

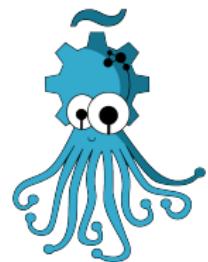
SODAR: Iniciación a Arduino + Processing

Un taller BricoLabs

ctemes eukelade Milo salvati

Asociación BricoLabs

7 noviembre / OSHWDem - 2014



Agenda

1 Presentación

- ¿Quienes somos?
- Requisitos

2 Arduino

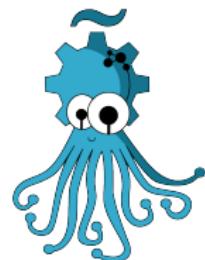
- Intro
- Montaje
- Conceptos CPP

3 SODAR

- Movimiento
- Sensor

4 Processing

- Introducción
- Geometría

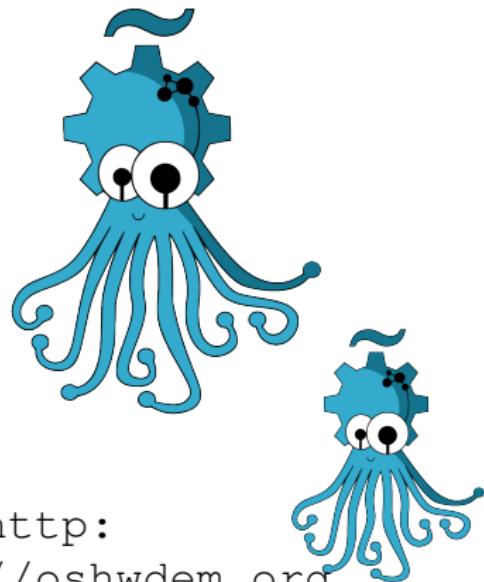


BricoLabs y la OSHWDem



BricoLabs

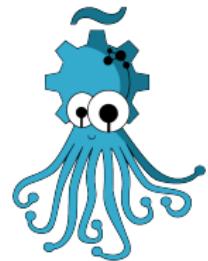
<http://bricolabs.cc/>



<http://oshwdem.org>

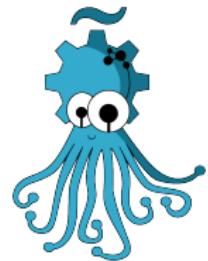
Ponentes

- @ctemes
- @pepdiz
- @Milo_1008
- @salvari

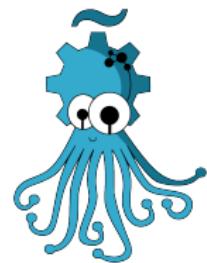
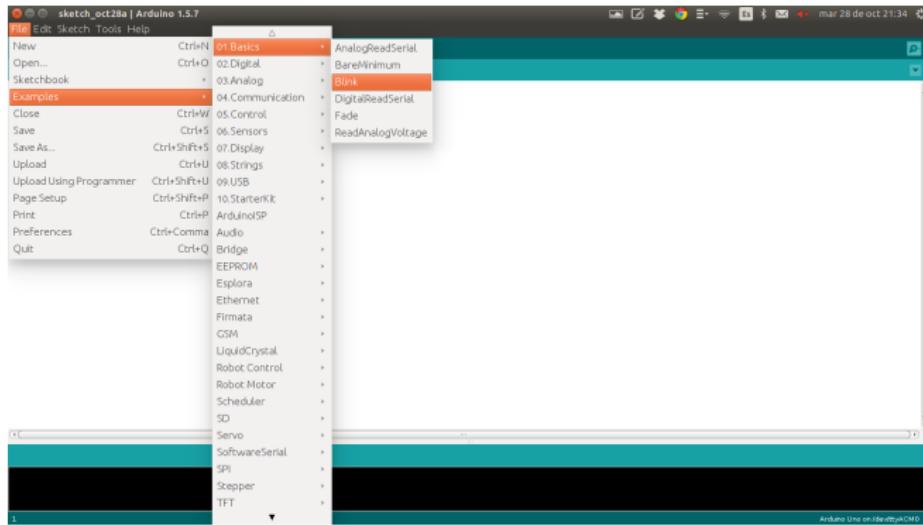


Asistentes

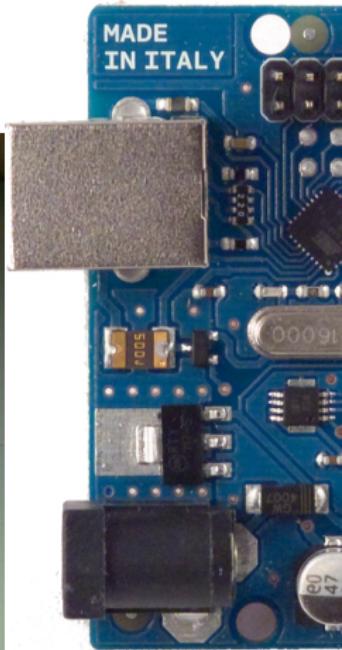
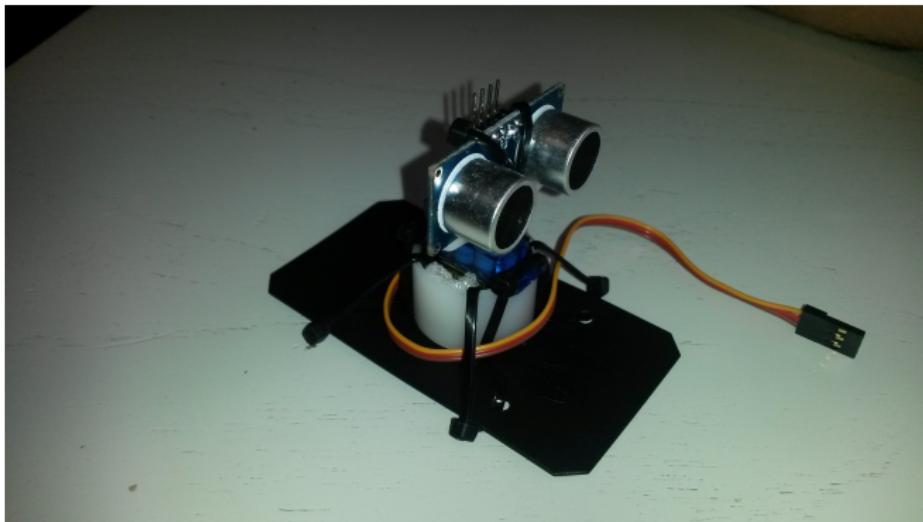
- ¿Quién ha programado antes?
- ¿Quién conoce el Arduino?
- ¿Quién conoce Processing?
- ¿Traéis los deberes hechos? ;)



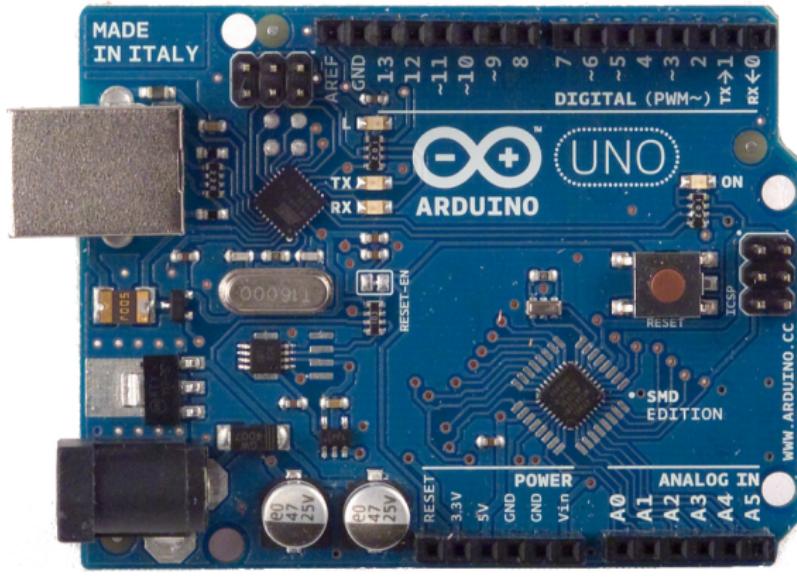
Revisar la instalación



SODAR



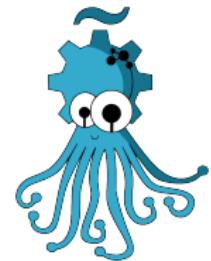
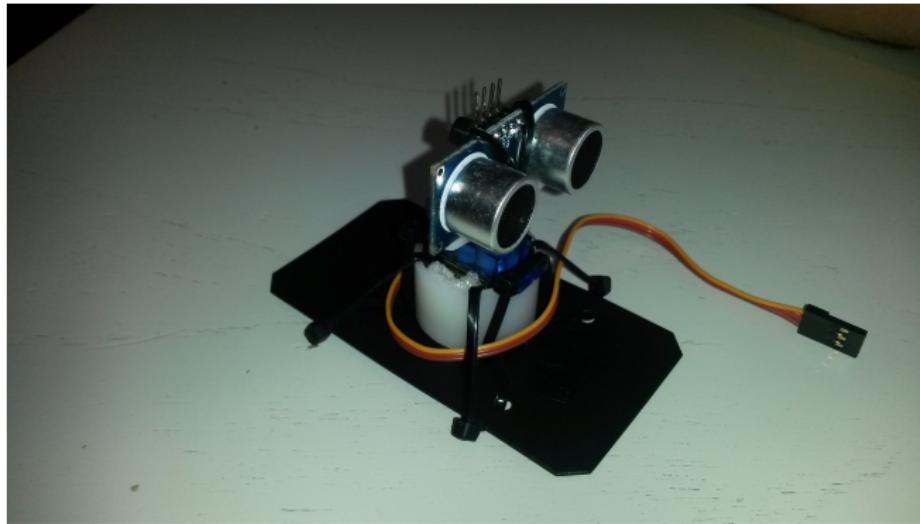
Arduino



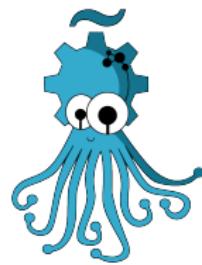
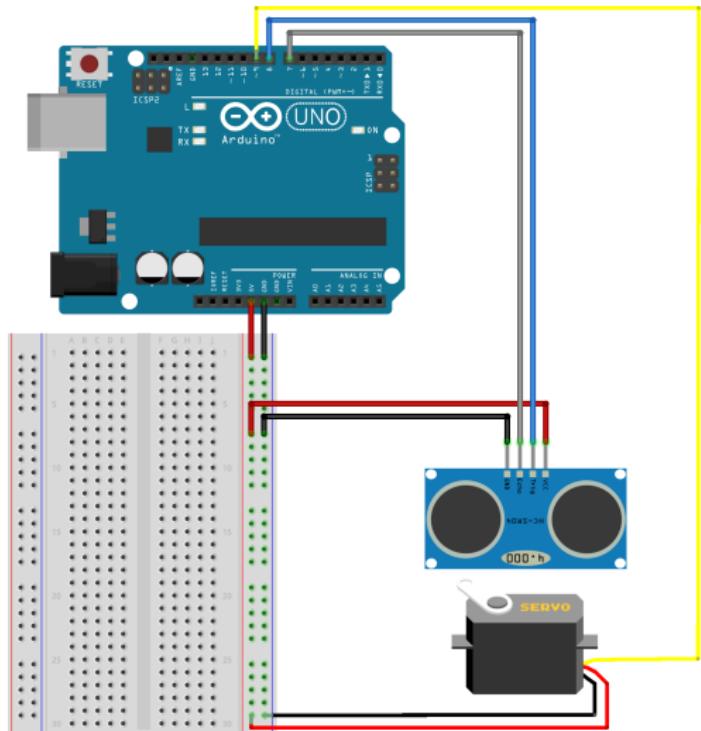
Página Principal

Foto Familia

Montaje I



Montaje II



Estructura de un programa Arduino

```
#include <Servo.h>

#define SERVO_PWM_PIN 9

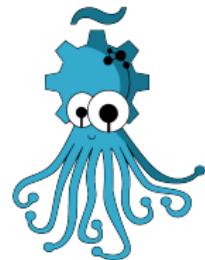
Servo myservo;

/*
  setup
  Se ejecuta una sola vez al principio del programa. O cuando el arduino
  se resetea.
*/
void setup() {

}

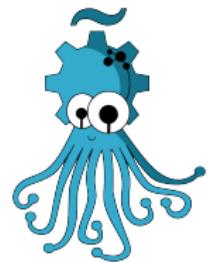
/*
  loop
  Se ejecuta siempre, hasta el fin de los tiempos :-)
*/
void loop() {

}
```



Función

```
int medida(int intentos){  
}  
  
void canta(){  
    Serial.println('La,_la,_la!');  
}
```



Serial

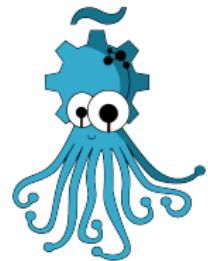
```
Serial.begin(9600);

Serial.print('Hola');      // Sin cambiar de linea

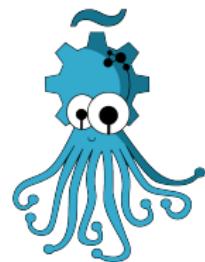
Serial.println('mundo');  // Con retorno de linea

Serial.print(10, DEC);     // Con formato (DEC, BIN, OCT, HEX)

Serial.println(10.1234, 2); // Imprime 10.12
```



Servo

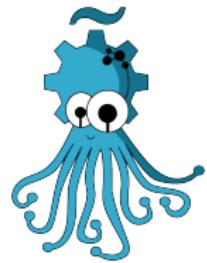


#include <Servo.h>

ctemes, eukelade, milo, salvari

Taller SODAR

Barridos



Una solución

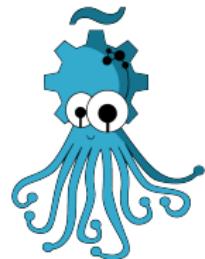
```
#define ANGULO_MIN 0      // angulo minimo del barrido (en grados)
#define ANGULO_MAX 180     // angulo maximo del barrido (en grados)
#define PASO_ANGULO 1      // paso de barrido

#define ATRAS -1
#define ADELANTE 1

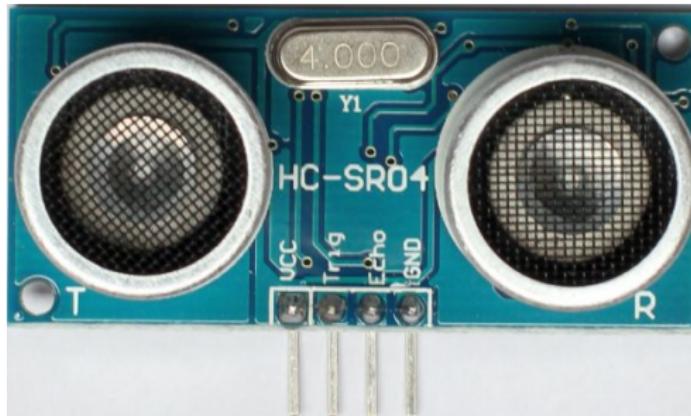
loop() {
    delay(50);           // espera 50 milisegundos
    miservo.write(angulo); // avanza el servomotor al angulo indicado

    if (angulo >= ANGULO_MAX) { direccion = ATRAS; }
    if (angulo <= ANGULO_MIN) { direccion = ADELANTE; }

    angulo += direccion * PASO_ANGULO; // incrementa el angulo de
                                      // barrido un paso de barrido
}
```

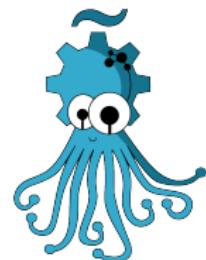


Sensor ultrasonidos



Sensor de distancia HC-SR04
Electric Parameter

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm



Protocolo

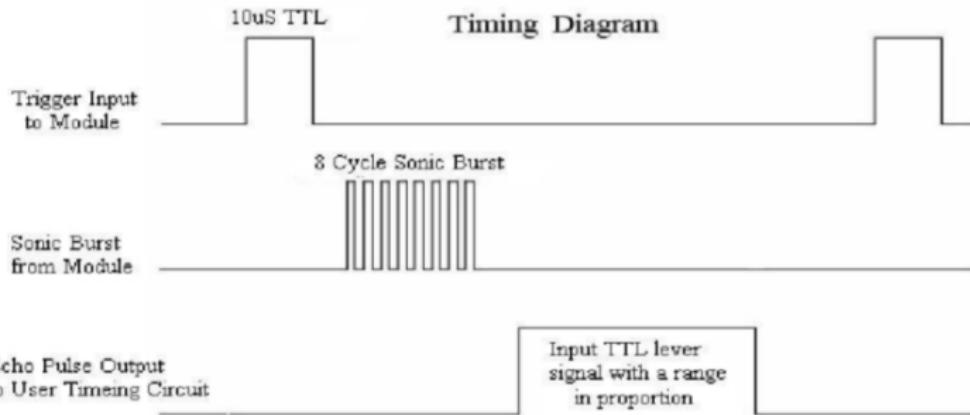
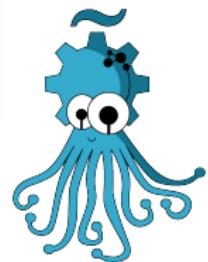


Diagrama de señales



Función Medida

Una función de bajo nivel

```
/*
 *-----*
 * setup
 * Se ejecuta una sola vez al principio del programa. O cuando el arduino
 * se resetea.
 *-----*/
void setup() {
    pinMode(TRIGGER_PIN, OUTPUT); // pin trigger es salida
    pinMode(ECHO_PIN, INPUT); // pin echo es entrada

    Serial.begin(9600); // Abrimos el puerto serie
}

/*
 *-----*
 * loop
 * Se ejecuta siempre repetidamente, hasta el fin de los tiempos :-)
 *-----*/
void loop() {
    // Preparamos el sensor
    digitalWrite(TRIGGER_PIN, LOW); // Nivel bajo para estabilizar
    delayMicroseconds(5); // garantizamos 5 microsegundos

    digitalWrite(TRIGGER_PIN, HIGH); // Enviamos un pulso de 10 microsegundos
    delayMicroseconds(10);
    digitalWrite(TRIGGER_PIN, LOW);

    time = pulseIn(ECHO_PIN, HIGH); // Leemos el echo viene codificado en
} //temes, eukelade, milo, salvari
```

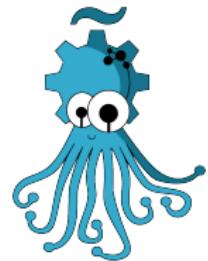


NewPing

```
#include <NewPing.h>

NewPing sonar(TRIGGER_PIN,
              ECHO_PIN,
              DISTANCIA_MAXIMA); // Creamos un objeto sensor

int cm = sonar.ping_cm(); // Medimos
```



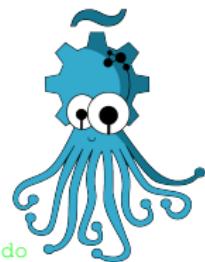
Solución SODAR

```
/*
 *-----*
 obtenerDistanciaEnviar
 esta funcion obtiene la distancia a la que se encuentra un objeto
 y envia por el puerto serie el angulo y la distancia del objeto encontrado
 -----*/
int obtenerDistanciaEnviar(int angulo) {
    int cm = sonar.ping_cm();           // obtiene la distancia en cm
    Serial.print(angulo, DEC);          // envia por puerto serie la distancia y el angulo
    Serial.print(",");                // formato:
    Serial.println(cm, DEC);           // angulo,distancia<FINLINEA>
}

#define ANGULO_MIN 0      // angulo minimo del barrido (en grados)
#define ANGULO_MAX 180   // angulo maximo del barrido (en grados)
#define PASO_ANGULO 1    // paso de barrido

#define ATRAS -1
#define ADELANTE 1

loop(){
    delay(50);                      // espera 50 milisegundos
    miservo.write(angulo);           // avanza el servomotor al angulo indicado
    obtenerDistanciaEnviar(angulo);  // obtiene la distancia y envia
                                    // los datos por el puerto serie
}
```



Processing



Estructura de un programa

```
import processing.serial.*;

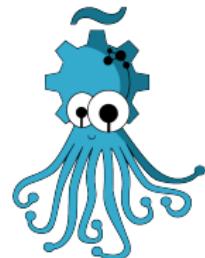
// ---- variables ----

int numPuntos=0;
String contenidoSerie;
Serial miPuerto;

/*
  setup
  Se ejecuta una sola vez al principio del programa
  -----
void setup() {
  background(255);
  size(MAXX, MAXY);
}

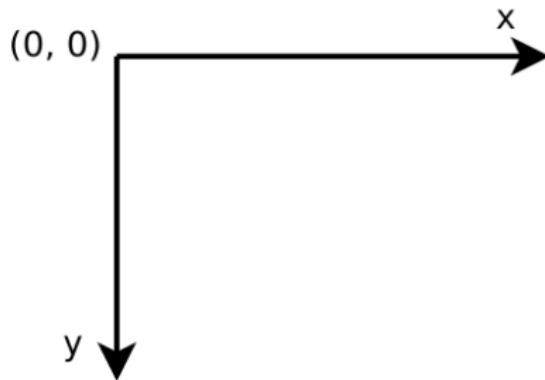
/*
  draw
  Se ejecuta eternamente hasta el fin de los tiempos
  -----
void draw() {

}
```

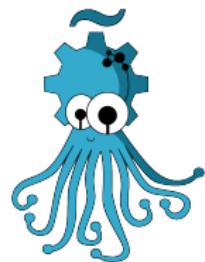


Pantalla de Processing

Coordenadas

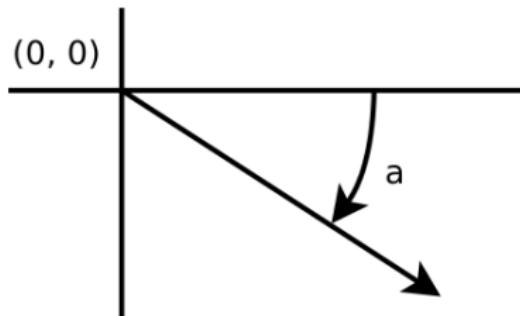


Coordenadas en Processing



Pantalla de Processing

Ángulos



$$\beta = 2\pi - \text{radians}(\alpha)$$

