|  |
| --- |
| Software para Robots Jordán Pascual : pascualjordan@uniovi.es |

|  |  |
| --- | --- |
| 8 | Objetos conectados - Parte 2 |

# Objetos conectados

En esta práctica se presenta como utilizar varios componentes electrónicos y como conectar Arduino a internet para realizar peticiones Web (**Cliente**).

# Ejemplos

## Cliente Ethernet

En este ejemplo vamos a utilizar un sensor de colisión y la placa de Arduino LAN para notificar cada vez que un sensor cambia de valor.

El sistema emula un sensor situado en una plaza de aparcamiento, cuando el vehículo está encima se envía la señal 1 , cuando el vehículo se retira se envía la señal 0.

Cada dispositivo envía a un servidor centralizado: el identificador de la plaza, el estado de la plaza ocupado/libre.

La información solamente se envía cuando la plaza cambia de estado.

|  |  |
| --- | --- |
|  | **Sensor de colisión:** permite alimentaciones entre 3V - 12V.  Al detectar una colisión se enciende el led integrado y registra una señal digital de 0 (De lo contrario cuando no está pulsado envía una señal digital 1). |
| https://www.arduino.cc/en/uploads/Main/ArduinoEthernetShield_R3_Front_450px.jpg | **Arduino Ethernet Shield**: esta placa de extensión permite conectar Arduino a internet mediante un cable de red.  Soporta varios sockets de conexión simultáneos. Para su uso se recomienda utilizar la Ethernet Library. [https://www.arduino.cc/en/Reference/Ethernet](https://www.arduino.cc/en/Reference/Ethernet%20)  Incluye lector de tarjetas micro SD. |

**Construcción del circuito**

1. Conectamos el sensor de colisión a: **GND (a GND), VCC (a 5V) , OUT (a un pin digital).**

**Programa**  
  
Importante: preguntar al profesor la dirección sobre la que tienen que dirigirse las peticiones. Cada grupo tiene asignada una URL.

**Versión 1**, sin conexión. El sensor registra los cambios de estados cada 3000 ms. Es decir, no registra cambios de estado "rápidos", como máximo uno cada 3000ms.

|  |
| --- |
| int pin\_sensor **=** 8**;**  int ultimoValor **=** **-**1**;**  long ultimoValorTime **=** 0**;**  void setup**()** **{**  Serial**.**begin**(**9600**);**  pinMode**(**pin\_sensor**,** INPUT**);**  **}**  void loop**()** **{**  int valorActual **=** digitalRead**(**pin\_sensor**);**    **if** **(**millis**()** **-** ultimoValorTime **>** 3000 **&&** valorActual **!=** ultimoValor**){**  // Cambia de estado, tienen que pasar 3000ms como mínimo    Serial**.**println**(**"Sensor: "**+**String**(**ultimoValor**));**  ultimoValor **=** valorActual**;** // Actualizar ultimo valor  ultimoValorTime **=** millis**();**  **}**  **}** |

**Versión 2:** Enviar petición.  
http://www.mderg.com/robotica/parking**<identificador\_parking>**/actualizarplazag.php?i=**<identificador plaza>**&o=**<estado>**

Ejemplo:  
<http://www.mderg.com/robotica/parking11/actualizarplazag.php?i=1011&o=0>

|  |
| --- |
| #include <SPI.h>  #include <Ethernet.h>  // IMPORTANTE!!!!!CADA UNO QUE PONGA UNA MAC  // ultimo digito segun su grupo 0x21 - 0x22 , 0x22  byte mac**[]** **=** **{**0x54**,** 0x55**,** 0x58**,** 0x10**,** 0x00**,** 0x24**};**  IPAddress dnServer**(**156**,** 35**,** 14 **,** 2**);**  IPAddress gateway**(**192**,** 168**,** 61**,** 13**);**  IPAddress subnet**(**255**,** 255**,** 255**,** 0**);**  // IMPORTANTE!!!!!CADA UNO QUE PONGA UNA IP  // ultimo digito según su grupo 51, 52, 53, 54  IPAddress ip**(**192**,** 168**,** 61**,** 250**);**  char server**[]** **=** "www.mderg.com"**;**  EthernetClient client**;**  int pin\_sensor **=** 8**;**  int ultimoValor **=** **-**1**;**  long ultimoValorTime **=** 0**;**  String identificadorPlaza **=** "1111"**;**  void setup**()** **{**  Serial**.**begin**(**9600**);**  pinMode**(**pin\_sensor**,** INPUT**);**  Ethernet**.**begin**(**mac**,** ip**,** dnServer**,** gateway**,** subnet**);**  Serial**.**println**(**Ethernet**.**localIP**());**  delay**(**1000**);**  **}**  void loop**()** **{**  int valorActual **=** digitalRead**(**pin\_sensor**);**    **if** **(**millis**()** **-** ultimoValorTime **>** 3000 **&&** valorActual **!=** ultimoValor**){**  Serial**.**println**(**"Enviar peticion valor="**+**String**(**valorActual**));**  actualizar**(**identificadorPlaza**,** String**(**valorActual**));**    ultimoValor **=** valorActual**;** // Actualizar ultimo valor  ultimoValorTime **=** millis**();**  **}**  **}**  String respuesta **=** ""**;**  void actualizar**(**String idPlaza**,**String estadoPlaza**){**  **if** **(**client**.**connect**(**server**,** 80**))** **{**  Serial**.**println**(**"conectado"**);**    client**.**println**(**"GET /robotica/parking11/actualizarplazag.php?i="  **+**idPlaza**+**"&o="**+**estadoPlaza**+**" HTTP/1.1"**);**  client**.**println**(**"Host: www.mderg.com"**);**  client**.**println**(**"Connection: close"**);**  client**.**println**();**    delay**(**3000**);** // Esperar por seguridad    **while** **(**client**.**available**())** **{**  char c **=** client**.**read**();**  respuesta**+=**c**;**  **}**  **if** **(!**client**.**connected**())** **{**  Serial**.**println**(**"respuesta -------------------------------"**);**  Serial**.**println**(**respuesta**);**  Serial**.**println**(**"Desconectado ---------------------------"**);**  client**.**stop**();**  **}**    **}** **else** **{**  Serial**.**println**(**"Fallo de conexión"**);**  // RESET a veces falla y hay que reiniciarlo  client**.**stop**();**  delay**(**1000**);**  Ethernet**.** begin**(**mac**,** ip**,** dnServer**,** gateway**,** subnet**);**  delay**(**1000**);**  **}**  **}** |

**Ver datos registrados para el parking 11 (utilizar el parking correspondiente):**

* **HTML**: <http://www.mderg.com/robotica/parking11/plazas.php>
* **Texto**: <http://www.mderg.com/robotica/parking11/plazastexto.php>
* **XML**: <http://www.mderg.com/robotica/parking11/plazasxml.php>

## Cliente Wifi para placa Wemos D1

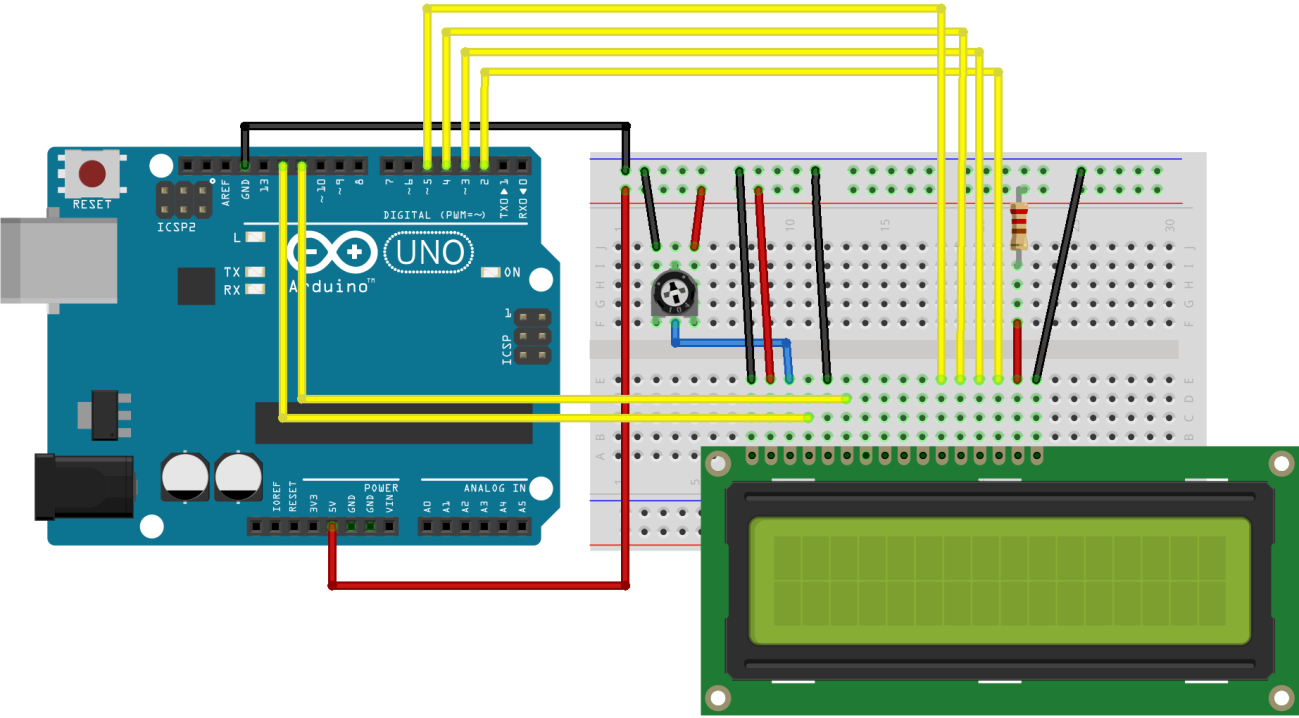
Se debe tener la placa instalada tal y como se indicó en la sesión anterior.

|  |
| --- |
| #include <ESP8266WiFi.h>  const char**\*** ssid **=** "robotica"**;**  const char**\*** password **=** "holahola"**;**  // IMPORTANTE!!!!!CADA UNO QUE PONGA UNA MAC  // ultimo digito segun su grupo 0x21 - 0x22 , 0x22  byte mac**[]** **=** **{**0x54**,** 0x55**,** 0x58**,** 0x10**,** 0x00**,** 0x24**};**  IPAddress dnServer**(**156**,** 35**,** 14 **,** 2**);**  IPAddress gateway**(**192**,** 168**,** 61**,** 13**);**  IPAddress subnet**(**255**,** 255**,** 255**,** 0**);**  // IMPORTANTE!!!!!CADA UNO QUE PONGA UNA IP  // ultimo digito según su grupo 41, 42, 43, 44  IPAddress ip**(**192**,** 168**,** 61**,** 240**);**  char server**[]** **=** "www.mderg.com"**;**  WiFiClient client**;**  int pin\_sensor **=** D7**;**  int ultimoValor **=** **-**1**;**  long ultimoValorTime **=** 0**;**  String identificadorPlaza **=** "2222"**;**  void setup**()** **{**  pinMode**(**pin\_sensor**,** INPUT**);**  Serial**.**begin**(**9600**);**  delay**(**10**);**  // We start by connecting to a WiFi network    Serial**.**println**();**  Serial**.**print**(**"Conectandose a "**);**  Serial**.**println**(**ssid**);**    WiFi**.**config**(**ip**,** gateway**,** subnet**);**  WiFi**.**begin**(**ssid**,** password**);**    **while** **(**WiFi**.**status**()** **!=** WL\_CONNECTED**)** **{**  delay**(**500**);**  Serial**.**print**(**"."**);**  **}**  Serial**.**println**(**""**);**  Serial**.**println**(**"WiFi conectado"**);**  Serial**.**println**(**"Mi IP es: "**);**  Serial**.**println**(**WiFi**.**localIP**());**  **}**  void loop**()** **{**  int valorActual **=** digitalRead**(**pin\_sensor**);**  Serial**.**println**(**"valorActual "**+**String**(**valorActual**)** **+** " ultimoValor "**+**String**(**ultimoValor**)** **);**  **if** **(**millis**()** **-** ultimoValorTime **>** 3000 **&&** valorActual **!=** ultimoValor**){**  Serial**.**println**(**"Enviar peticion valor="**+**String**(**valorActual**));**  actualizar**(**identificadorPlaza**,** String**(**valorActual**));**    ultimoValor **=** valorActual**;** // Actualizar ultimo valor  ultimoValorTime **=** millis**();**  **}**  **}**  String respuesta **=** ""**;**  void actualizar**(**String idPlaza**,**String estadoPlaza**){**  **if** **(**client**.**connect**(**server**,** 80**))** **{**  Serial**.**println**(**"conectado"**);**    client**.**println**(**"GET /robotica/parking11/actualizarplazag.php?i="  **+**idPlaza**+**"&o="**+**estadoPlaza**+**" HTTP/1.1"**);**  client**.**println**(**"Host: www.mderg.com"**);**  client**.**println**(**"Connection: close"**);**  client**.**println**();**    delay**(**3000**);** // Esperar por seguridad    **while** **(**client**.**available**())** **{**  char c **=** client**.**read**();**  respuesta**+=**c**;**  **}**  **if** **(!**client**.**connected**())** **{**  Serial**.**println**(**"respuesta -------------------------------"**);**  Serial**.**println**(**respuesta**);**  Serial**.**println**(**"Desconectado ---------------------------"**);**  client.stop();  }    } else {  Serial.println("Fallo de conexión");  // RESET  client.stop();  delay(1000);  WiFi.config(ip, gateway, subnet);  WiFi.begin(ssid, password);  delay(1000);  }  } |

## Pantalla

En este ejemplo veremos cómo utilizar una pantalla.

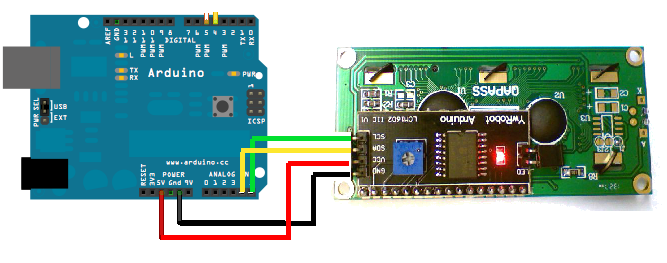
|  |  |
| --- | --- |
| http://mec-s2-p.mlstatic.com/pantalla-lcd-20x4-blackligth-azul-pic-avr-arduino-330301-MEC20304363221_052015-O.jpg | **Pantalla** de 4 líneas HD44780 retro iluminada 16x4 caracteres . |
| LCD-A0A1A2.jpg | **LCD I2C: Inter- Integrated Circuit:** es un bus de comunicación que se utiliza en circuitos integrados.  Cuenta con dos líneas SDA (para datos) y SCL (para reloj)  Se conecta a dos pines analógicos. |

****

Conexión de pantalla

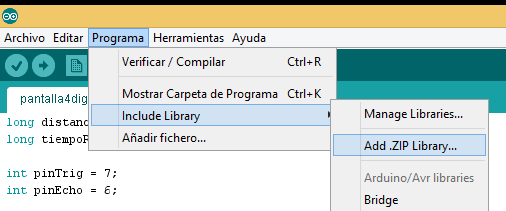
**Construcción del circuito**

* Conexiones con LCD - I2C
  + **VCC** a 5v , **GND** a tierra
  + **SDA** (datos) a A4, **SCL** (Reloj) a A5
  + **Siempre deben ir a esos pines**



**Programa**

Descargar la librería **LiquidCrystal** del campus virtual.



A partir de este momento la librería estará disponible en la lista de librerías : **Programa -> Include Library .**

Usar la librería **Wire** para manejar I2C (**oficial incluida en el entorno**). Referencia: [https://www.arduino.cc/en/Reference/Wire](https://www.arduino.cc/en/Reference/Wire%20)

|  |
| --- |
| **Importante**: dependiendo de la pantalla la dirección del LCD-I2C puede cambiar |

|  |
| --- |
| #include <Wire.h>  #include <LiquidCrystal\_I2C.h> // Descargar la lib liquidcrystal  // el primer código puede ser 0x27 o 0x3F  LiquidCrystal\_I2C lcd**(**0x3F**,** 2**,** 1**,** 0**,** 4**,** 5**,** 6**,** 7**,** 3**,** POSITIVE**);** // Set the LCD I2C address  void setup**()** **{**  Serial**.**begin**(**9600**);**  lcd**.**begin**(**20**,**4**);** // 20 caracteres 4 lineas  // 3 parpadeos al inicio  **for(**int i **=** 0**;** i**<** 3**;** i**++){**  lcd**.**backlight**();**  delay**(**250**);**  lcd**.**noBacklight**();**  delay**(**250**);**  **}**    // Poner luz  lcd**.**backlight**();**  // Colocar un texto  // La posición del cursor comienza en 0  lcd**.**setCursor**(**5**,**0**);**  lcd**.**print**(**"Hola Mundo"**);**  delay**(**1000**);**  lcd**.**setCursor**(**0**,**1**);**  lcd**.**print**(**"Software para Robots"**);**  delay**(**1000**);**  lcd**.**setCursor**(**5**,**2**);**  lcd**.**print**(**"Uniovi 2015"**);**  lcd**.**setCursor**(**1**,**3**);**  delay**(**2000**);**  lcd**.**print**(**"Informatica Oviedo"**);**  **}**  void loop**()** **{**  // Descomentar despues  /\*\*  for (int positionCounter = 0; positionCounter < 16; positionCounter++) {  lcd.scrollDisplayLeft();  delay(600);  }  \*\*/  **}** |

## \*Extensión, pantalla

Consulta los datos del parking y muéstralos en la pantalla

[http://www.mderg.com/robotica/parking11/plazastexto.php](http://www.mderg.com/robotica/parking11/plazastexto.php%20)

Arduino ha de actuar como un cliente y realizar una petición al servicio y mostrar los datos, refrescándolos cada 10 segundos.

|  |
| --- |
| Suponemos que se trata de un panel situado en la ciudad que muestra el estado de las plazas.  En la mayor parte de casos sería más óptimo que el panel con la pantalla sea un **Servidor** y recibirá la información de las plazas de parking solo cuando estas cambian. |

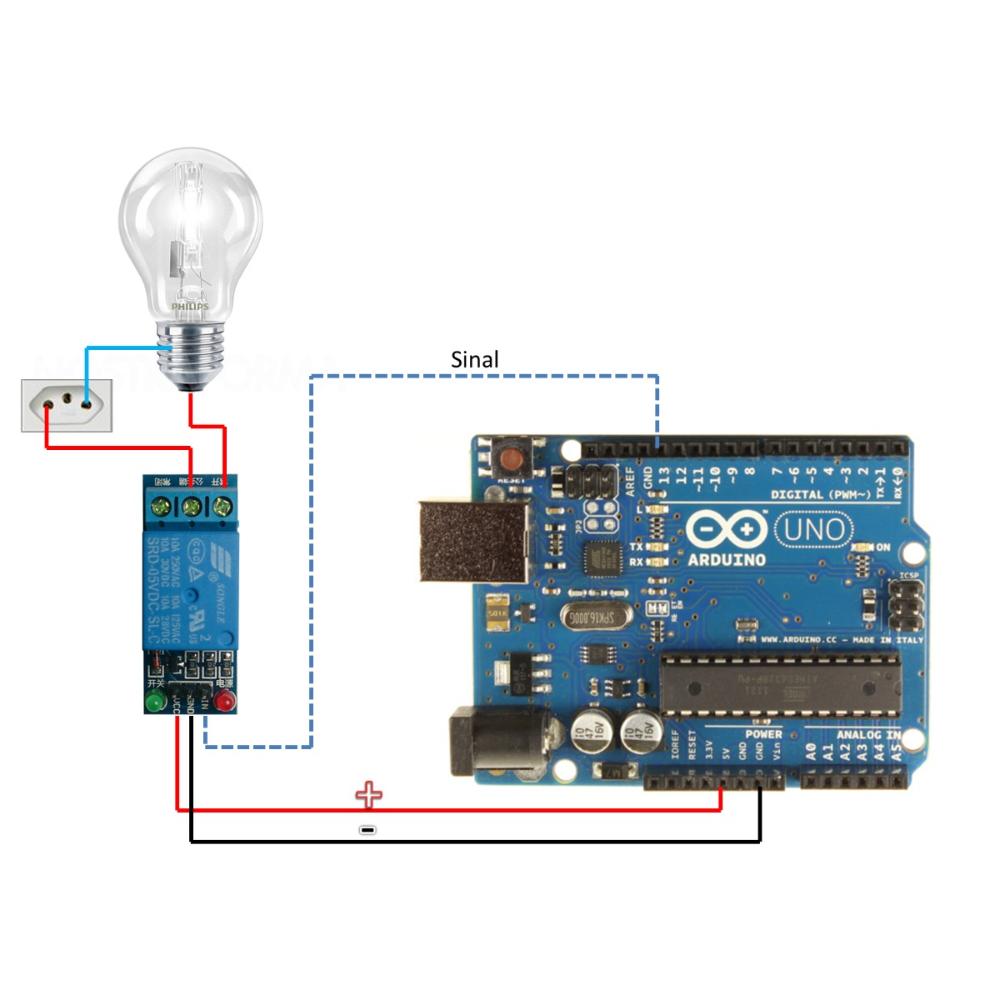
## Relé

En este ejemplo utilizamos un relé. Un relé es un interruptor que podemos activar mediante una señal eléctrica.

Utilizando Arudino es muy sencillo utilizar un relé para encender una luz, un electrodoméstico, la calefacción, etc.

Realmente en está practica nosotros no vamos a encender / apagar ningún componente electrónico, vamos a omitir esa parte, pero sería sencilla.

* Cortamos el cable del dispositivo que tenga que ser alimentado eléctricamente (Por ejemplo, una lámpara) Conectamos la parte del cable cortado que se enchufa a la fuente al punto central del relé (Común).
* Conectamos la parte del cable cortado que está unida la lámpara al relé (parte izquierda o derecha dependiendo de si queremos que por defecto el circuito este abierto o cerrado por defecto).



|  |  |
| --- | --- |
| http://g01.a.alicdn.com/kf/HTB1Qg3CIVXXXXbgXVXXq6xXFXXXC/5-unids-1-canales-5-V-m&oacute;dulo-de-rel&eacute;-de-bajo-nivel-para-SCM-electrodom&eacute;sticos-env&iacute;o.jpg | **Relé**  El relé se conecta a una fuente de alimentación de 5V (Arduino, es el que se encarga de encenderlo/apagarlo)  Las especificaciones del fabricante son las siguientes:  Soporta un máximo de voltaje de 250V AC  Capacidad para carga resistiva: 28VDC@7A, 125VAC@10A, 240VAC@7A  Capacidad para carga inductiva: 120VAC@5A, 28VDC@5A |

**Construcción del circuito:**

Conectamos el pin **In** al **pin digital 7**, el pin **GND** a la **G**, el VCC al **V**, (utilizamos la placa de sensores para conectar el relé).

**Programación de Arduino:**

El relé se maneja de la misma forma que un Led, al escribir la señal HIGH se enciende y al escribir la señal LOW se apaga.

|  |
| --- |
| int pinRele **=** 7**;**  void setup**(){**  pinMode**(**pinRele**,** OUTPUT**)** **;**  **}**  void loop**(){**  digitalWrite**(**pinRele**,** HIGH**);**  delay**(**4000**);**  digitalWrite**(**pinRele**,** LOW**);**  delay**(**4000**);**  **}** |

## Mando IR

En ese ejemplo vamos a utilizar un mando de infrarrojos

|  |  |
| --- | --- |
|  | **Mando y receptor de infrarrojos**.  Alcance corto.  Cada botón del mando genera una señal, que puede ser captada por el receptor.  Transmisión de frecuencia: 38KHz; |

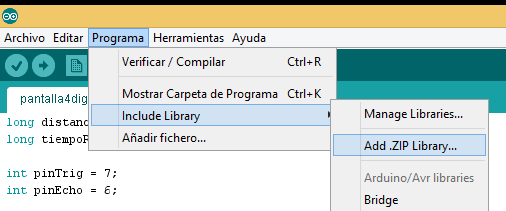
**Construcción del circuito:**

1. **-** a GND, **central** a 5V, **S** a pin 2

**Si usamos la librería siempre debe ir a ese pin**

**Programación de Arduino:**

Para poder usar este modelo lo más sencillo es utilizar la librería **IRLremote.h** descargamos la librería del campus virtual.



A partir de este momento la librería estará disponible en la lista de librerías: **Programa -> Include Library .**

|  |
| --- |
| #include "IRLremote.h" // Importar librería  const int IRInterrumpir **=** 0**;** // Arduino interrupción 0: Pin 2  uint8\_t IRprotocolo **=** 0**;** // Variables para recibir los datos  uint16\_t IRAddress **=** 0**;**  uint32\_t IRComando **=** 0**;**  void setup**()** **{**  Serial**.**begin**(**9600**);**  Serial**.**println**(**"Inicio"**);**  IRLbegin**<**IR\_ALL**>(**IRInterrumpir**);**  **}**  void loop**()** **{**  uint8\_t oldSREG **=** SREG**;** // Parar las interrupciones  cli**();**  **if** **(**IRprotocolo**){** // Si reconoce el protocolo  Serial**.**println**(**IRComando**);;**  IRprotocolo **=** 0**;**  **}**  SREG **=** oldSREG**;**  **}**  void IREvent**(**uint8\_t protocol**,** uint16\_t address**,** uint32\_t command**){**  IRprotocolo **=** protocol**;** // Recogemos los valores y  IRComando **=** command**;**  **}** |