



WAKE ME UP!

Reloj despertador y luz ambiente

Autora:

Elvira Castillo Fernández

Twitter: @layoel

Instagram: eca_ingeniera









WAKE ME UP!



- ☐ ¿Qué hace este proyecto?
 - Selecciona el color de los leds.
 - ☐ Diferentes **secuencias** de colores.
 - Velocidad de encendido de led.
 - Muestra la hora
 - ☐ **Programar** una alarma
 - ☐ Alarma visual o sonora

ITE ANIMAS A DARLE VIDA!







Hardware necesario para Wake ME UP!



WAKE ME UP!



- ARDUINO NANO
- ☐ Pantalla LCD + I2C
- ☐ Pila y reloj
- Mando y sensor IR
- Anillo leds RGB
- Buzzer
- Cables arduino







¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca_ingeniera

- Aprenderemos a:
 - ☐ ¿Que es l2C?
 - Conectar la pantalla.
 - Escribir mensajes.







Antes de empezar... ¿ Cavé tenso ave nacer para caroar vua lubrería?





- ☐ Las librerías las vamos a tener que cargar TODOS los días en los ordenadores del campus.
- Pero primero, antes de cargarlas, tenemos que configurar el IDE....







☐ Para ello, entramos en el IDE de arduino, y nos vamos a preferencias.

Archivo	<u>E</u> ditar	Programa	Herramien <u>t</u> as	Ayuda
00				
			Archivo Editar	Programa Herramier
			Nuevo	Ctrl+N
			Abrir	Ctrl+O
			Abrir Reciente	>
			Proyecto	>
			Ejemplos	→
			Cerrar	Ctrl+W
			Salvar	Ctrl+S LS,
			Guardar Como	. Ctrl+Mayús+S
			Configurar Pági	na Ctrl+Mayús+P
			Imprimir	Ctrl+P
			Preferencias	Ctrl÷Coma ls.o
			Salir	Ctrl+Q





- Aquí nos saldrá una pestaña como la que sale aquí, vemos que sale "Operador".
- Esto significa que se está usando la carpeta de operador, pero nosotras queremos nuestra carpeta "engiXX".
- Para ello...









Para ello...

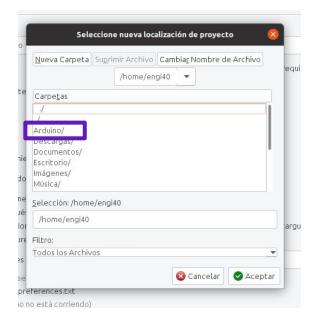
e proyecto	Seleccione nueva localización de proyecto				
ador/Arduino					
na:	Nueva Carpeta Suprimir Archivo Cambiar Nombre de Archivo /home/operador ▼	equiere			
año de Fuente	Carpetas	1			
z:	./ ./ Arduino/				
ı detallada mie	Descargas/ Documentos/ Escritorio/				
del compilado	Imágenes/ Música/				
imeros de líne	Selección: /home/operador				
ódigo despué	Arduino				
ar actualizacior	Aramo	argue			
sibility feature	Filtro:				
_s Adicionales	Todos los Archivos	•			
cias pueden se	⊗ Cancelar ⊘ Acepta	эг			
/.arduino15/pre	terences by t				

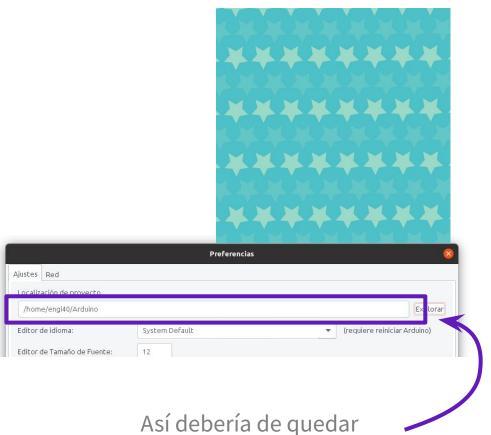
oyecto							
r/Arduino	Seleccione nueva localización de proyecto						
	Nueva Carpeta Suprimir Archivo Cambiar Nombre de Archivo						
	/home ▼	equiere					
de Fuente	Carpetas						
	V						
	engi40/						
	operadory						
:allada mie							
compilado							
ros de líne	Selección: /home						
go despué	/home						
tualizacior:							
ity feature	Filtro: Todos los Archivos						
dicionales	(IOOOS IOS AICHIVOS						
pueden se	⊗ Cancelar ⊘ Aceptar						
duino15/pre	rerences.txt						





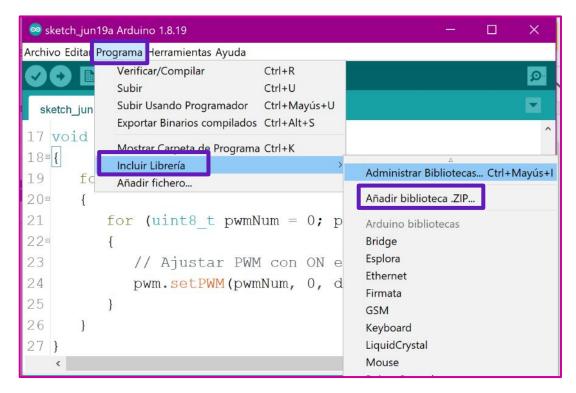
Para ello...











Ya después de esta configuración,

podemos añadimos las **librerías** necesarias.

Repetir este paso para cada una de las librerías.





Conectando el hardware del Wake Me UP!



Escribiendo en la pantalla



https://sereingeniera.ugr.es/index.php/2020/07/08/ampliaciones-del-proyecto-laser-tag/



PANTALLA LCD CON MÓDULO IZC



☐ ¿Qué es y cómo funciona una pantalla LCD?

Una **pantalla LCD** es una pantalla de retroiluminación LED que permite mostrar dos filas de 16 caracteres con los que podemos escribir texto.

Realmente no vamos a aprender a conectar la pantalla LCD directamente, ya que tiene muchos pines y si la conectáramos nos quedaríamos sin pines en el Arduino para todo lo demás. Por eso vamos a utilizar un **controlador I2C**.





PANTALLA LCD CON MÓDULO IZC



☐ ¿Que es I2C y para qué sirve?

El controlador I2C nos permite controlar la pantalla utilizando solamente dos cables de control (además del cable de 5v y la toma a tierra GND) a través de un tipo de comunicación entre placas llamada I2C.

- Los dos cables utilizados son SCL, que se conecta al pin analógico A5, y SDA, que se conecta al pin analógico A4:
- SCL envía señales de reloj (clock), que se encargan de decidir quién habla en cada momento, para evitar conflictos y SDA envía los datos.

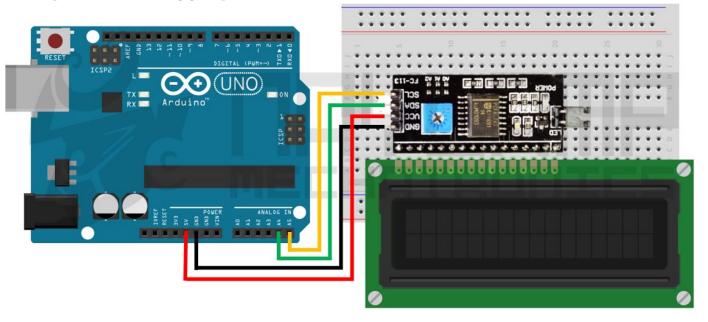




MONTAJE PANTALLA LCD

SCL-> A5 GND-> GND

SDA-> A4 VCC-> 5V







Programando el Wake Me UP!



Escribiendo en la pantalla



https://sereingeniera.ugr.es/index.php/2020/07/08/ampliaciones-del-proyecto-laser-tag/



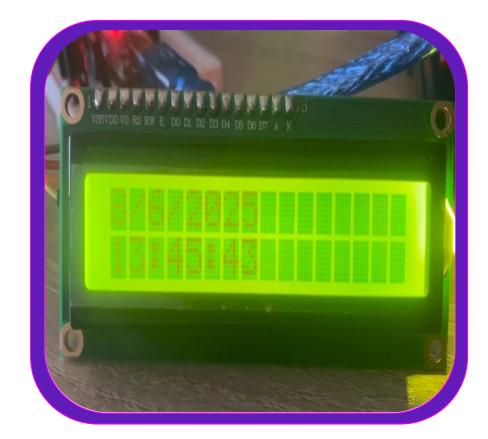
S Officia de Conversible ESCRIBIENDO MENSAJES PANTALLA LCD



En esta pantalla mostraremos el día y la hora y los mensajes que haya de configuración de nuestro reloj despertador

Recuerda los pines son:

SCL-> A5 GND-> GND SDA-> A4 VCC-> 5V





ESCRIBIENDO MENSAJES PANTALLA LCD universidad de Granada

- 1. Incluir en el IDE la librería LiquidCrystal_I2C.h
- ☐ Hay **dos formas de incluir librerías** a la biblioteca de nuestro arduino.
 - ☐ Si tenemos el archivo comprimido con extensión .zip
 - ☐ Buscarlas en el **repositorio** de Arduino

- Para la pantalla tenemos el archivo comprimido con el nombreLiquidCrystal_I2C-1.1.2.zip
- La **descargamos** y la incluimos de la 1º forma que vemos a continuación.





1. Descarga la librería y después la importamos en el arduino:

Está en el enlace:

http://downloads.arduino.cc/libraries/github.com/marcoschwartz/LiquidCrystal I2C-1.1.2.zip





- Para poder utilizar la pantalla con I2C además de descargarnos la librería de Arduino llamada <LiquidCrystal_I2C.h>, también necesitamos cargar otra llamada <Wire.h>.
- Al trabajar con funciones de la librería descargada, a las que no les hemos puesto el nombre nosotras, tenemos que acostumbrarnos y entender lo que hacen.

personas y que nosotras podemos aprovechar y utilizar siempre que lo necesitemos.





1. Cargar librerías en el arduino:

En arduino hacemos clic en **Programa-> incluir libreria-> añadir biblioteca Zip.**







1. Cargar librerías en el arduino:

una vez añadida, para poder usarla hay que ir a:

Programa-> incluir biblioteca/libreria-> LiquidCrystal



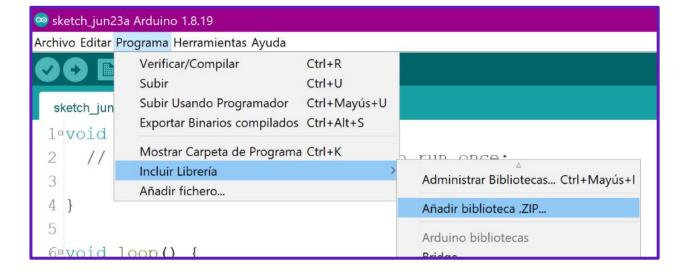






1. Incluir en el IDE la librería LiquidCrystal.h





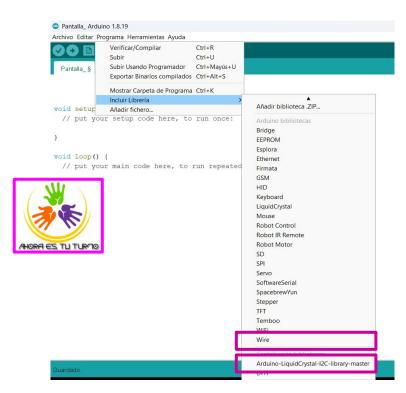


PANTALLA LCD



2. Para poder usar la librería tenemos que añadirla en el archivo. Vamos a añadir

también wire.



```
Pantalla_ Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
  Pantalla
#include <Wire.h>
#include <LiquidCrystal I2C.h>
void setup() {
  // put your setup code here, to run once:
void loop() {
  // put your main code here, to run repeatedly:
```



PANTALLA LCD



3. Declaramos un objeto basándonos en nuestra pantalla

- □ Para ello utilizaremos la siguiente función de la librería LiquidCrystal_I2C que acabamos de incluir.
 - LiquidCrystal_I2C lcd(0x27, 16, 2);
- → Se inicializa con:
 - Ox27 porque se inicializa en esa dirección
 - ☐ 16 = número de caracteres por línea
 - ☐ 2 = número de líneas

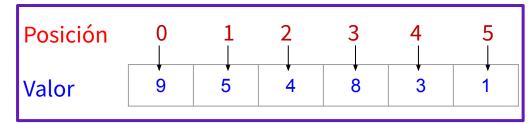
■ Vale pero, ¿qué tenemos que hacer y cómo? Poco a poco



PANTALLA LDC



- 4. Aprendemos a usar la librería para utilizar cada punto de nuestra pantalla.
- Antes de empezar a usar las funciones de la librería necesitamos saber cómo usar las posiciones de cada línea.
- ¿Sabes qué es un vector?



- El vector tiene **posiciones**
- La primera posición siempre empieza en 0
- **Cada posición** tiene **un valor** (en la imagen, la posición 0 tiene valor 9, la posición 4 tiene valor 3...)
- El tamaño máximo del vector es el número de posiciones que tiene. En este ejemplo el vector tiene 6 posiciones que van del 0 al 5.
- En nuestra pantalla, habrá dos vectores, que irán del... 0 al 15



PANTALLA LDC



4. Aprendemos a usar la librería para mostrar mensajes por pantalla:

- Para poder utilizar nuestra pantalla vamos a tener que aprender las funciones de la librería:
 - lcd.begin() se utiliza para inicializar la comunicación con la pantalla LCD.
 - □ lcd.backlight() enciende la retroiluminación de la pantalla LCD.
 - lcd.setCursor(0, 0) establece la posición del cursor en la columna 0 y la fila 0.
 - □ lcd.print("Hola") muestra el texto "Hola" en la pantalla LCD.
 - □ **lcd.clear() borra** el contenido de la pantalla LCD.
 - Esas son las básicas, pero hay alguna más a ver si eres capaz de localizarlas ;)

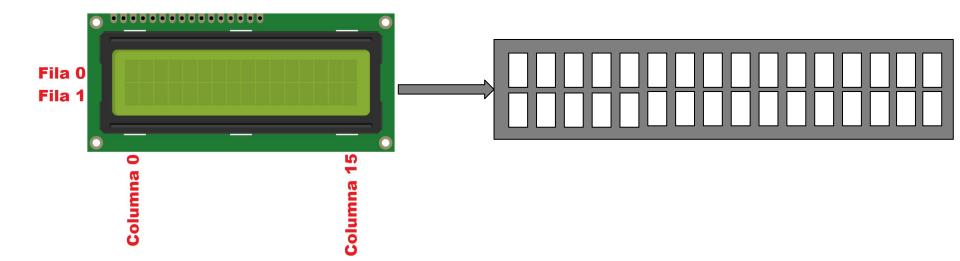
¿Creéis que podríais hacer que la pantalla muestre en la primera línea "Hola"?



PANTALLA LCD



- 4. Aprendemos a usar la librería para mostrar mensajes por pantalla: EXTRA:
- ☐ Vamos a entrar más en profundidad en cómo se muestran las letras en pantalla:
 - ☐ Siempre se escribe de la fila 0 posición 0 a la fila 0 posición 15
 - segunda fila es la fila 1 posición 0 a la fila 1 posición 15





PANTALLA LDC



4. Aprendemos a usar las funciones de la librería para mostrar mensajes por

pantalla:

- lcd.init() inicia la pantalla
- lcd.backlight() enciende la luz
- lcd.setCursor(0, 0) posiciona el cursor
- lcd.print("Hola") escribe en la pantalla
- □ lcd.clear() borra de la pantalla

pseudocódigo a código real con las funciones que te hemos enseñado



```
#include <LiquidCrystal I2C.h>
#include <Wire.h>
LiquidCrystal I2C lcd(0x27, 16, 2);
 // Inicia el LCD en la dirección 0x27, con 16 caracteres y 2 líneas
void setup()
       inicio la pantalla
       luz de fondo
 void loop() {
       empiezo a escribir en línea0 pos0
       añado el texto que quiera
       espero 2 segundos
       limpio pantalla
```





- Ahora tienes que escribir en la pantalla en la primera línea la fecha y en la segunda línea la hora. (escribirlos de forma manual)
- Prueba diferentes formatos de fecha y hora
- Prueba a añadir espacios para centrar los mensajes
- Prueba a que se desplace hacia la derecha y/o izquierda
 el texto (investiga en internet cual es la función que lo hace)
- □ Elige el formato de fecha y hora que más te guste. por ej: lun 3 mar 2025, 3/3/2025...





CÓDIGO DE PRUEBA



https://controlautomaticoeducacion.com/sistemas-embebidos/arduino/bus-comunicacion-i2c/





¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca ingeniera



- Configurar el módulo de Reloj.
- Mostrar en el monitor serie la fecha y hora
- Mostrar en la pantalla la fecha y la hora.
- Crear funciones.
- Programar una alarma.







Conectando el hardware del Waxe Me UP!



Reloj de tiempo real MÓDULO DS1302 RTC





MÓDULO RELAT DE TIEMPO REAL



- El arduino no es capaz de hacer varias cosas a la vez.
- Cuando se desconecta de la corriente y se vuelve a conectar, se reinicia el programa.
- Pero, si le conectamos un reloj, podemos indicar al arduino que realice diferentes acciones según la hora del día.
- Es por eso que vamos a usar este componente reloj que tiene unas características peculiares.





MÓDULO RELAT DE TIEMPO REAL



- El módulo reloj DS1302 RTC permite **ajustar la fecha y la hora**, contiene un reloj-calendario en tiempo real.
- El módulo reloj proporciona información de segundos, minutos, días, semanas, meses y años.
- Incluye una pila con la que puede mantener la actualización de la fecha y hora aunque se apague el arduino no se desconfigura a menos que le quitemos la pila.



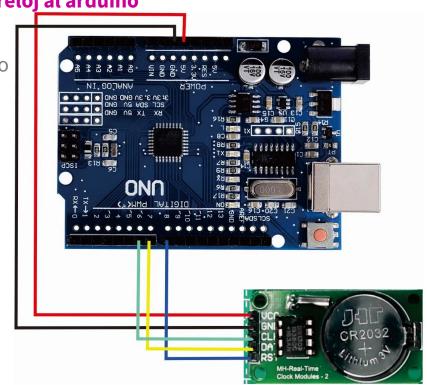




Instrucciones para conectar el módulo de reloj al arduino

- Conectar **VCC** con el pin de **5V** de Arduino
- ☐ GND con el GND de Arduino
- ☐ CLK al pin D6 de Arduino
- **DAT** con el pin **D7** de Arduino
- RST al pin D8 de Arduino









Programando el Wake Me UP!



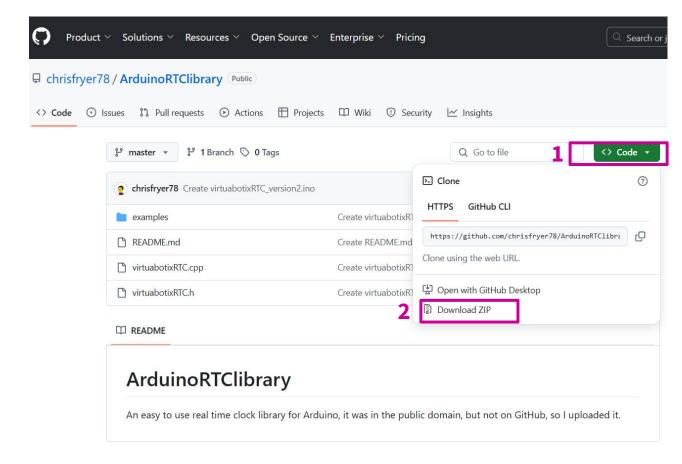
Reloj de tiempo real MÓDULO DS1302 RTC







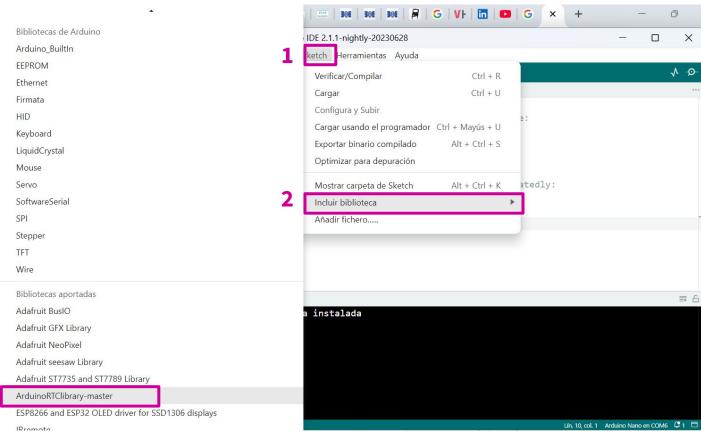
Descargamos la biblioteca del enlace:
https://github.c
om/chrisfryer78
/ArduinoRTClibr
ary/tree/master







■ Añadimos la biblioteca al IDE de arduino







- Las **funciones** que podemos usar con este módulo de reloj son:
 - □ Para configurar la fecha y hora exactas
 reloj.setDS1302Time(segundos, minutos, horas, día de la semana, dia del mes, mes, año)
 - Para consultar la fecha y hora reloj.updateTime()







- Comencemos con el programa:
- Incluye la librería
- Crea las variables con los pines de arduino donde has conectado el reloj.
- Creamos el objeto reloj utilizando la función virtuabotixRTC
- En el setup inicia el monitor serie
- Fija la configuración del reloj usando la función **SET** que vimos antes.
- En el **loop** actualiza la fecha y hora con la función **UPDATE**

```
no IDE 2.1.1-nightly-20230628
Editar Sketch Herramientas Ayuda
     4 Arduino Nano
    #include "virtuabotixRTC.h"
    //define los pines PinSCLK, PinDAT, pinRST donde van conectados al arduino
     //creamos el objeto reloj
    virtuabotixRTC reloj(PinSCLK, PinDAT, pinRST);
    void setup() {
      Serial.begin(9600);
      //reloj.set.... Configuración del tiempo actual
      // Recuerda comentar la línea anterior, una vez configurado
12
    void loop(){
    // actualiza fecha y hora
    // imprime en el monitor serie la fecha y la hora
```





MOTA: Cambia los comentarios por el código correspondiente.

//define los pines...

//reloj.set...

//actualiza fecha y hora

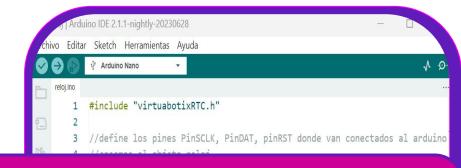
//imprime en el monitor serie la fecha y la hora. (lo subrayado lo veremos más adelante)

```
YOU CAN DO IT!
                        2.1.1-nightly-20230628
                   #include "virtuabotixRTC.h"
                    //define los pines PinSCLK, PinDAT, pinRST donde van conectados al arduino
                    //creamos el objeto reloj
                    virtuabotixRTC reloj(PinSCLK, PinDAT, pinRST);
                    void setup() {
                     Serial.begin(9600);
                     //reloj.set.... Configuración del tiempo actual
                     // Recuerda comentar la línea anterior, una vez configurado
                    void loop(){
                   // actualiza fecha y hora
                    // imprime en el monitor serie la fecha y la hora
```





- Comencemos con el programa:
- Incluye la librería
- Crea las variables con los pines de arduino donde has conectado el reloj.
 - Creamos el obieto reloi utilizando la



NOTA: El reloj se configura una sola vez y se guarda para siempre mientras tenga la pila puesta el módulo.

Una vez configurado, comenta la línea del **reloj. Set** que escribiste en el setup para definir la fecha, hora, día etc...

con la función **UPDATE**

```
15 // actualiza fecha y hora
16 // imprime en el monitor serie la fecha y la hora
17
18 }
```





- Ya hemos visto que las **funciones** que podemos usar con este módulo de reloj son:
 - Para configurar la fecha y hora exactas
 reloj.setDS1302Time(segundos, minutos, horas, día de la semana, dia del mes, mes, año)
 - Para consultar la fecha y hora reloj.updateTime()
- Además, después de usar updateTime podemos consultar las variables imprimibles que son:
 - □ reloj.dayofmonth Muestra el día del mes
 - reloj.dayofweek Muestra el dia de la semana (en número, no letras)
 - reloj.month Muestra el mes (en número)
 - □ reloj.year Muestra el año
 - □ reloj.hours Muestra la hora
 - □ reloj.minutes Muestra los minutos
 - □ reloj.seconds Muestra los segundos





- Ahora, en el loop guarda los valores de las variables del reloj anteriores en palabras es decir:
 - ☐ Crea las variables de **tipo int** con los siguientes nombres: diadelmes, anio, hora, minuto, segundo
 - ☐ Crea las variables de **tipo String** con los siguientes nombres: diadelasemana, mes
 - Guarda en diadelmes reloj.dayofmonth, en anio reloj.year, en hora reloj.hours, en minuto reloj.minutes, en segundo reloj.seconds

 Las variables de tipo string las iniciamos con la cadena vacía las usaremos más adelante...
 - **□** Ejemplos:

```
int diadelmes = reloj.....;
string diadelasemana = " ";
```







```
void loop() {
   // put your main code here, to run repeatedly:
   reloj.updateTime();
                                     NOTA: escribe la variable correcta en
   int diadelmes = reloj....;
                                               cada caso.
   int anio = reloj....;
                                     Los string son cadenas vacías por lo
   int hora = reloj....;
                                      que entre las "" llevan un espacio
   int minuto = reloj....;
                                     hasta que les demos otro valor que
   int segundo = reloj....;
                                            será más adelante.
   String diadelasemana = " ";
   String mes = " ";
```





- ☐ Imprime en el monitor serie el valor de las **variables**:
 - diadelmes
 - ☐ reloj.dayofweek
 - reloj.month
 - anio
 - hora
 - minuto
 - segundo









```
NOTA: Para darle formato, tienes que ponerlo por partes:
```

```
imprime reloj.dayofweek
imprime diadelmes
imprime "/"
imprime reloj.month
imprime "/"
imprime anio
```

La hora igual también por partes:

```
imprime hora
imprime ":"
imprime minuto
imprime ":"
imprime segundo
```

```
Arduino IDE 2.1.1-nightly-20230628
chivo Editar Sketch Herramientas Ayuda
         4 Arduino Nano
        #include "virtuabotixRTC.h"
         //define los pines PinSCLK, PinDAT, pinRST donde van conectados al anduino
         //creamos el objeto reloj
         virtuabotixRTC reloj(PinSCLK, PinDAT, pinRST);
         void setup() {
          Serial.begin(9600);
          //reloj.set.... Configuración del tiempo actual
          // Recuerda comentar la línea anterior, una vez configurado
         void loop(){
         // actualiza fecha y hora
         // imprime en el monitor serie la fecha y la hora
```





- ☐ Continúa editando el programa para realizar los siguientes cambios:
- Muestra el **día de la semana en texto** en el monitor serie.
 - Recuerda los **condicionales** que vimos en el módulo de introducción a arduino:
 - El pseudo código sería:
 - ☐ Si **reloj.dayofweek** es igual a 1 entonces

diadelasemana = "Lun"

escribe en el monitor serie diadelasemana;

si **reloj.dayofweek** es igual a 2 entonces

diadelasemana = "Mar"

escribe en el monitor serie diadelasemana;





```
if (reloj.month == 1){
   mes = " Ene ";
if (reloj.month == 2){
   mes = " Feb ";
if (reloj.month == 3){
   mes = " Mar ";
if (reloj.month == 4){
   mes = " Abr ";
if (reloj.month == 5){
   mes = " May ";
if (reloj.month == 6){
   mes = " Jun ";
if (reloj.month == 7){
   mes = " Jul ";
```





- ☐ Continúa editando el programa para realizar los siguientes cambios:
- ☐ Muestra el **mes en texto** abreviado en el monitor serie. (Ene, Feb, Mar, Abr,...)
 - ☐ Es igual que los días de la semana, un **condicional**.
 - El pseudo código sería:
 - ☐ Si **reloj.month** es igual a 1 entonces

escribe en el monitor serie mes

☐ Si **reloj.month** es igual a 2 entonces

```
mes = " Feb "
```

escribe en el monitor serie mes

_ ...





```
if (reloj.month == 1){
   mes = " Ene ";
if (reloj.month == 2){
   mes = " Feb ";
if (reloj.month == 3){
   mes = " Mar ";
if (reloj.month == 4){
   mes = " Abr ";
if (reloj.month == 5){
   mes = " May ";
if (reloj.month == 6){
   mes = " Jun ";
if (reloj.month == 7){
   mes = " Jul ";
```





Ahora que ya vemos la fecha y hora en el monitor serie...

¿Serías capaz de mostrarla en la pantalla LCD que programamos ayer?







- Ahora que ya vemos la fecha y hora en el monitor serie...
- ☐ Una ayudita... tan solo tienes que:
 - Incluir en el código que tienes para el reloj en el código de ayer de la pantalla. (te recomiendo que crees un scketch nuevo que se llame PantallaReloj donde unas ambos códigos (en lugar de editar los que ya te funcionan de manera individual será más fácil detectar errores)
 - ☐ En el **setup** iniciar la pantalla y la luz de fondo.
 - Escribe en la **línea 0 la fecha** de hoy usando las función de la lcd para definir la posición del cursor y la función de imprimir con las variables que hemos creado en el programa del reloj (las numéricas **actualizar_hora**, **diadelmes**, **reloj.month**, **anio**, **hora**, **minuto**, **segundo**, o si lo prefieres combinalas con las que están en palabra **diadelasemana**, **mes**,)
 - Escribe en la **línea 1 la hora** (usa el formato que más te guste, también puedes añadir el dia de la semana en texto o el mes en texto como hicimos en el ejemplo anterior)
 - No te olvides de usar la función actualizar del reloj para que se actualice en la pantalla periódicamente.









S Officia de la CONTROLLA CONTROLLA CONTROLLA DE GRANADA DE GRANADA

Ahora que ya sabemos como escribir en la pantalla, vamos a **crear un código que**:

Actualiza el valor de la variable actualizar_hora igual a reloj.updateTime();

Si hora es mayor o igual que 7 y hora es menor que 12

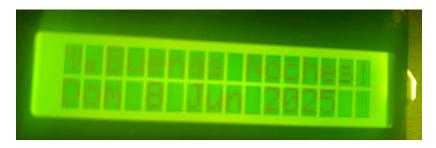
Escribe en la primera línea de la pantalla el mensaje ¡Buenos Días!

sino Si hora es mayor que 12 y hora es menor que 19

Escribe en la primera línea de la pantalla el mensaje ¡Buenas Tardes! sino (en cualquier otro caso):

Escribe el mensaje ¡Buenas noches! en la pantalla.

Escribe en la segunda línea de la pantalla la fecha de hoy



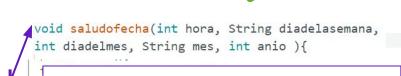


PASAMOS EL CÓDIGO A FUNCIONES





- Las funciones son agrupaciones de código
- ☐ Ya conocemos las funciones
 - void setup(){ ... }
 - □ void loop() { ... }
- Cuando delante del nombre de la función aparece void significa que la función no devuelve nada.
- Las funciones pueden devolver variables.
- Delante del nombre de la función hay que añadir el tipo de variable que devuelve. /
- Esta función es void porque no devuel√e nada, ejecuta todo lo de dentro, pero no devuelve ninguna variable.



METE AQUÍ DENTRO EL CÓDIGO QUE HAS ESCRITO



Esto hay que hacerlo en un nuevo skech. Las funciones se añaden debajo del loop o encima del setup



PASAMOS EL CÓDIGO A FUNCIONES



- Ahora que vas teniendo mucha funcionalidad, vamos a pasar el código a funciones para que el loop se vea más limpio y sea más fácil de leer.
- Copia el código correspondiente de cada funcionalidad dentro de su función.
- Las funciones se definen fuera del loop.
- En la siguiente diapositiva tienes la estructura de las funciones.
- Cambia los ... por el código que tienes escrito que hace esa funcionalidad dentro de tu loop.

En estos casos no hay return, la función no

devuelve ningún valor, por eso es de tipo **void**

Si te fijas, en estos casos devuelven una cadena de texto, por eso delante del nombre de la función aparece el tipo String. Tanto dia

como **mes** son

texto, son **strings**.

. . .

. . .

String DimeDia(int numDia){ String dia= " "; return dia;

escribimos los meses abreviados String DimeMes(int numMes){ String mes= " "; . . . return mes;

//escribimos los dias de la semana en texto

void fechahora(String diadelasemana, int diadelmes,

String mes, int anio, int hora, int minuto, int segundo){



PASAMOS EL CÓDIGO A FUNCIONES



Ahora el loop debe quedar así:

```
void loop() {
  // put your main code here, to run repeatedly:
 reloj.updateTime();
 int diadelmes = reloj.dayofmonth;
  int anio = reloj.year ;
  int hora = reloj.hours ;
  int minuto = reloj.minutes ;
  int segundo = reloj.seconds ;
  String diadelasemana = " ";
  String mes = " ";
mes = DimeMes(reloj.month);
diadelasemana = DimeDia(reloj.dayofweek);
//fechahora(diadelasemana, diadelmes, mes, anio, hora, minuto, segundo);
saludofecha(hora, diadelasemana, diadelmes, mes, anio);
```



LA ALARMA



Ahora que ya tenemos el código más limpio, vamos a programar nuestra alarma. Tenemos que crear dos variables nuevas:

alarm_hora y alarm_minutos ahora os dejo el pseudo código:

Si (hora es igual a alarm_hora Y minutos es igual a alarm_minutos){

limpia la pantalla pon el cursor en la posición inicial escribe en la pantalla ¡Despierta! espera medio segundo





void loop() {

delay(500);

// put your main code here, to run repeatedly: reloj.updateTime(); int diadelmes = reloj.dayofmonth ; int anio = reloj.year ;

```
int hora = reloj.hours ;
int minuto = reloj.minutes ;
int segundo = reloj.seconds ;
String diadelasemana = " ";
String mes = " ";
//¿a que hora saltará la alarma?
int alarm hora = 23;//hora de programación de la alarma
int alarm_minuto = 35; //minutos de programación de la alarma
mes = DimeMes(reloj.month);
diadelasemana = DimeDia(reloj.dayofweek);
lcd.clear();
//si es la hora, saltará la alarma durante 1 minuto)
if(hora == alarm_hora && minuto == alarm_minuto){
    Thunder();
   lcd.clear();
   lcd.setCursor(0, 0);
    lcd.print("¡Despierta!");
```





El arduino no es capaz de hacer varias cosas a la vez.

Cuando se desconecta de la corriente y se vuelve a conectar, se reinicia.

Ahora programamos que en la pantalla aparezca la fecha y la hora actual.

Este coincide los pines que uso en la imagen con el ejemplo

https://eloctavobit.com/modulos-sensores/real-time-clock-rtc-ds1302-modulo-reloj

https://www.youtube.com/watch?v= QNrMw79qZ8I

<u>https://miliohm.com/ds1</u> <u>302-rtc-with-arduino-tut</u> <u>orial/</u> codigo de prueba

PROGRAMAR acciones con el reloj DS1302 RTC en Arduino

https://www.youtube.com/watch?v= u3kUj-hV6Rk

https://www.tecneu.com/blogs/tutoriales-de-electronica/integracion-del-ds1302-con-arduino-configuracion-de-fecha-y-hora?srsltid=AfmBOorrqAdgS2MnOqaHgb1fnwlrtPMSehnev3Synbwwhimiw3OF2hi9C





¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca_ingeniera

- Aprenderemos a
 - Conector los leds.
 - Programar los leds, creando diferentes secuencias.



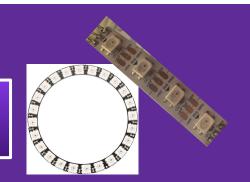




Montaje del Wake Me UP!



¡Hágase la luz! Los Led RGB



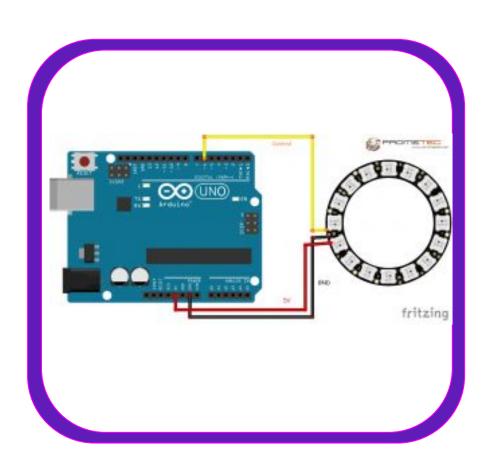


JUGUEMOS CON LOS LEDS 21



- ☐ Leds RGB direccionables.
 - Tienen 4 pines:
 - ☐ **DIN** pin de entrada
 - **DO** pin de salida
 - +5V proporciona energía a los leds
 - ☐ GND Toma de tierra

Si usaramos otro anillo de led, el **DIN** del primero debe ir conectado al **DO** del segundo.









- **☐** Leds RGB directionables.
- Se llaman direccionables porque las señales se transmiten en una dirección a través de los leds.
- ☐ Si fuera una tira de led, podríamos ver cómo están conectados como aparece en las imágenes
- El anillo está conectado igual pero no lo vemos.
- Tu cable del pin correspondiente de arduino siempre tiene que ir al DIN de tu anillo de leds.

GND	2	GND
DIN	5 1 20	DO
+5V	a la	+5V
	7 4 0 0 1	_







IHÁGASE LA LUZ!



Vamos a soldar los cables
Pero...

CUIDADO CON QUEMAR5E!!



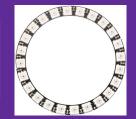




Conectando el hardware del Wake Me UP!



¡Hágase la luz! Los Led RGB





WAKE ME UP!



□ Leds RGB direccionables.

Se llaman direccionables porque las señales se transmiten en una dirección a través

de los leds

cta el último

Conectai

- **DIN**
- GND
- **□** 5V a

Nota: El anillo de led es igual que la tira de led (ambos están formados por Leds RGB direccionables con chip propio integrado) pero en el caso del anillo su localización es circular. Las conexiones son las mismas pero en el anillo están integradas y sólo nos muestra, en el inicio del anillo los pines **DIN GND VCC/5V** y en el fin del anillo el **DO** en lugar de aparecer entre todos los leds como en la tira de led.





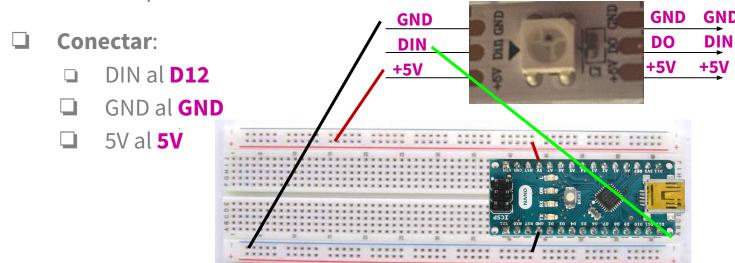




WAKE ME UP!



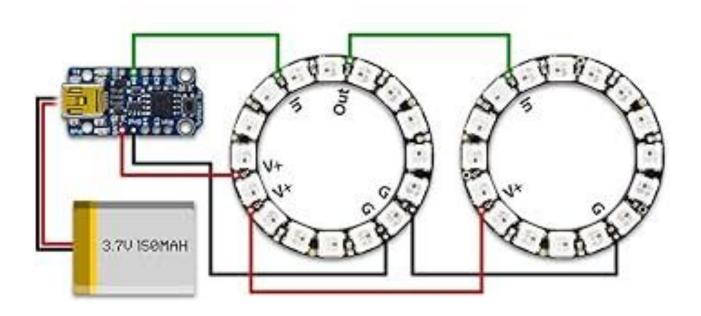
- **■** Leds RGB directionables.
- Se llaman direccionables porque las señales se transmiten en una dirección a través de los leds. **Del Arduino al primer led, al segundo, al tercero... hasta el último**
- □ El **chip** que contienen **integrado**, "**asigna un número**" (dirección de memoria) a cada led para identificarlo en la tira.





RGB Ring Connection Figure 3



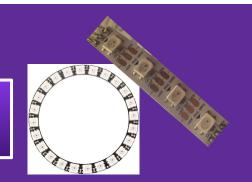






Programando el Wake Me UP!

¡Hágase la luz! Los Led RGB







Abrimos un nuevo sketch de arduino en el IDE para hacer pruebas con el aro de leds.

Después lo integraremos con todo lo anterior





- 1. Incluir en el IDE la librería Adafruit_NeoPixel
- Hay dos formas de incluir librerías a la biblioteca de nuestro arduino.
 - ☐ Si tenemos el archivo comprimido con extensión .zip
 - Buscarlas en el repositorio de Arduino

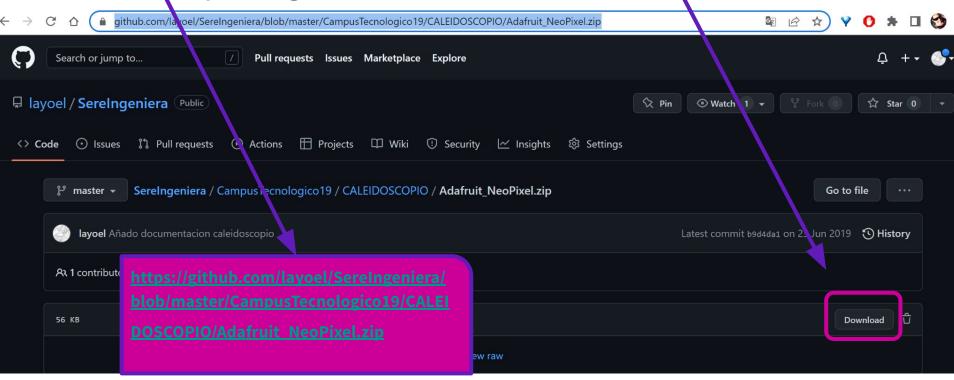
- → Para la tira de led tenemos el archivo comprimido con el nombre Adafruit_NeoPixel.zip
- La **descargamos** y la incluimos de la 1º forma.





1. Incluir en el IDE la librería Adafruit_NeoPixel

Vamos al enlace y descargamos la librería haciendo clic en download



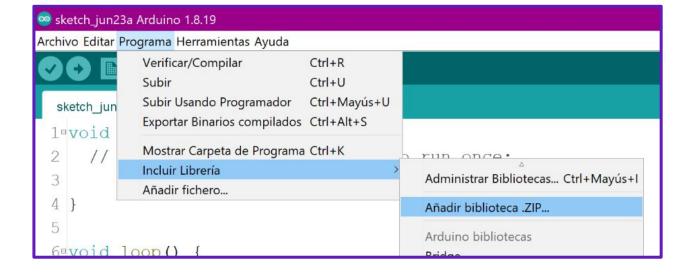






1. Incluir en el IDE la librería Adafruit_NeoPixel

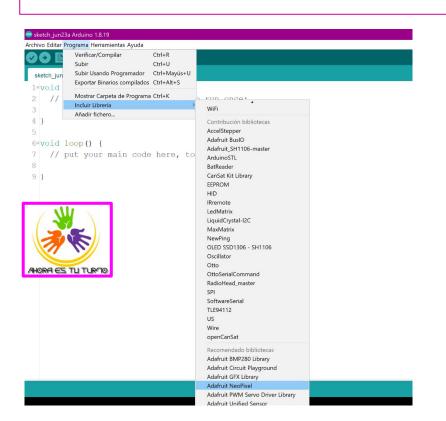








2. Para poder usar la librería tenemos que añadirla en el sketch.



```
🔯 sketch_jun23a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
  sketch jun23a 8
   #include <Adafruit NeoPixel.h>
 3 void setup() {
      // put your setup code here, to run once:
 8 void loop() {
      // put your main code here, to run repeatedly:
10
```





- 3. Declaramos un objeto con la cadena de leds que vamos a emplear.
- Para ello utilizaremos la siguiente **función de la librería NeoPixel** que acabamos de incluir.
 - Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
- Los parámetros que se pasan a la función son:
 - NUMPIXELS que será una variable donde pondremos el número de leds que queremos controlar
 - PIN que será el pin de arduino donde hemos conectado el IN de la tira de led
 - NEO_GRB + NEO_KHZ800 este parámetro indica que usaremos leds de colores y la controladora que utilizan.

☐ Vale pero, ¿qué tenemos que hacer y cómo?





3. Declaramos un objeto con la cadena de leds que vamos a emplear.

Aquí tenéis un ejemplo

para controlar 3 leds,
vuestra tarea es
adaptarlo para poder
usar los 24 leds.

Esto hay que hacerlo en el sketch donde vamos a programar los leds para probarlos y añadirlo al sketch del programa final

```
🔯 sketch jun23a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
  sketch jun23a §
 1 #include <Adafruit NeoPixel.h>
  3 #define PIN
 4 #define NUMPIXELS 3
 6 using namespace std;
 8 Adafruit NeoPixel pixels (NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
10 void setup() {
     // put your setup code here, to run once:
13
15 void loop() {
     // put your main code here, to run repeatedly:
```





- 4. Aprendemos a usar la librería para encender uno a uno los leds del aro de led de colores.
- Antes de empezar a usar las funciones de la librería necesitamos saber cómo acceder a las posiciones de cada led.

¿Sabes que es un vector?

Posición	0	1 ↓	2	3	4 ↓	5 ↓
Valor	9	5	4	8	3	1

- El vector tiene **posiciones**
- La primera posición siempre empieza en 0
- **Cada posición** tiene **un valor** (en la imagen, la posición 0 tiene valor 9, la posición 4 tiene valor 3...)
- El tamaño máximo del vector es el número de posiciones que tiene. En este ejemplo el vector tiene 6 posiciones que van del 0 al 5.
- Los led se encienden en el mismo orden que el vector del 0 al 23 en nuestro caso si usas un aro o al 47 si usas dos.





4. Aprendemos a usar la librería para encender uno a uno los leds del aro de led de colores.



☐ ¿Cómo encedemos un led?







4. Aprendemos a usar la librería para encender uno a uno los leds de la tira de led de colores.



¿Cómo encedemos un led?

¡Usaremos las funciones ya predefinidas de la librería Neopixel!







- 4. Aprendemos a usar la librería para encender uno a uno los leds de la tira de led de colores.
- Utilizaremos estas funciones ya predefinidas de la librería Neopixel.
 - **pixels.begin();** esta función inicializa los leds.
 - **pixels.clear();** esta función borra la configuración guardada en los leds.
 - pixels.Color(R, G, B); esta función es para seleccionar el color.
 - R es el parámetro donde se pondrá el valor del color **rojo** (toma valores de 0 a 255)
 - ☐ G es el parámetro donde se pondrá el valor del color **verde** (toma valores de 0 a 255)
 - B es el parámetro donde se pondrá el valor del color azul (toma valores de 0 a 255)
 - pixels.setPixelColor(led, color);
 - led es el número de led que queremos que se encienda del color que pongamos.
 - color es el color que se va a poner y se sustituye por: pixels.Color(R, G, B);





4. Aprendemos a usar la librería para encender uno a uno los leds de la tira de led

de colores.

☐ En este ejemplo se usan 4 leds.

Prueba el código en tu sketch

- ☐ ¿Qué leds se encienden?
- ⊒ ¿En qué color?

Adapta el código para los 24 leds

■ Ve haciendo pruebas para encender uno u otro led o varios a la vez y probar los diferentes colores.

Esto hay que hacerlo en el sketch donde vamos a crear el programa del reloj

```
1 #include <Adafruit NeoPixel.h>
3 #define PIN
4 #define NUMPIXELS 4
6 using namespace std;
8 Adafruit NeoPixel pixels (NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
    // put your setup code here, to run once:
    pixels.begin();
19evoid loop() {
    // put your main code here, to run repeatedly:
    pixels.clear();
    pixels.setPixelColor(0, pixels.Color(30, 0, 0));
    pixels.setPixelColor(1, pixels.Color(0, 30, 0));
    pixels.setPixelColor(2, pixels.Color(0, 0, 30));
    pixels.setPixelColor(3, pixels.Color(30, 30, 30));
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

- Para poder hacer esto, vamos a aprender a escribir condicionales.
 - Condicional simple
 - Si es cierto... entonces...
 - Condicional doble
 - Si es cierto... entonces.... sino... entonces...
 - Condicional compuesto
 - ☐ Si esto o esto es cierto... entonces...
 - Si esto y esto es cierto... entonces...







5. Encender de uno en uno los led pares en rojo y led impares en verde

Para hacer condicionales **necesitamos operadores** que pueden ser:

Р	!P
True	False
False	True

- ☐ Matemáticos:
 - **-** +, -, *, /, %.
- ☐ Relacionales:
 - □ ==,!=,<, >, >=,<=
- Lógicos
 - **□** &&, and, ||, or, !

Р	Q	P&&Q	P Q
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False





5. Encender de uno en uno los led pares en rojo y led impares en verde

```
Podemos usar un condicional simple usando los operadores ==
```

```
(igual) y != (distinto)
```

```
If (condición){
    sentencias;
```

}

```
== → ¿Son iguales?
!= → ¿Son distintos?
```

```
8 Adafruit NeoPixel pixels(NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
    // put your setup code here, to run once:
    pixels.begin();
19 void loop() {
     int numLed = 2; //El led 2
     *Calcula par o impar. si el resto es 0 es par sino es impar
    int calculo = numLed % 2;
    if(calculo == 0){
      pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
    if(calculo != 0) {
      pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

- Veamos que hace el código:
 - Creo una variable numLed y le asigno el valor 2. (línea 21)
 - ☐ Creo una variable calculo a la que le asigno el resto de dividir numLed entre 2 (para saber si es par o impar) línea 23.
 - ☐ Si el valor de calculo es igual a 0
 - entonces enciende el led numLed de color rojo.
 - Si el valor de calculo es distinto de 0
 - entonces enciende el led numLed de color verde.

```
8 Adafruit NeoPixel pixels(NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
     // put your setup code here, to run once:
     pixels.begin();
16
17 }
19 void loop() {
20
     int numLed = 2; //El led 2
    /*Calcula par o impar. si el resto es 0 es par sino es impar*/
     int calculo = numLed % 2;
24
25₽
     if(calculo == 0){
26
       pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
27
28
29₽
     if(calculo != 0){
30
       pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

- ☐ Veamos que hace el código:
 - Creo una variable numLed y le asigno el valor 2. (línea 21)
 - ☐ Creo una variable calculo a la que le asigno el resto de dividir numLed entre 2 (para saber si es par o impar) línea 23.
 - ☐ Si el valor de calculo es igual a 0
 - entonces enciende el led numLed de color rojo.
 - Si el valor de calculo es distinto de 0
 - entonces enciende el led numLed de color verde.

```
8 Adafruit NeoPixel pixels(NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
     // put your setup code here, to run once:
     pixels.begin();
16
17 }
19 void loop() {
     int numLed = 2; //El led 2
    /*Calcula par o impar. si el resto es 0 es par sino es impar*/
     int calculo = numLed % 2;
     if(calculo == 0){
     pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
     if(calculo != 0) {
      pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
```





- 5. Encender de uno en uno los led pares en rojo y led impares en verde
- ☑ ¿Y si queremos que haga las comprobaciones de manera automática para todos los leds?
 - ¡Usaremos bucles!





BUCLES O CICLOS



```
Bucles
```

```
Loop(){
 Lo que se va a repetir siempre
for (inicio; fin; incremento){
 Lo que se va a repetir desde inicio a fin
while(condición){
lo que quiero que haga mientras se cumpla la condición
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

```
Bucles:
Loop(){
 Lo que se va a repetir siempre
for (inicio; fin; incremento){
 Lo que se va a repetir desde inicio a fin
while(condición){
lo que quiero que haga mientras se cumpla
la condición
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

```
■ Bucles:
```

Loop(){
Lo que se va a repetir siempre
}

```
_ / J
18_
19 void loop() {
20
    int numLed = 2; //El led 2
    /*Calcula par o impar. si el resto es 0 es par sino es impar*/
     int calculo = numLed % 2;
23
24
    if(calculo == 0){
25□
26
      pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
27
     }else{
28
         pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
29
```





5. Encender de uno en uno los led pares en rojo y led impares en verde



- Bucles:
- for (inicio; fin; incremento){
 Lo que se va a repetir desde inicio a fin
 }





5. Encender de uno en uno los led pares en rojo y led impares en verde

```
Bucles:
```

while(condición){
lo que quiero que haga mientras
se cumpla la condición
}





6. Aprendemos el condicional simple y el condicional compuesto

Condicional doble

```
If (condición) {
    sentencia1;
    sentencia1;
    si se cumple condicion
    entonces ejecuta sentencia1
    sino se cumple condición
    sentencia2;
}
```





6. Aprendemos el condicional simple y el condicional compuesto

- Veamos el ejemplo de par o impar, usando un condicional doble.
- → Sería:
 - Si el cálculo es igual a cero,
 - entonces enciende el led2 de color rojo.
 - Sino,
 - entonces enciende el led 2 de color verde.

```
#detine NUMPIXELS 3
 6 using namespace std;
 8 Adafruit NeoPixel pixels (NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
    // put your setup code here, to run once:
15
    pixels.begin();
16
17 }
18
19 void loop() {
20
    int numLed = 2; //El led 2
    /*Calcula par o impar. si el resto es 0 es par sino es impar*/
    int calculo = numLed % 2;
24
25□
    if(calculo == 0){
       pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
     }else{
         pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
```





6. Aprendemos el condicional simple y el condicional compuesto



- Veamos el ejemplo de par o impar, usando un condicional doble.
- → Sería:
 - Si el cálculo es igual a cero,
 - entonces enciende el led2 de color rojo.
 - ☐ Sino,
 - entonces enciende el led 2 de color verde.

```
#detine NUMPIXELS 3
 6 using namespace std;
 8 Adafruit NeoPixel pixels (NUMPIXELS, PIN, NEO GRB + NEO KHZ800);
13 void setup() {
    // put your setup code here, to run once:
15
    pixels.begin();
16
17 }
18
19 void loop() {
    int numLed = 2; //El led 2
    /*Calcula par o impar. si el resto es 0 es par sino es impar*/
    int calculo = numLed % 2;
    if(calculo == 0){
      pixels.setPixelColor(numLed, pixels.Color(30, 0, 0));
  }else{
        pixels.setPixelColor(numLed, pixels.Color(0, 30, 0));
```





5. Encender de uno en uno los led pares en rojo y led impares en verde

Ahora que ya sabes encender los leds, edita el código para que, se encienda de uno en uno los led pero que se apague el anterior.

Te dejo el algoritmo. Piensa como se programa aplicando lo aprendido antes.



Para i=0 mientras i sea menor que el número de pixel incrementa i en 1

función que pone el color al pixel i si i es distinto de 0

Esto hay que hacerlo en el sketch del reloj de arena

función que pone el color al pixel i anterior y poner el color 0 que es apagado aplicar la configuración a los pixel esperar medio segundo





7. Crear funciones.

- ☐ Las funciones son agrupaciones de código
- ☐ Ya conocemos las funciones
 - void setup(){ ... }
 - **□** void loop() { ... }
- Cuando delante del nombre de la función aparece void significa que la función no devuelve nada.
- Las funciones pueden devolver variables.
- Delante del nombre de la función hay que añadir el tipo de variable que devuelve. /
- Esta función es void porque no devuelve nada, aplica la configuración pero no devuelve ninguna variable.



Esto hay que hacerlo en el sketch final donde ampliamos funcionalidad. Las funciones se añaden debajo del loop o encima del setup



LA ALARMA



¿Recuerdas cuando creamos la alarma?

Este era el pseudo código:

Si (hora es igual a alarm_hora Y minutos es igual a alarm_minutos)

limpia la pantalla

pon el cursor en la posición inicial

escribe en la pantalla ¡Despierta!

espera medio segundo

Ahora cuando salte la alarma ¡haz que se encienda la secuencia de luces que más te guste!





PROGRAMANDO LA ALARMA DE LUZ

```
UNIVERSIDAD DE GRANADA
```

```
void loop() {
 // put your main code here, to run repeatedly:
 reloj.updateTime();
 int diadelmes = reloj.dayofmonth ;
 int anio = reloj.year ;
 int hora = reloj.hours ;
 int minuto = reloj.minutes ;
 int segundo = reloj.seconds ;
 String diadelasemana = " ";
 String mes = " ";
 //¿a que hora saltará la alarma?
 int alarm hora = 23;//hora de programación de la alarma
 int alarm minuto = 35; //minutos de programación de la alarma
 mes = DimeMes(reloj.month);
 diadelasemana = DimeDia(reloj.dayofweek);
 lcd.clear();
 //si es la hora, saltará la alarma durante 1 minuto)
  if(hora == alarm hora && minuto == alarm minuto){
     Thunder();
     lcd.clear();
     lcd.setCursor(0, 0);
     lcd.print(";Despierta!");
     delay(500);
 fechahora (diadelasemana, diadelmes, mes, anio, hora, minuto, segundo);
```





¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca_ingeniera

- Aprenderemos a:
 - ☐ Conector el sensor IR.
 - Detectar los códigos de un mando cualquiera.
 - Utilizar los códigos para interactuar con el reloj despertador.







Conectando el hardware del Wake Me UP!



¡Control remoto a distancia!



https://sereingeniera.ugr.es/index.php/2021/07/08/9-de-julio-dia-6-juntemos-todo/

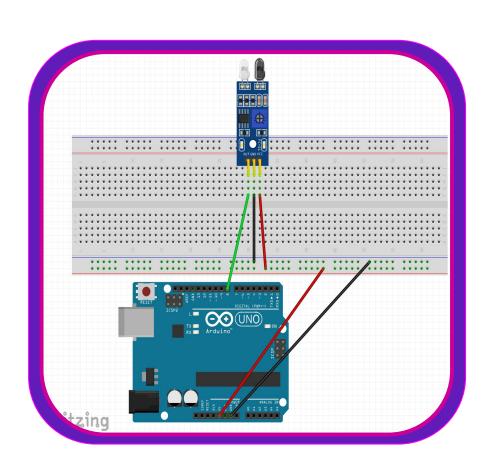


ICONTROL REMOTO!



- ☐ Conectamos el **sensorIR del mando**
 - ☐ El sensor tiene 3 patillas
 - ⊒ s
 - □ VCC
 - **-**
- S va al pin de arduino que enviará la señal pin2
- ☐ VCC va al pin de arduino 5v
- va al pin tierra **GND** de Arduino

OJO mirad bien el nombre de los pines en el componente. Algunos vienen cambiados (podéis usar el polímetro para comprobarlo)







Montaje del Wake ME VP!



¡Control remoto a distancia!









Programar el mando IR

- Incluimos la librería para el mando
- Declaramos la variable donde tenemos conectado el pin del IR
- En el setup habilitamos la recepción del IR.
- ☐ Si se decodifica:
 - ☐ imprimir el hexadecimal
 - imprimir la decodificación
 - imprimir seleccionar—>
 - ☐ imprimir la tecla
 - imprimir —----
 - enviar datos.

```
Mando IR Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda
                                                     to pin 2
  Mando IR
#include <IRremote.hpp>
                                                                 GND
const int RECV PIN = 2;
void setup() {
  Serial.begin(9600);
  //comando para establecer la comunicación con el sensor IR
  IrReceiver.begin (RECV PIN, ENABLE LED FEEDBACK);
void loop() {
  //con este código podemos obtener los valores obtenidos por nuestro mando
  if (IrReceiver.decode()) {
    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);
    Serial.println(IrReceiver.decodedIRData.decodedRawData);
    Serial.println("Selectionar--> ");
    Serial.println(IrReceiver.decodedIRData.command);
    Serial.println("\n-----");
    IrReceiver.resume();
```







Programar el mando IR

- Utilizando el programa anterior:
- Rellenar las variables con el valor que obtengamos al pulsar la tecla del mando.
- ☐ Con el valor que te devuelve por el puerto serie, rellena el valor de las variables (cambiando el 00 por el valor que se muestra en el puerto serie que será un número entero distinto para cada tecla.

```
const int KEY UP = 00;
const int KEY LEFT = 00;
const int KEY OK = 00;
const int KEY RIGHT = 00;
const int KEY DOWN = 00;
const int KEY 1 = 00;
const int KEY 2 = 00;
const int KEY 3 = 00;
const int KEY 4 = 00;
const int KEY 5 = 00;
const int KEY 6 = 00;
const int KEY 7 = 00;
const int KEY 8 = 00;
const int KEY 9 = 00;
const int KEY 0 = 00;
const int KEY ASTERISK = 00;
const int KEY POUND = 00;
```







Programar el mando IR

Una vez tenemos la codificación, vamos a utilizar otra estructura condicional que es switch case.

```
//Según el valor recibido se realiza un bloque u otro
switch( )

{
    case KEY_OK:
        break;
    case KEY_UP:

    Repite los case para el resto de teclas que vayas a utilizar del mando

break;
    case KEY_DOWN:

    break;
}

break;

break;

case KEY_DOWN:
```







Programar el mando IR

- Crea una variable (fuera del setup y del loop) que sea de tipo entero que se llame teclaPulsada.
- Según la estructura de switch case, añade un case para cada tecla que vayas a usar.
- Dentro escribirás lo que quieres que haga cada tecla cuando la pulses.

```
void loop() {
  //leer del mando la tecla pulsada
   if (IrReceiver.decode()) {
      if(IrReceiver.decodedIRData.decodedRawData != 0){
         teclaPulsada = IrReceiver.decodedIRData.command;
       }else{
        teclaPulsada = 0;
      IrReceiver.resume();
  // Esto es como un "if else" ---> if teclapulsada es el switch y
   // cada uno de los case son los else
  switch(teclaPulsada){
      case KEY 1:
     //agui lo que quieres que haga si pulsa la tecla keyl
     break:
     case KEY 2:
     //aqui lo que quieres que haga si pulsa la tecla key2
      break:
     //sique añadiendo case key... para cada una de las teclas
     //del mando que vas a utilizar
```







Programar el mando IR

En mi caso utilizo las funciones que creamos anteriormente y alguna más.

```
switch (teclaPulsada){
 case KEY 1: //muestra el buenos días y la fecha en la pantalla si quieres añade luz
    saludofecha(hora, diadelasemana, diadelmes, mes, anio);
   break:
  case KEY 2: //muestra la fecha y la hora en la pantalla. Aqui no puedes usar luz ni sonido.
   fechahora (diadelasemana, diadelmes, mes, anio, hora, minuto, segundo);
   break:
 case KEY 3: // funcion que muestra a que hora se ha programado la alarma
   programada alarma(alarm hora, alarm minuto);
   break;
  case KEY RIGHT: //apaga las luces de tu lampara.
     ApagarLuz();
 break:
 case KEY LEFT: //enciende las luces de tu lampara.
     Luz Ambiente();
 break;
delay(500):
```





Como se trata de hacer un reloj despertador, podemos añadir en los cases de las teclas las funciones que ya tenemos creadas, o añadir nuevas.

Por ejemplo estas son las funciones que yo he creado:

<pre>programada_alarma(alarm_hora, alarm_minuto);</pre>
Muestra en la LCD a que hora está programada la alarma
<pre>saludofecha(hora, diadelasemana, diadelmes, mes, anio);</pre>
Muestra en la LCD buenos días, tardes o noches dependiendo de la hora y debajo el día.
<pre>fechahora(diadelasemana, diadelmes, mes, anio, hora, minuto, segundo);</pre>
Muestra en la LCD la fecha y en la siguiente línea la hora
ApagarLuz();
Apaga todos los leds del anillo de leds
<pre>Luz_Ambiente();</pre>
Enciende todos los leds de color blanco.

Pero tu puedes crear diferentes funciones con colores para cambiar de color los leds o secuencias.



¿TE ANIMAS A CREAR EL CÓDIGO DE LAS QUE TE FALTAN?



```
switch (teclaPulsada){
 case KEY 1: //muestra el buenos días y la fecha en la pantalla si quieres añade luz
   saludofecha(hora, diadelasemana, diadelmes, mes, anio);
   break;
 case KEY 2: //muestra la fecha y la hora en la pantalla. Aqui no puedes usar luz ni sonido.
   fechahora(diadelasemana, diadelmes, mes, anio, hora, minuto, segundo);
   break;
 case KEY 3: // funcion que muestra a que hora se ha programado la alarma
   programada alarma(alarm hora, alarm minuto);
   break;
 case KEY_RIGHT: //apaga las luces de tu lampara.
     ApagarLuz();
 break:
 case KEY LEFT: //enciende las luces de tu lampara.
     Luz Ambiente();
 break:
```





¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca_ingeniera

- Aprenderemos a
 - Conector el buzzer.
 - Programar diferentes melodías para el despertador.







Montaje del Wake Me UP!



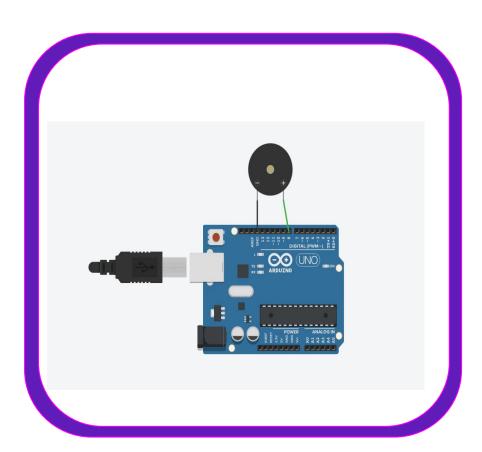
Buzzer para hacer la alarma del despertador







- Usaremos ahora el zumbador(buzzer) para crear una melodía:
- ★ Una de despertador, cuando suene la alarma









- Usaremos ahora el zumbador(buzzer) para crear una melodía:
- ★ Una de despertador, cuando suene la alarma

El buzzer se conecta al pin 10







Programando el Wake Me UP!



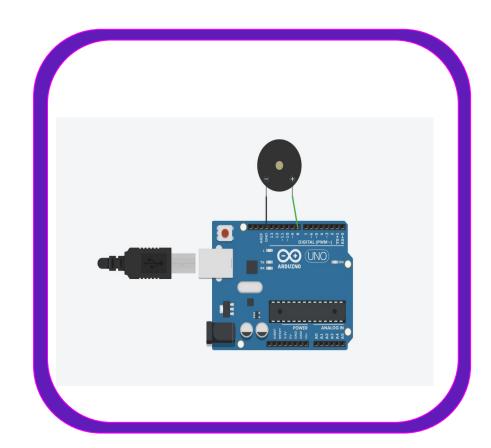
Buzzer para hacer la alarma del despertador





Recordad...

- No podemos usar dos funciones tones() al mismo tiempo.
 Se debe de dejar un pequeño intervalo de tiempo para que suene la nota → delay(time)
 - ★ La frecuencia de sonido va desde 31 Hz a 65535 Hz.









■ Recordad

ì		FRECUENCIA DE LAS NOTAS MUSICALES EN HERCIOS (Hz)								
- 5		OCTAVA 0	OCTAVA 1	OCTAVA 2	OCTAVA 3	OCTAVA 4	OCTAVA 5	OCTAVA 6	OCTAVA 7	OCTAVA 8
Ī	Do	16,3516	32,7032	65,4064	130,813	261,626	523,251	1046,50	2093,00	4186,01
J	Do# / Reb	17,3239	34,6479	69,2957	138,591	277,183	554,365	1108,73	2217,46	4434.92
V	Re	18,3540	36,7081	73,4162	146,832	293,665	587,330	1174,66	2349,32	4698,64
fu ti∈ S∈	Re# / Mib	19,4454	38,8909	77,7817	155,563	311,127	622,254	1244,51	2489,02	4978,04
	Mi	20,6017	41,2035	82,4069	164,814	329,628	659,255	1318,51	2637,02	5274,04
	Fa	21,8268	43,6536	87,3071	174,614	349,228	698,456	1396,91	2793,83	5587,66
)€_	Fa# / Solb	23,1246	46,2493	92,4986	184,997	369,994	739,989	1479,98	2959,96	5919,92
ie-	Sol	24,4997	48,9995	97,9989	195,998	391,995	783,991	1567,98	3135,96	6271,92
16	Sol# / Lab	25,9565	51,9130	103,826	207,652	415,305	830,609	1661,22	3322,44	6644,88
nc I	La	27,5000	55,0000	110,000	220,000	440,000	880,000	1760,00	3520,00	7040,00
	La#/Sib	29,1353	58,2705	116,541	233,082	466,164	932,328	1864,66	3729,31	7458,62
	Si	30,8677	61,7354	123,471	246,942	493,883	987,767	1975,53	3951,07	7902,14

desde 31 Hz a 65535 Hz.









Pasa este pseudocódigo a lenguaje del arduino.

```
Inicializamos buzzer
     void setup() {
       // put your setup code here, to run once:
       buzzer es un pin de salida (OUTPUT)
 8
10
     void loop() {
11
       // put your main code here, to run repeatedly:
12
       suena el buzzer con una frecuencia
13
       dejar x milisegundos
14
       apagar el sonido del buzzer
15
16
```





- *
- Programar un buzzer
- Pasa este pseudocódigo a lenguaje del arduino.

- 1 /*
- 2 Inicializamos buzzer

Pero,...¿Esto cómo se vería en código?

```
tup code here, to run once:
in de <mark>salida</mark> (OUTPUT)
```

```
9
10 void loop() {
11    // put your main code here, to run repeatedly:
12    suena el buzzer con una frecuencia
13    dejar x milisegundos
14    apagar el sonido del buzzer
15  }
16
```



¿CÓMO LO HARÍA?



Se haría algo parecido a como se explica en este código.

- ☐ Veamos el ejemplo de música.
- Primero, inicializamos el buzzer.
 - Se conecta al pin 10.
 - Además, en **void setup()**, se indica en **pinMode(buzzer)**, OUTPUT) que es **OUTPUT**(salida).
 - Después, en el void loop(), indicamos con tone(), la frecuencia que queremos.
 - Después dejamos que suene durante x milisegundos.
 - Y después hacemos que deje de sonar con **noTone()**.

```
1 int buzzer = 9;
 3 void setup() {
     Serial.begin (9600);
     pinMode (buzzer, OUTPUT);
 6 1
 8 void loop() {
     tone (buzzer, 311.13);
     delay(480);
     noTone (buzzer);
     delay(50);
13
14
     tone (buzzer, 311.13);
     delay (260);
     noTone (buzzer);
     delay(50);
18
19
      tone (buzzer, 261.63);
     delay (250);
      noTone (buzzer);
     delay(50);
23
24 }
```



¿CÓMO LO HARÍA?



Se haría algo parecido a como se explica en este código.

- Veamos el ejemplo de música.
- Primero, inicializamos el buzzer.
 - Se conecta al pin 10.
 - Además, en **void setup()**, se indica en **pinMode(buzzer)**, OUTPUT) que es **OUTPUT**(salida).
 - Después, en el void loop(), indicamos con tone(), la frecuencia que queremos.
 - Después dejamos que suene durante x milisegundos.
 - Y después hacemos que deje de sonar con **noTone()**.

```
1 int buzzer = 9;
 3 void setup() {
     Serial.begin(9600);
 pinMode (buzzer, OUTPUT);
 6 1
 8 void loop() {
    tone (buzzer, 311.13);
     delay(480);
     noTone (buzzer);
     delay(50);
     tone (buzzer, 311.13);
     delay (260);
     noTone (buzzer);
     delay(50);
18
19
     tone (buzzer, 261.63);
     delay (250);
      noTone (buzzer);
     delay(50);
22
23
24 }
```





- ★ En el código indicaremos estas instrucciones donde ponemos la condición de sonarAlarma (la melodía de nuestro despertador).
- ★ Tendremos que inventarnos esa melodía, e indicar en el código cuando debería de sonar;)







- ☐ Las funciones son **agrupaciones de código**
- Ya conocemos las funciones
 - uoid setup(){ ... }
 - **□** void loop() { ... }
- Cuando delante del nombre de la función aparece void significa que la función no devuelve nada.
- Las funciones pueden devolver variables.
- Delante del nombre de la función hay que añadir el tipo de variable que devuelve. /
- Esta función es void porque no devuelve nada, ejecuta la música solo.

void despierta(){

METE AQUÍ DENTRO EL CÓDIGO
QUE HAS ESCRITO

Esto hay que hacerlo en el scketch donde vamos a crear el despertador. Las funciones se añaden debajo del setup





Imaginemos que el código de abajo es el de despierta...para decir que debe sonar la alarma, la función despierta que hemos creado, la tenemos que poner aquí.

si es_la_hora, entonces

función despierta

ponemos la variable es_la_hora a false

si hora_alarma es igual hora_actual y minuto_alarm es igual minuto_actual, entonces

ponemos es_la_hora igual a true

encendemos los led con el color que queramos

los enseñamos

esperamos 500ms

limpiamos la configuración

lo enseñamos

sonar alarma





¿QUÉ HAREMOS HOY?

Autoras: Elvira Castillo

Twitter: @layoel

Instagram: eca_ingeniera

- Aprenderemos a
 - Conectar el sensor de color.
 - Programar la detección de colores.







Montaje del Wake Me UP!

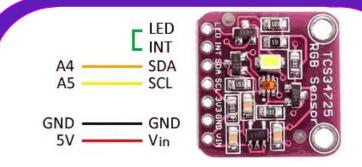


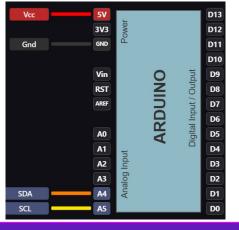
¡Cambiemos de color los leds!





- ☐ Usaremos ahora el sensor de color RGB (*TCS34725*) para cambiar el color de los leds:
- ★ Aunque no vamos a entrar a explicar cómo funciona la comunicación I2C (inter integrated circuits), debemos saber que se trata de un protocolo basado en la existencia de dos líneas de comunicación, una de ellas lleva la señal de reloj y es conocida como (SCL), y la otra línea lleva los datos y es conocida como (SDA).
- ★ Los Pines **SDA** y **SLC** se encuentran especificados en todos los componentes que usan este tipo de protocolo de comunicación, como podemos ver si volvemos a la primera foto de arriba.
- ★ Por tanto, para hacer funcionar nuestro sensor, alimentamos nuestro sensor de color desde Arduino mediante los pines G y 5V y conectamos el pin SDA y SCL del sensor a los 2 pines de nuestro arduino nano SCL y SDA.

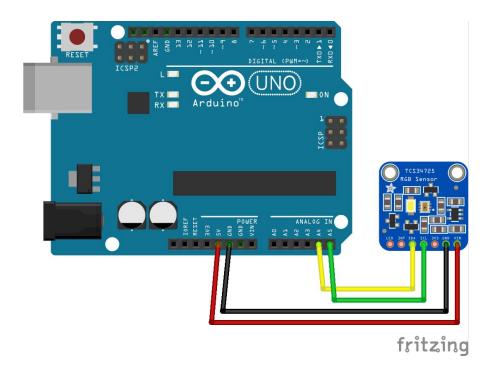








Ahora que sabemos donde conectarlo, seguiremos el esquema siguiente:







Programando el Wake Me UP!



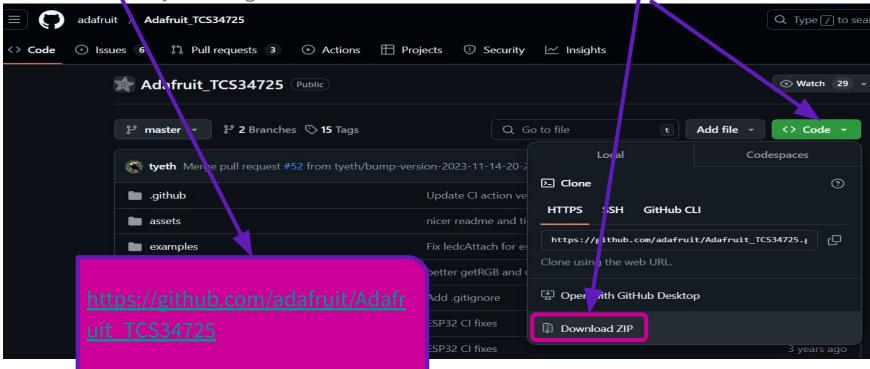
¡Cambiemos de color los leds!





1. Incluir en el IDE la librería Adafruit_TCS34725

Vamos al enlace y descargamos la librería haciendo clic en download







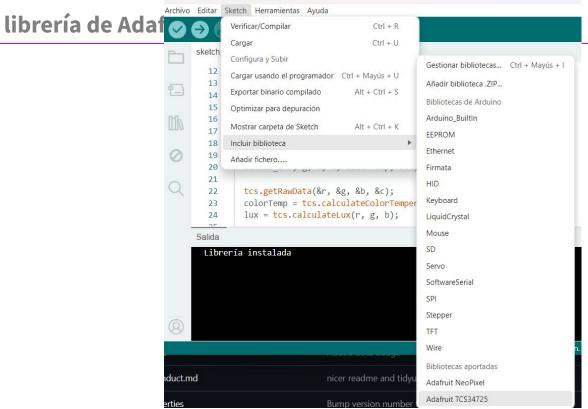
 Incluir en el IDE la librería Adafruit_TCS34725 recuerda cómo lo hicimos con la librería de Adafuit_Neopixel para los leds.

∞ sketch_jun23a Arduino 1.8.19							
Archivo Editar Pro	ograma Herramientas Ayuda						
sketch_jun	Verificar/Compilar Subir Subir Usando Programador Exportar Binarios compilados						
2 //	Mostrar Carpeta de Programa	Ctrl+K	n run once.				
3	Incluir Librería		Administrar Bibliotecas Ctrl+Mayús+I				
4 }	Añadir fichero		Añadir biblioteca .ZIP				
5 6 void l	000 () {		Arduino bibliotecas				





1. Incluir en el IDE sketch feb22a l'Arquino IDE 2.3.4 do fruit TCC 2.472E rocupado cómo lo hicimos con la

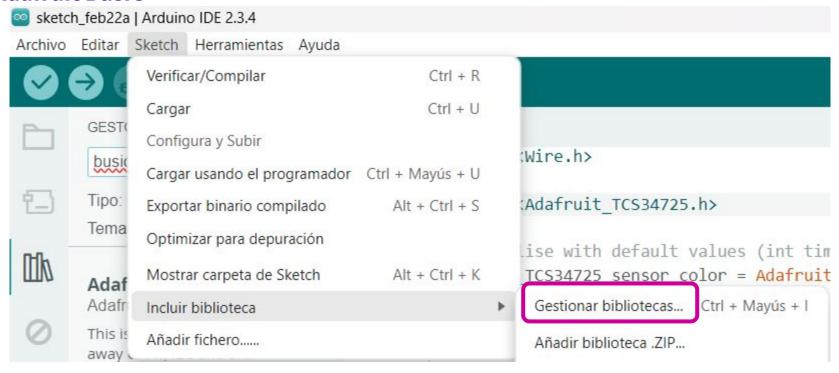






Hay que instalar otra librería más ya que la librería del sensor utiliza I2C y es la librería

Adafruit BusiO

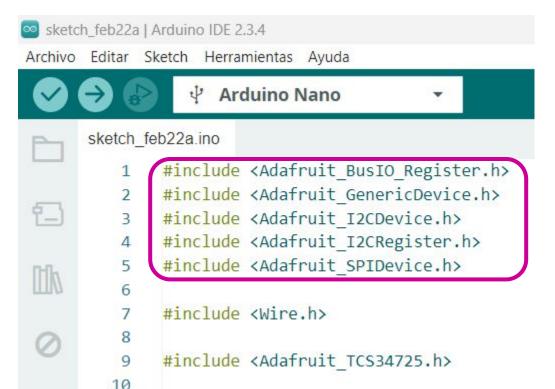






Al incluirla en nuestro scketch veremos que se añaden otras librerías más. Las necesita arduino para saber usar los sensores I2C











- ☐ Vamos a intentar detectar algunos colores con nuestro sensor.
- Comenzaremos con el Rojo, Verde y Azul.
- Después añadiremos alguno más.

Mi programa es colores. Funciona y cambia los colores del led