Práctica 11: Arduino y sensor lumínico

¿Te acuerdas de los **bastones**, aquellas células **fotorreceptoras** de las que hablabamos en este apartado? Pues el funcionamiento del **sensor** que vamos a utilizar aquí es bastante similar...

Ha llegado el momento de que veamos un nuevo sensor, el cual resulta interesante tanto por la facilidad de su uso como por su reducido coste. Existen diferentes versiones, pero para esta práctica, vamos a utilizar la versión sencilla, directamente la **fotorresistencia**.

Antes de empezar, ¿qué es un divisor de voltaje?

Ya hemos visto que para trabajar con Arduino, son necesarias una serie de estructuras de programación. Aparte de eso, a veces vamos a necesitar ciertos componentes que permitan leer los valores de nuestros sensores.

La mayor parte de los sensores que utilizamos con Arduino son **plug-and-play**, es decir, no necesitan nada extra para funcionar, pero, en ocasiones, necesitaremos alguna cosilla extra. Ese es el caso de nuestra fotorresistencia. Para poder leer esa variación en la fotorresistencia que se produce cuando aumenta o disminuye la luz, necesitaremos conectar una **resistencia extra** que funcionará como **divisor de voltaje**. Esta resistencia también podría ser un potenciómetro, o lo que es lo mismo, una resistencia con un valor variable que podemos ajustar.

Con un divisor de voltaje, seremos capaces de obtener a partir de la variación en la resistencia, una variación en el voltaje que será lo que lea nuestro Arduino a través del pin que configuremos como entrada.

> Para saber más sobre divisores de voltaje puedes visitar esta página.

Copyright 2026

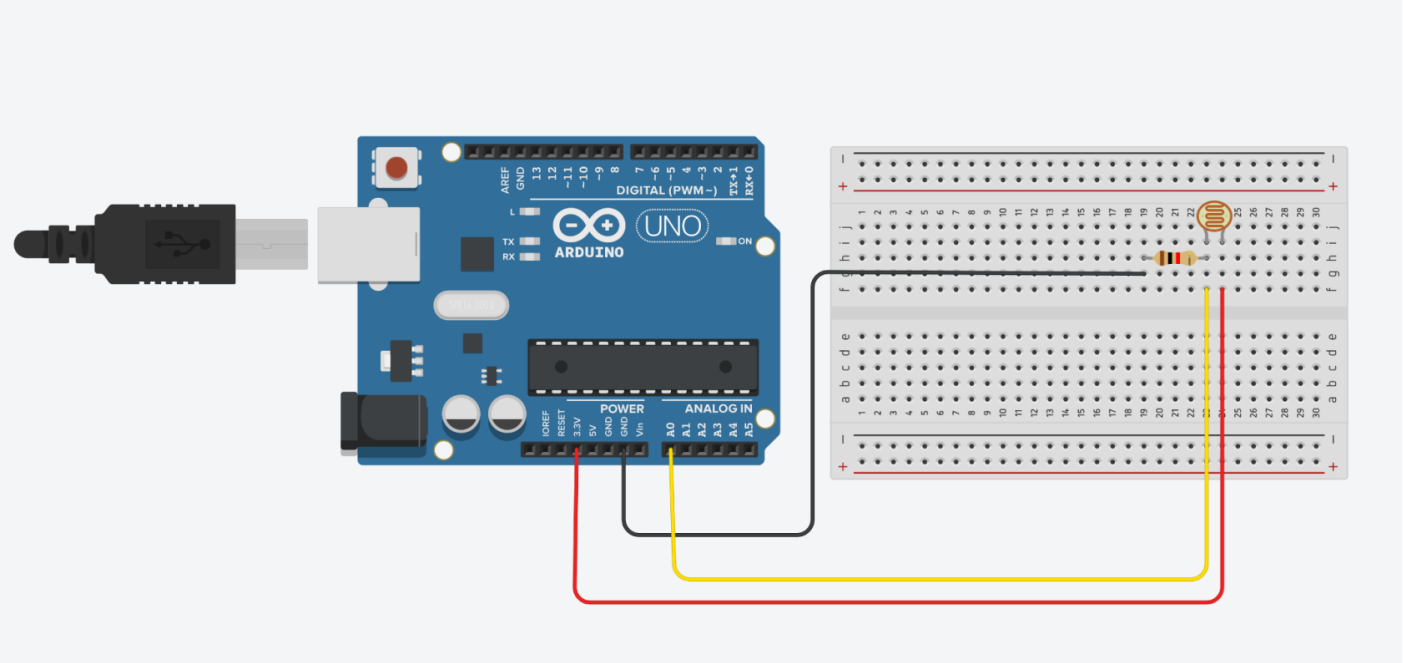
Materiales

Para realizar esta práctica necesitaremos:

1. Nuestro **Arduino UNO**
2. **3 Cables macho-macho**. ¿Por qué se llaman así? pues porque ambos extremos se insertan en alguna parte. Existen también cables macho-hembra y hembra-hembra, los cuales son útiles dependiendo del tipo de pines que tengan tanto el microcontrolador como los sensores y actuadores que usemos.
3. 1 resistencia de **10KOhm** que funcionará como divisor de voltaje.
4. **Cable AB USB** para conectar nuestro Arduino.
5. La **fotorresistencia**.

Estableciendo la conexión

Las conexiones que tenemos que realizar son las siguientes:



> Para realizar este circuito se ha utilizadoTinkercad, el cual nos permite diseñar circuitos de una manera visual.

> Recuerda cómo debes colocar los componentes en la protoboard, ya lo vimos en este apartado. ¡No realices cortocircuitos!

Ahí vemos que estamos empleando 3 pines:

A0: va desde ahí hasta una pata de nuestra **resistencia,** la cual está conectada a una pata de la **fotorresistencia.**

3.3V: va desde ahí hasta la pata de la **fotorresistencia que no está conectada** a la resistencia.

GND: va desde ahí hasta la pata de la **resistencia que no está conectada** a nada.

> Tenemos que tener cuidado con varios aspectos a la hora de realizar las conexiones en nuestra protoboard. El más importante es que todos esos puntitos están conectados unos con otros en línea. Es decir, en la posición en la que está colocada la protoboard de la imagen superior, los puntos están conectados en líneas verticales, de arriba a abajo, pero no de forma horizontal. Exceptuando las líneas roja y negra señalizadas con el símbolo + y - respectivamente, las cuales están conectadas de izquierda a derecha, si mantenemos la posición de la protoboard de la imagen de arriba.
> Por tanto, vemos que nuestro cable negro está conectado a uno de los extremos de nuestra resistencia, mientras que el otro extremo se conecta a un extremo de la fotorresistencia y al pin A0 de nuestro Arduino.

Una vez hemos terminado nuestro circuito va a llegar el momento de ver el código que necesitaremos para que funcione:

```
*
 * Created by ArduinoGetStarted.com
 *
 * This example code is in the public domain
 *
```

```
 * Tutorial page: https://arduinogetstarted.com/tutorials/arduino-light-sensor
 * Modificado por Marta Pérez
 */

void setup() {
  // comienza la comunicación del puerto serie en 9600 bits por segundo:
  Serial.begin(9600);
}

void loop() {
  // Lee el valor de A0 (entre 0 y 1023)
  int analogValue = analogRead(A0);

  Serial.print("Lectura analógica: ");
  Serial.print(analogValue);   // the raw analog reading

  // Definiremos los umbrales dependiendo de la cantidad de luz recibida
  if (analogValue < 10) {
    Serial.println(" - Oscuro");
  } else if (analogValue < 200) {
    Serial.println(" - Tenue");
  } else if (analogValue < 500) {
    Serial.println(" - Iluminado");
  } else if (analogValue < 800) {
    Serial.println(" - Bastante iluminado");
  } else {
    Serial.println(" - Mucha luz");
  }

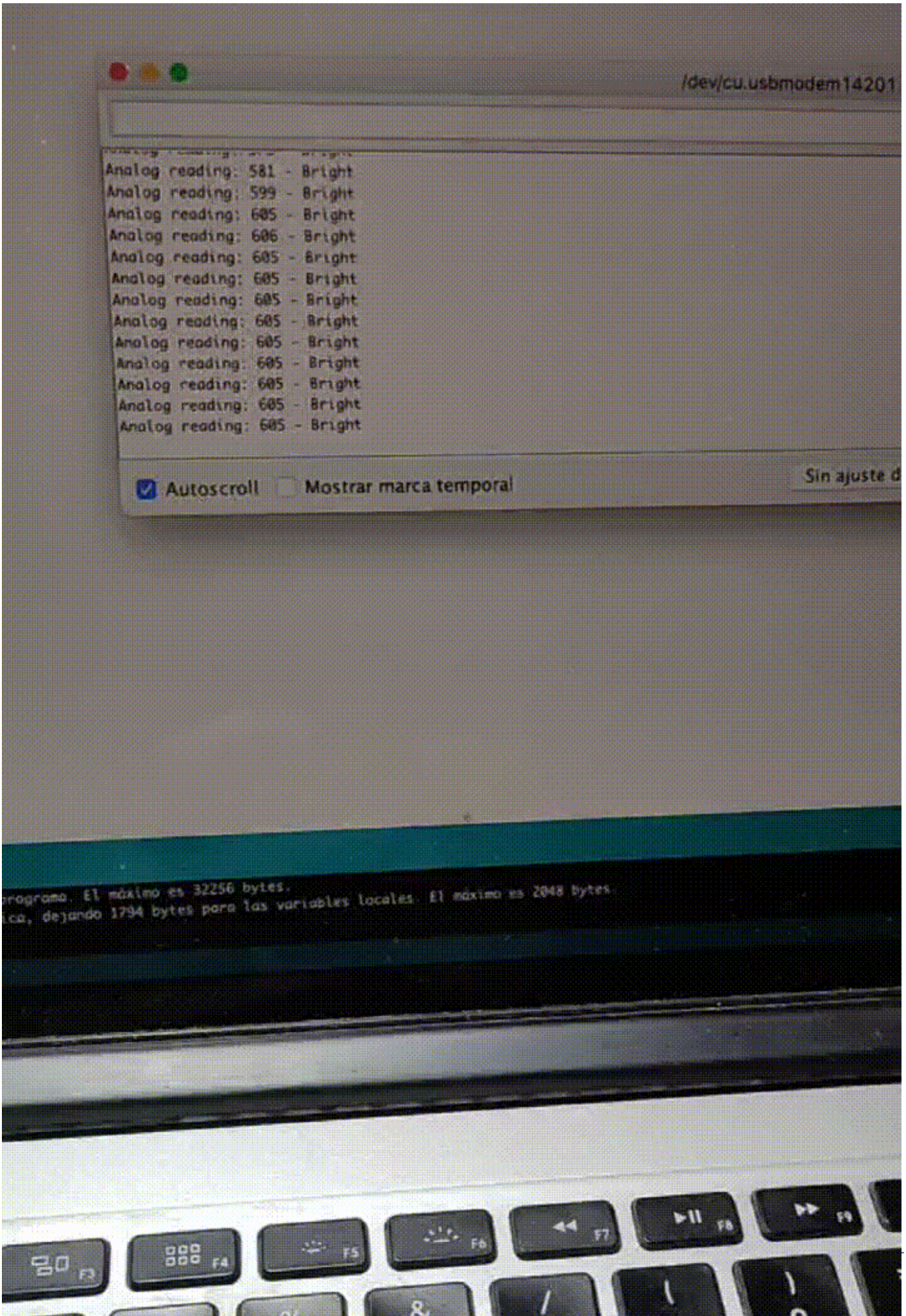
  delay(500);
```

Anteriormente, en la práctica anterior, se ha explicado línea a línea; un código compuesto por condicionales en su mayoría, por lo que no se ha considerado necesario explicar este código línea a línea.

Ahora conectamos nuestro Arduino al ordenador y subimos el código. El resultado que obtenemos una vez hayamos subido el código es:

Copyright 2026

Copyright 2026


/dev/cu.usbmodem14201
Analog reading: 581 - Bright
Analog reading: 599 - Bright
Analog reading: 605 - Bright
Analog reading: 606 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Analog reading: 605 - Bright
Autoscroll
Mostrar marca temporal
Sin ajuste d
rograma. El máximo es 32256 bytes.
ica, dejando 1794 bytes para las variables locales. El máximo es 2048 bytes.