el led
  }
}
```

Reto 11: Control de led desde el ordenador

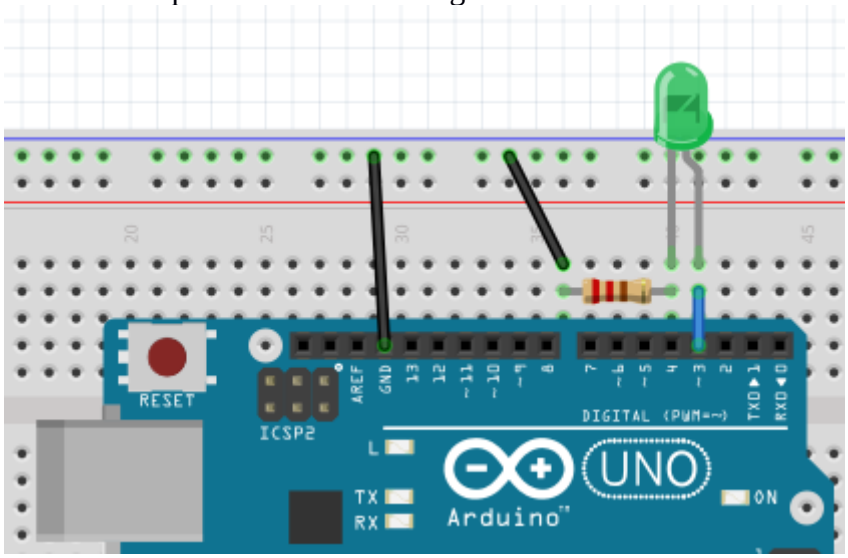
Se trata de diseñar un dispositivo que haga apagar o encender un led conectado al pin 3 desde el ordenador. Si pulsamos la tecla 1 se encenderá y si pulsamos la 0 se apagará.

Los objetivos:

- Repaso de visualización de datos en consola de puerto serie.
- Envío de datos por puerto serie a Arduino.
- Conocer el código ASCII.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

Solución 1

reto12

```
int led = 3;
int val = 0;


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

void loop() {

  if (Serial.available() > 0) {
    val = Serial.read();
  }
  Serial.println(val, DEC);

  if (val == '1') {
    digitalWrite(led, HIGH);
  }
  else {
    digitalWrite(led, LOW);
  }

}
```

Solución 2

reto12_b

```
int led = 3;
int val = 0;


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

void loop() {

  if (Serial.available() > 0) {
    val = Serial.read();
  }
  Serial.println(val, DEC);

  switch (val) {
    case 48:
      {
        digitalWrite(led, LOW);
        break;
      }
    case 49:
      {
        digitalWrite(led, HIGH);
        break;
      }
  }
}
```

Solución 3

reto12_c

```
int led = 3;
int val = 0;


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

void loop() {

  if (Serial.available() > 0) {
    val = Serial.read();
  }
  Serial.println(val, DEC);

  switch (val) {
    case '0':
      {
        digitalWrite(led, LOW);
        break;
      }
    case '1':
      {
        digitalWrite(led, HIGH);
        break;
      }
  }
}
```

Reto 12: Control de servomotor estándar

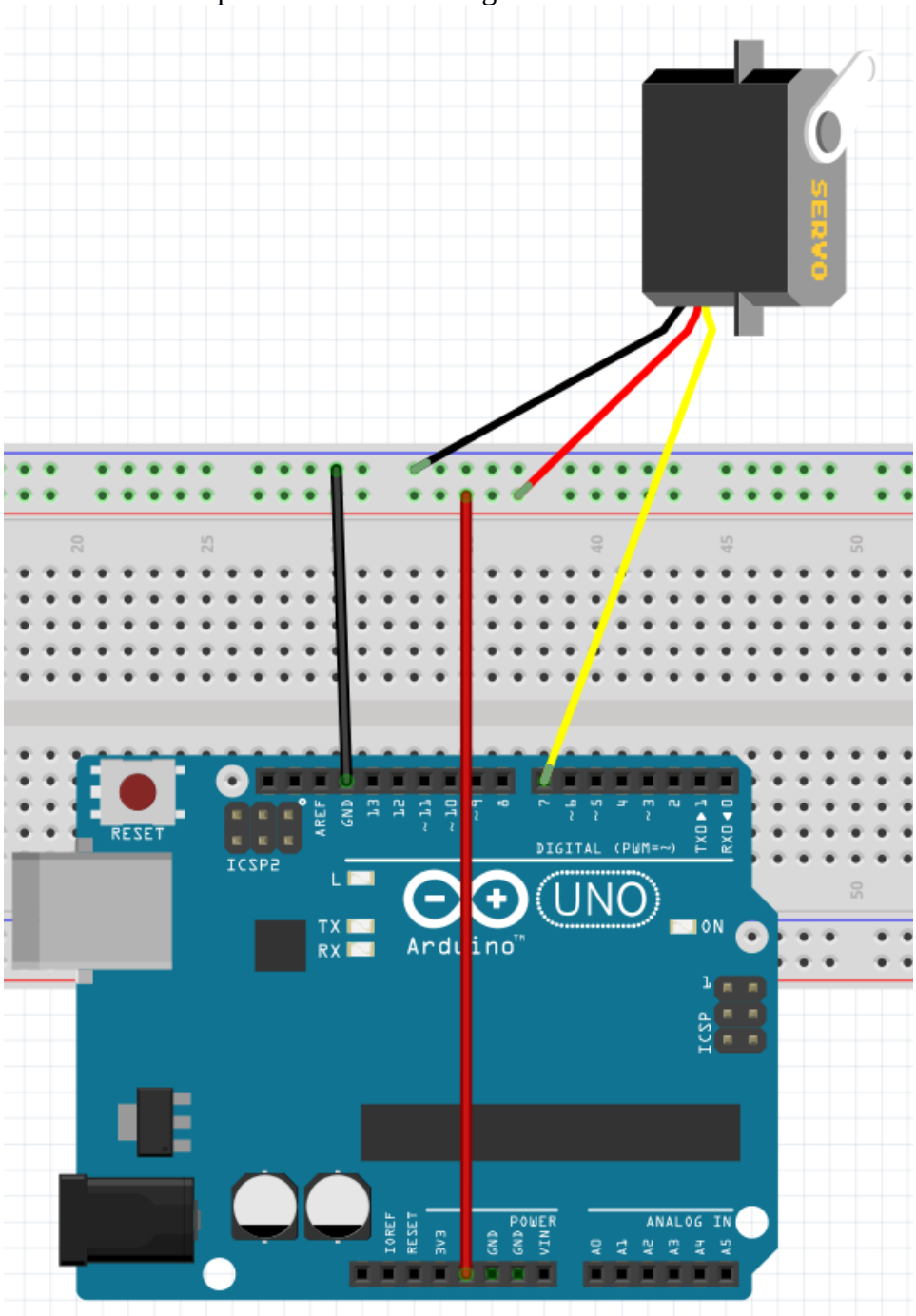
El reto consiste en conectar un servomotor estándar al pin 7 (está especialmente dispuesto en edubasica para ello), haciendo que el motor gire continuamente desde 0º a 180º y viceversa.

Objetivos:

- Conexionado de Servomotor.
- Cargar y usar librerías.
- Repaso de orden de control for.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto13

```
#include <Servo.h> //incluimos la libreria Servo

Servo miservo; //declaramos un objeto de tipo Servo llamado miservo

int val = 0;
int i = 0;

void setup() {
  miservo.attach(7);//conectamos el objeto miservo al pin digital 7
}

void loop() {
  for (i = 0; i < 180; i++) {
    miservo.write(i);
    delay(10);
  }
  for (i = 180; i > 0; i--) {
    miservo.write(i);
    delay(10);
  }


}
```

Reto 13: Control de servomotor estándar con potenciómetro

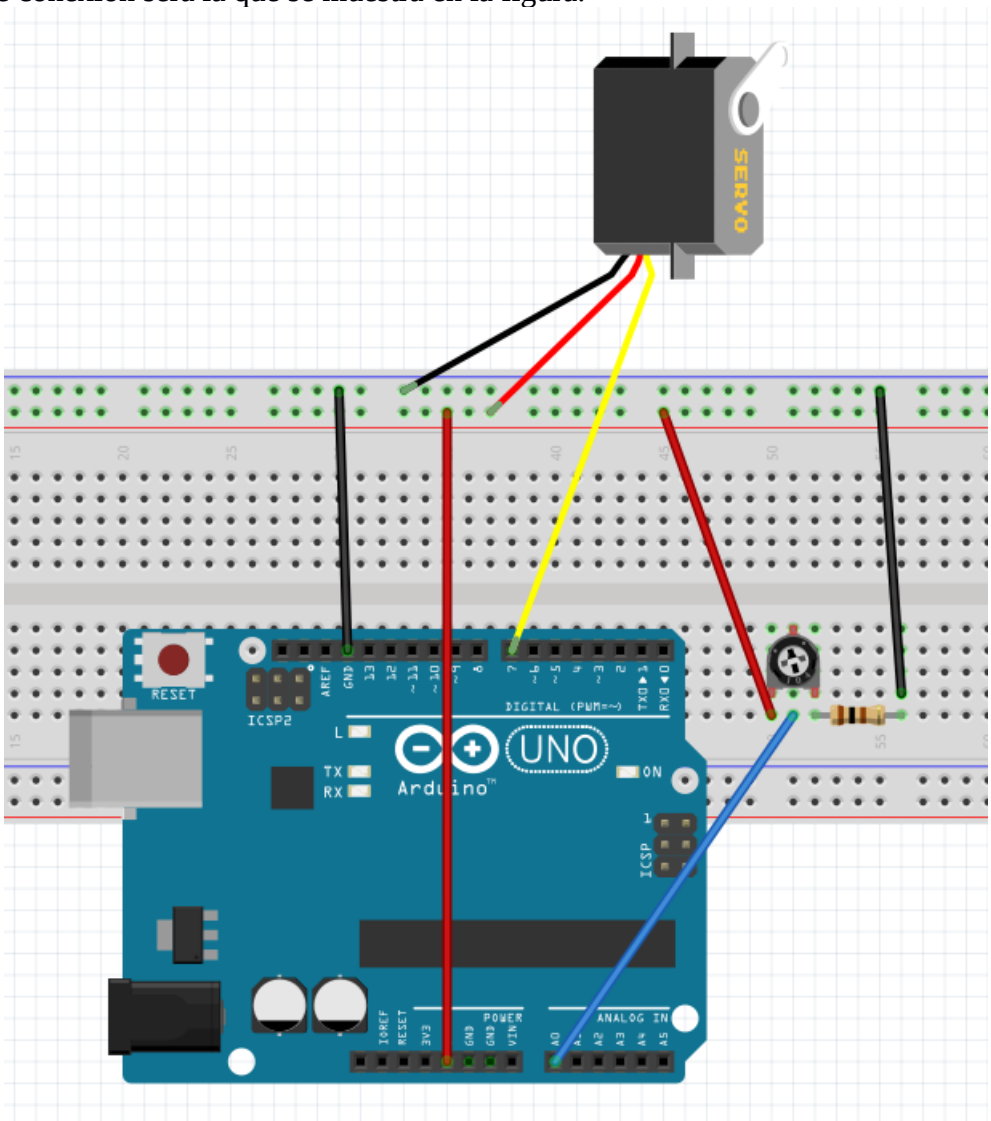
El reto consiste en conectar un servomotor estándar al pin 7 (está especialmente dispuesto en edubasica para ello) que estará controlado por un potenciómetro conectado al pin analógico 0. El sistema tiene que funcionar de tal manera que cuando el potenciómetro de un valor 0 de tensión en su entrada analógica el servo tiene que estar a 0º, y en la otra posición extrema del potenciómetro (marcando valor 1023) el servo tiene que estar a 180º.

Objetivos:

- Repaso de entradas analógicas
- Repaso de conexionado de servomotor.
- Repaso de carga y uso de librerías.
- Escalado de valores con la orden map.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto14

```
#include <Servo.h> //incluye la librería Servo

Servo miservo;//creamos un objeto tipo Servo llamado miservo
int valorPoten;
int valorServo;
int poten = 0;

void setup() {
  // put your setup code here, to run once:
  miservo.attach (7); //indicamos el pin digital dónde se conecta el servo
  Serial.begin (9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  valorPoten = analogRead (poten);
  valorServo = map(valorPoten, 0, 1023, 0, 179); // reescala el valor leido en el potenciómetro
  Serial.print ("el angulo en º es ...");
  Serial.println(valorServo);
  miservo.write (valorServo);

}
```

Reto 14: Servomotor de rotación continua

El reto consiste en conectar un servomotor de rotación continua al pin digital 7 y hacer que:

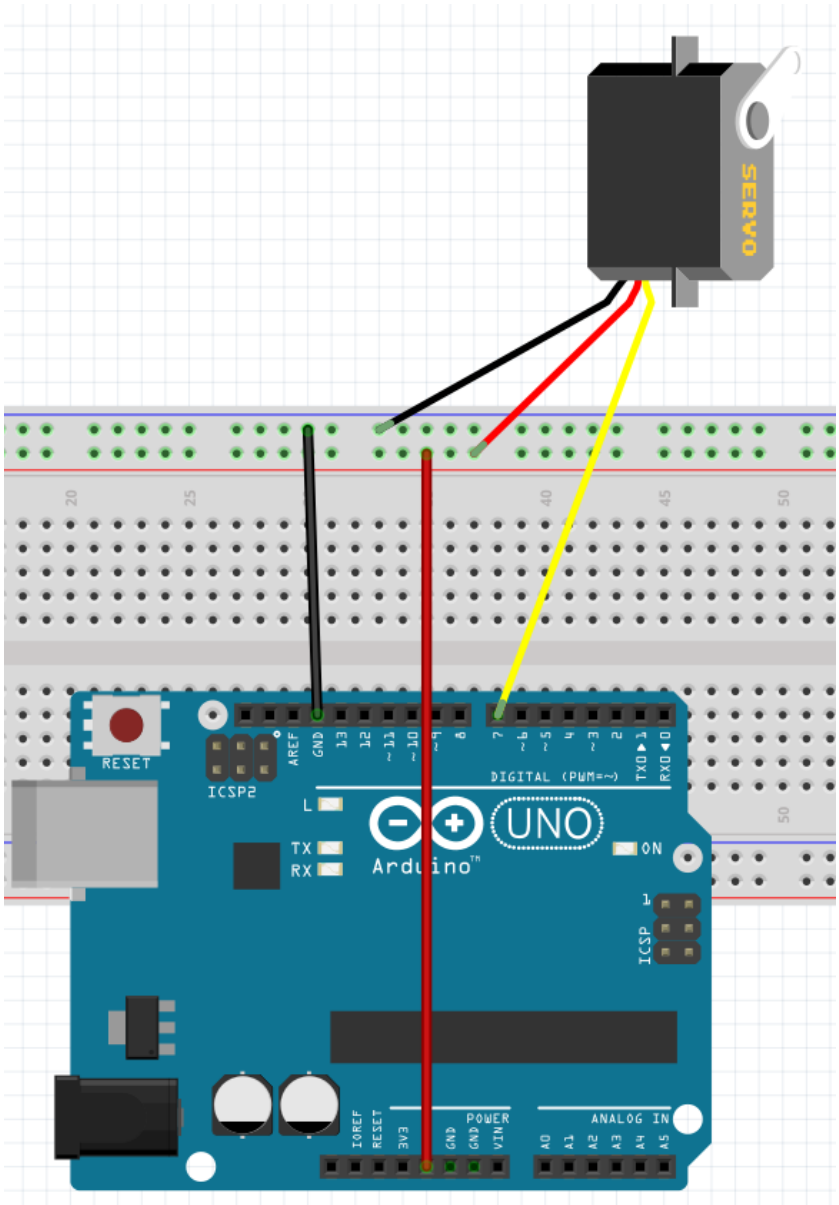
- Gire durante 3 segundos a máxima potencia en un sentido.
- Esté parado durante 2 segundos.
- Gire en sentido contrario al primero durante otros 3 segundos.

Objetivos:

- Conexionado de servomotor de rotación continua.
- Carga y uso de librerías en este tipo de servo.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto15

```
#include <Servo.h> //incluye la libreria Servo

Servo miservo;//declara objeto miservo de tipo Servo

void setup() {
  miservo.attach(7);//indica el pin de conexión de miservo
}

void loop() {

  miservo.write(0);//gira en un sentido a velocidad máxima
  delay(3000);
  miservo.write(90);//para
  delay(2000);
  miservo.write(180);//gira en el otro sentido a velocidad máxima
  delay(3000);


}
```

Reto 15: Comunicación de ordenador con arduino. Encender led con una tecla.

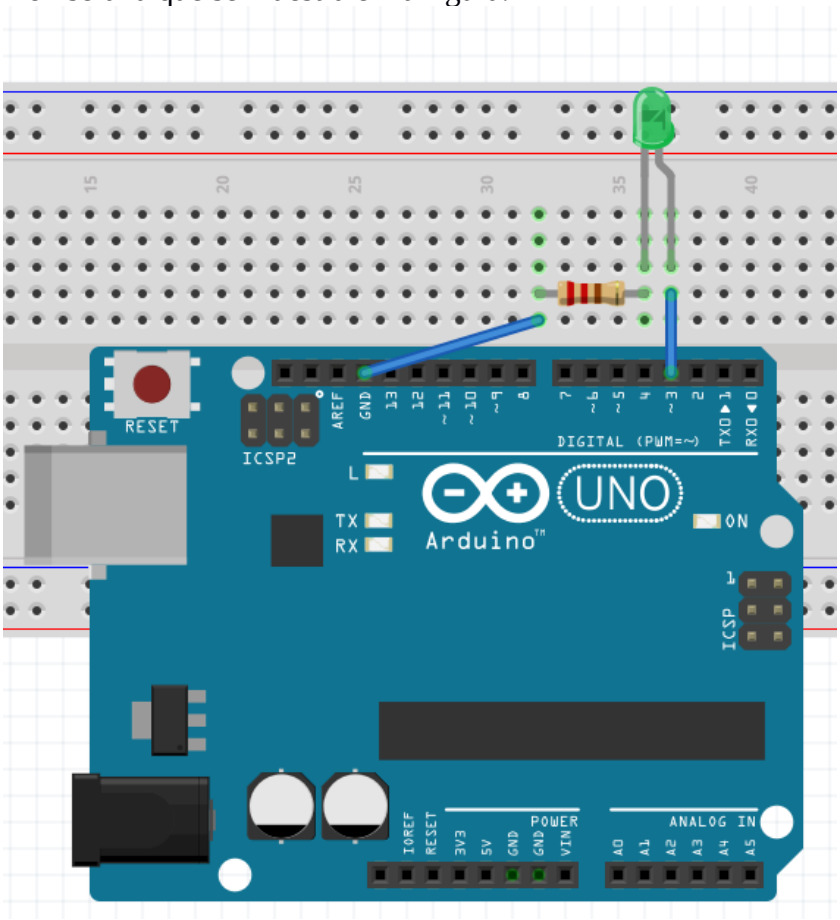
El reto consiste en encender o apagar led a través de una tecla del ordenador estableciendo una comunicación serie en el sentido ordenador Arduino.

Objetivos:

- Control serie sentido Ordenador Arduino.
- Utilizar variables booleanas.

Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto15

```
int led = 11;
int dato;
bool encendido = false;

void setup() {
  // put your setup code here, to run once:
  pinMode (led, OUTPUT);
  Serial.begin (9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (Serial.available() > 0) {
    dato = Serial.read();

    if (dato == 'a' || dato == 'A') {
      encendido = !encendido;

      if (encendido == true) {

        digitalWrite (led, HIGH);

      }

      else {

        digitalWrite (led, LOW);
      }
    }
  }
}
```

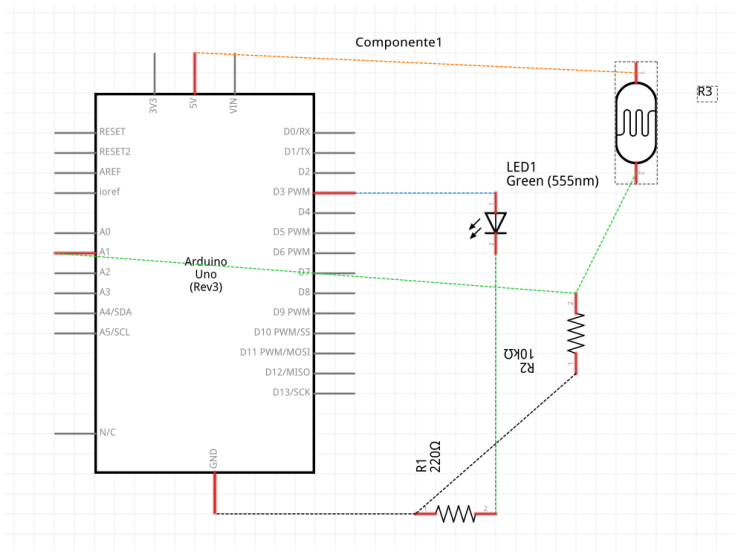
Reto 16: Variación de luz de led (PWM) con ldr.

El reto consiste en encender en variar la luminosidad de un led conectado a una salida PWM, de tal forma que a menos luz exterior más luminosidad del led y viceversa.

Objetivos:

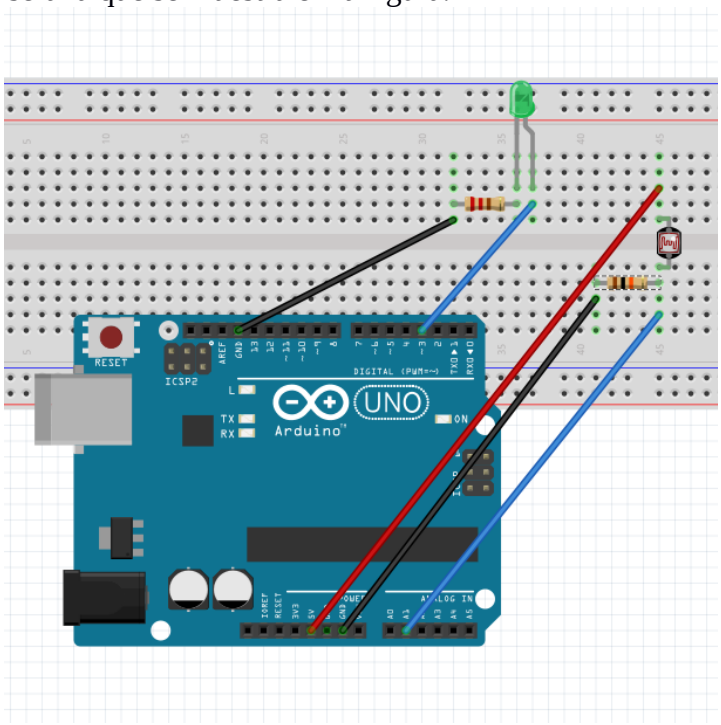
- Repasar conexionado divisor de tensión, PULL DOWN en este caso.
- Conocer o repasar la orden map.
- Repasar salidas PWM.

Esquema



Conexionado

La forma de conexión será la que se muestra en la figura.



Código fuente

reto16

```
int luz;
int luzled;
int led = 11;
int ldr = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  luz = analogRead (ldr);
  Serial.print("la luz es ...");
  Serial.println(luz);
  luzled = map (luz, 150, 913, 255, 0);
  analogWrite (led, luzled);
}
```

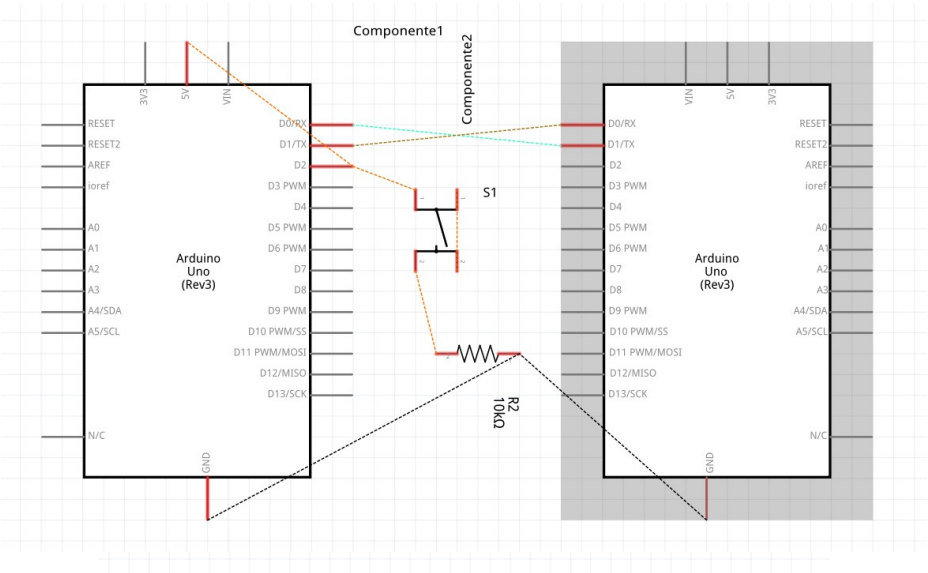
Reto 17: Pulsador de un arduino controla led de otro arduino.

El reto consiste en comunicar dos arduinos vía puerto serie, de tal manera que en el arduino maestro cuando se pulsa un pulsador (D2) se enciende led del arduino esclavo (D13).

Objetivos:

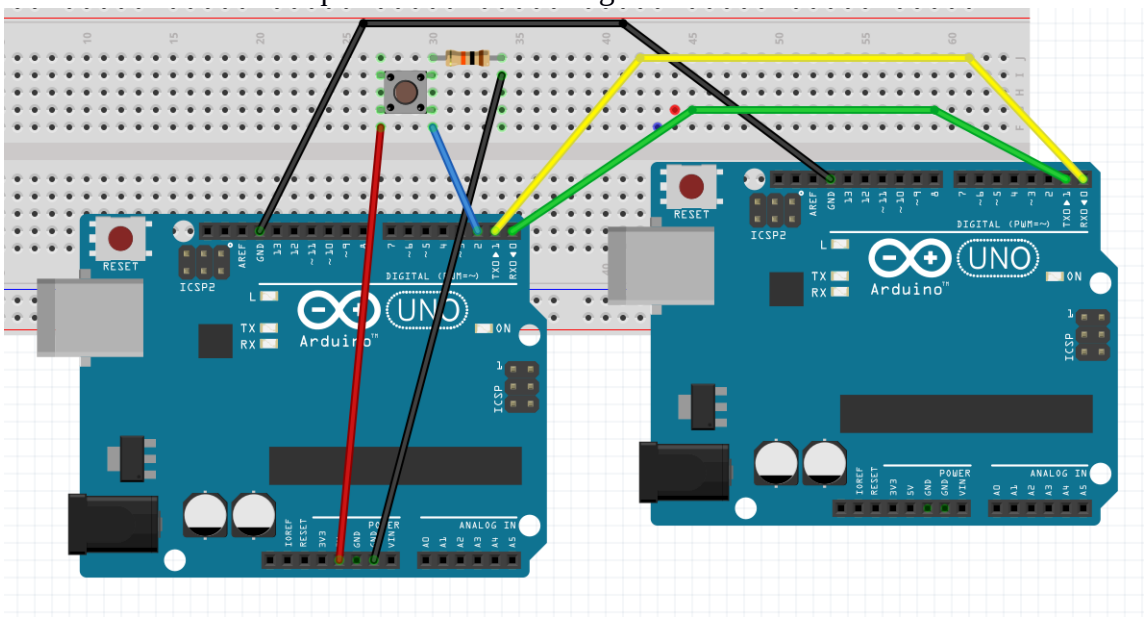
- Repasar conexionado divisor de tensión de pulsador, PULL DOWN en este caso.
- Conexionado para comunicación de arduino.
- Conocer órdenes para enviar y recibir datos con arduino. Serial.read(), Serial.write() y Serial.available().

Esquema



Conexionado arduino maestro

La forma de conexión será la que se muestra en la figura.



Código fuente arduino maestro

reto17a

```
int pin = 2;

void setup() {
  // put your setup code here, to run once:
  Serial.begin (9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (digitalRead (pin)) {

    Serial.write ('1');
  }

  else {
    Serial.write ('0');
  }

}
```

Código fuente arduino esclavo

reto17b

```
int dato;
void setup() {
  // put your setup code here, to run once:
  Serial.begin (9600);
  pinMode (13, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (Serial.available() > 0) {

    dato = Serial.read();

  }

  if (dato == '1') {

    digitalWrite (13, HIGH);
  }

  else {
    digitalWrite (13, LOW);
  }

}
```

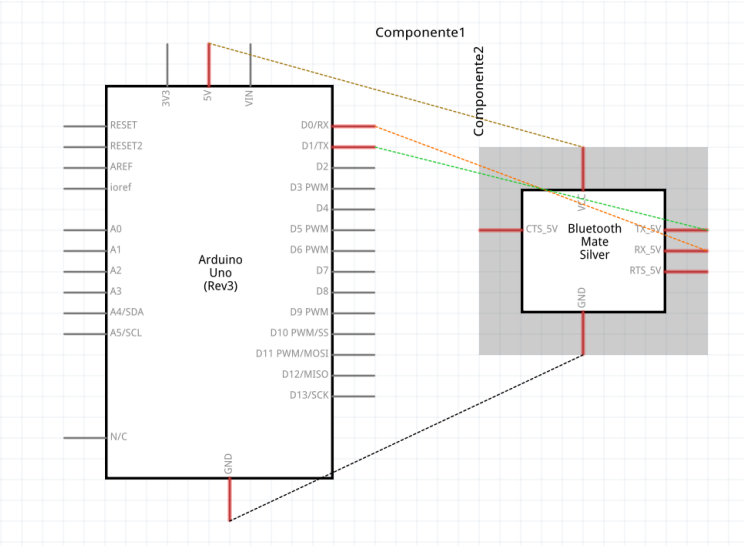
Reto 18: Móvil controlando led de arduino. Bluetooth

El reto consiste en comunicar un teléfono móvil con arduino mediante bluetooth, para ello conectaremos un dispositivo bluetooth a arduino que actuará como receptor y el móvil emitirá datos. Se establecerá una comunicación serie entre ellos, al arduino en función del dato recibido del móvil encenderá o apagará un led (D13).

Objetivos:

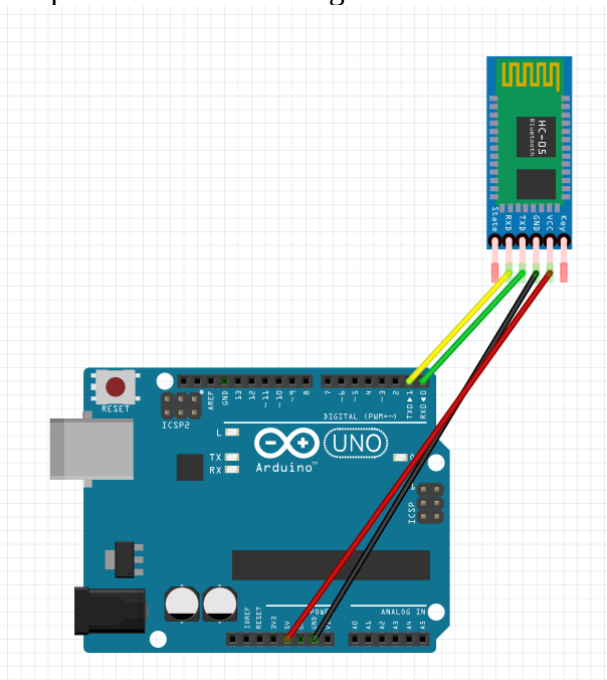
- Conexionado bluetooth para comunicación de móvil con arduino.
- Conocer o repasar órdenes para enviar y recibir datos con arduino. Serial.read() y Serial.available().

Esquema



Conexionado arduino maestro

La forma de conexión será la que se muestra en la figura.



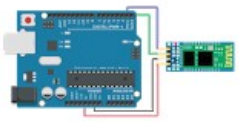
Código fuente

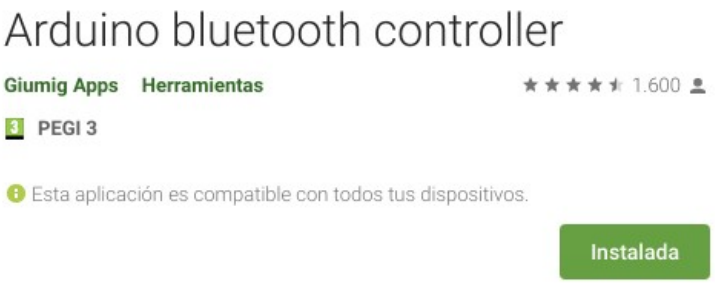
reto18

```
int dato;
void setup() {
  // put your setup code here, to run once:
  Serial.begin (9600);
  pinMode (13, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (Serial.available() > 0) {

    dato = Serial.read();

  }

  if (dato == '1') {

    digitalWrite (13, HIGH);
  }

  else {
    digitalWrite (13, LOW);
  }

}
```

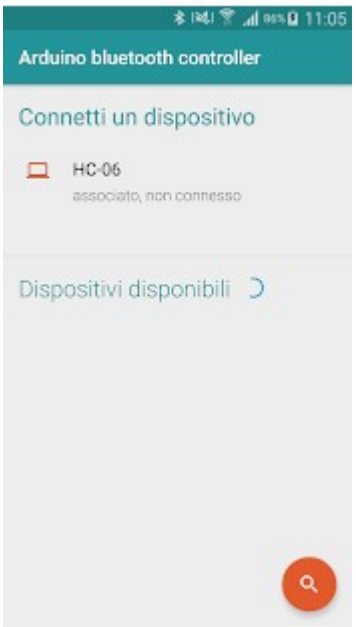
Aplicación en el móvil

Antes de nada en el móvil tengo que activar la conexión bluetooth, e instalar de la Tienda de aplicaciones (Google play Store en mi caso) la aplicación Arduino Bluetooth Controler.





Al abrir la app lo primero que haces es buscar dispositivos y emparejarlos, la contraseña de los dispositivos suele ser “1234”.



Más tarde usamos el modo “Controller mode” y configuramos los datos a enviar pulsando cada botón del interface, para ello pulsamos la rueda dentada. En nuestro caso un botón debe enviar el carácter “1” para encender el led y otro botón otro carácter para apagarlo.