

---

## Servomotores

---

Un servomotor es un dispositivo alimentado por corriente continua que puede controlar de modo muy exacto la posición (de 0° a 180°) o la velocidad (en revoluciones por minuto, rpm, en sentido horario o antihorario). Tienen 3 pines para su conexión: alimentación (5 V, normalmente), tierra (GND) y el pin de la señal. Por este último, a través del sistema de control le emitirá la señal PWM que le indicará al servomotor la posición o la velocidad que debe alcanzar, según el tipo de servomotor usado.

Suelen llevar tener un mecanismo reductor que les proporciona muy buen par y muy buena precisión ya que ellos mismos realizan de forma interna el control de precisión, siendo en otros motores necesario de forma externa. Sin embargo, proporcionan menos velocidad que los motores de corriente continua.

Estas características los hacen ideales para el control de aplicaciones de robótica como brazos robóticos u orientación de sensores o torretas.

Habitualmente hay 2 tipos:

Servomotores de rango de giro limitado que son los más usados, cuyo ángulo va de 0 a 180°.

Servomotores de rotación continua, los cuales giran 360° y se puede controlar tanto su giro como su velocidad.

---

## Funcionamiento de los servomotores

---

Los servos son motores de corriente continua con reductor acoplado para reducir la velocidad de giro, y mediante electrónica controlar su posición. El control de la posición se lleva a cabo mediante la transmisión de una señal de pulso modulada (PWM). Para entenderlo mejor puede consultar el artículo sobre PWM.

En resumen, según el ancho de pulso positivo (duración) que se le envíe al servo, se determinará una posición. Esta duración es proporcional al ángulo de giro del motor, siendo dependiendo del modelo:

Un pulso entre 500-1000 us corresponde con 0°.

Un pulso de 1500 ms corresponde con 90° (punto neutro).

Un pulso entre 2000-2500us corresponde con 180°.

---

---

## Modelos populares de servomotores

---

### SG90



Este es un “micro” servo estándar muy usado en todo tipo de aplicaciones, ya que es barato y ligero, cuyos engranajes son de plástico.

Las características del servo son:

Voltaje de funcionamiento: 3,0-7,2V

Velocidad (4.8V sin carga): 0.12sec/60 degrees

Fuerza: 1.2kg / 42.3oz(4.8V);1.6 kg / 56.4oz (6.0V)

Temperatura de trabajo: -30 to +60 grados °C

Longitud de cable: aprox. 23cm

Incluye accesorios de la foto y tornillos

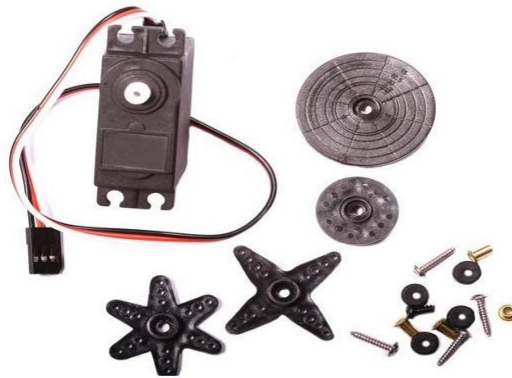
Dimensiones: 22x11.5x27mm

Tipos:

Servomotor 9G SG90 Servo Giro 360 grados Giro continuo

Servomotor 9g SG90 Servo Giro de 0 a 180°

**S3003**



Este servomotor es el comúnmente usado de tamaño “grande” y, como el anterior, también es ampliamente usado en aplicaciones. Muchos componentes como brazos robóticos o hexápodos vienen preparados para la instalación de este servo. Es perfecto para proyectos de Arduino, y aeromodelaje.

Las características del servo son:

Voltaje de operación: 4.8 V a 6 V

Torque máx...: 4,1kg/cm (6V)

Velocidad de funcionamiento: 0.23sec/60grados(4.8V), 0.19sec/60grados(6V)

Radio máx...: 360°

Rango de temperatura: -20 + 60 C grados

Peso: 38 g

Longitud de cable: 26cm

Dimensiones: 40,5 x39x 20 mm

Conector del servo: JR

Fabricación china

Tipos:

[Servomotor Tipo S3003 Servo Giro 360 Grados](#) Giro continuo

[Servomotor Tipo S3003 Servo Giro 180 Grados](#) Amplitud giro 180°

[Servomotor Tipo S3003 Servo Giro 90 Grados](#) Amplitud giro 90°

---

## Arduino

---

Estos componentes son muy útiles para muchas aplicaciones, por lo que son ampliamente usados en proyectos con Arduino debido a su facilidad. Arduino viene con una librería para su uso llamada "servo.h", con la cual puedes controlar hasta 12 servos usando Arduino UNO/Nano o incluso hasta 24 usando Arduino MEGA. Los servos se alimentan con los propios pines de 5V de la placa, y sin carga pueden funcionar bastantes motores. Nosotros recomendamos alimentar la placa de forma externa, ya que por USB podría tener más fallos o, alimentar los servos con una fuente externa que les proporcione una corriente suficiente para que incluso funcionen varios con carga. Hay que recordar que, si se usa alimentación externa, hay que unir las masas (GND).

### Ejemplos del código

En el IDE de Arduino viene por defecto dos ejemplos, Knob y Sweep, pero primero vamos a poner un ejemplo más sencillo y visual para entender su funcionamiento:

```
// Incluimos la librería para poder controlar el servo

#include

// Declaramos la variable para controlar el servo

Servo myservo;

void setup() {

    // Iniciamos el servo para que se asigne su control con el pin 9

    myservo.attach(9);

}

void loop() {

    // Desplazamos a la posición 0°

    myservo.write(0);

    // Esperamos 1 segundo

    delay(1000);

    // Desplazamos a la posición 90°

    myservo.write(90);
```

```
//Esperamos 1 segundo
delay(1000);

// Desplazamos a la posición 180°
myservo.write(180);

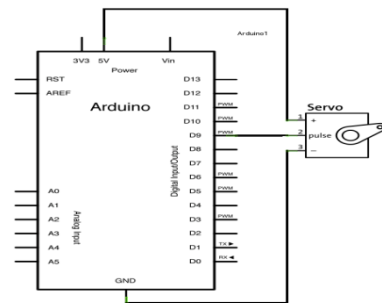
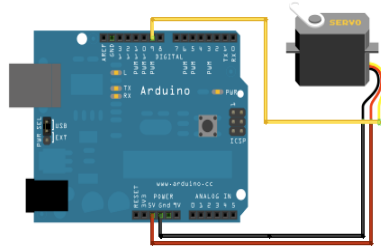
// Esperamos 1 segundo
delay(1000);
}
```

Con este código observamos que primero el servo hace girar las aspas hacia en sentido antihorario (CCW, para 0°), después se desplaza hasta los 90° y por último gira en sentido horario hasta los 180°.

## Sweep

Ahora vamos a ver el sketch **“Sweep”**, cuya traducción es barrido, entonces como su nombre indica, las aspas del servo irán haciendo un barrido por cada ángulo hasta llegar al indicado.

Conectamos un servo de giro continuo (360°) a la placa Arduino MEGA. Para ello, cargamos el Sketch de ejemplo (Archivo>Ejemplos>Servo>Sweep) y conectamos los componentes del siguiente modo:



El código del sketch es el siguiente:

```
Sweep Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

Sweep $
/* Sweep
by BARRAGAN <http://barraganstudio.com>
This example code is in the public domain.

modified 8 Nov 2013
by Scott Fitzgerald
http://www.arduino.cc/en/Tutorial/Sweep
*/

#include <Servo.h>

Servo myservo; // crea un nuevo objeto de la clase servo para controlar el servo
// hasta 12 objetos servo se pueden crear en la mayoría de las placas

int pos = 0; // variable que almacena la posición del servo

void setup() {
  myservo.attach(9); // conecta el servo en el pin 9 al objeto servo
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // va de 0 grados a 180 grados
    // in steps of 1 degree
    myservo.write(pos); // dice al servo que vaya a la posición de la variable 'pos'
    delay(15); // espera 15 ms para que el servo alcance la posición
  }
  for (pos = 180; pos >= 0; pos -= 1) { // va de 180 grados a 0 grados
    myservo.write(pos); // dice al servo que vaya a la posición en la variable 'pos'
    delay(15); // espera 15 ms para que el servo alcance la posición
  }
}
```

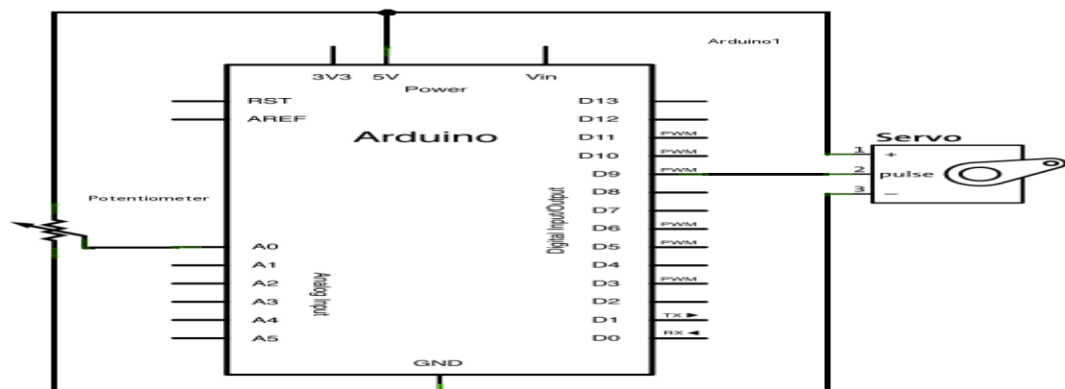
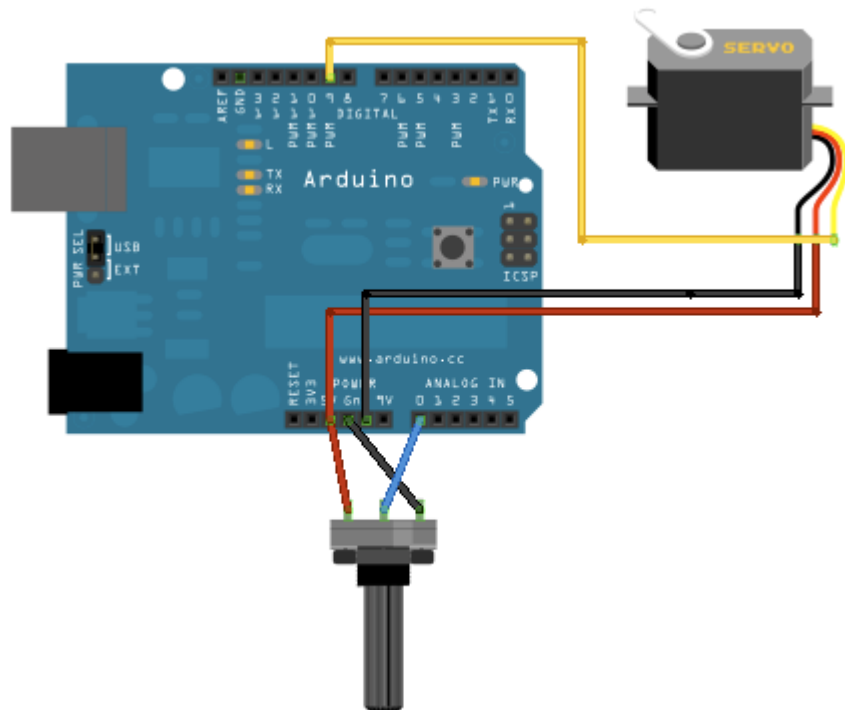
Podemos observar cómo va incrementando el ángulo de giro cada 15ms un grado cada vez. Esto se puede ver en los bucles “for” donde el primero va de 0 a 180°, y el segundo desde los 180 a 0°.

Ahora prueba a aumentar el tiempo, cambiando el 15 por 150 por ejemplo. De este modo observará más claramente cómo funcionan los servos, donde su giro comienza lentamente hasta que coge velocidad y luego esta va reduciéndose hasta que llega a la posición deseada.

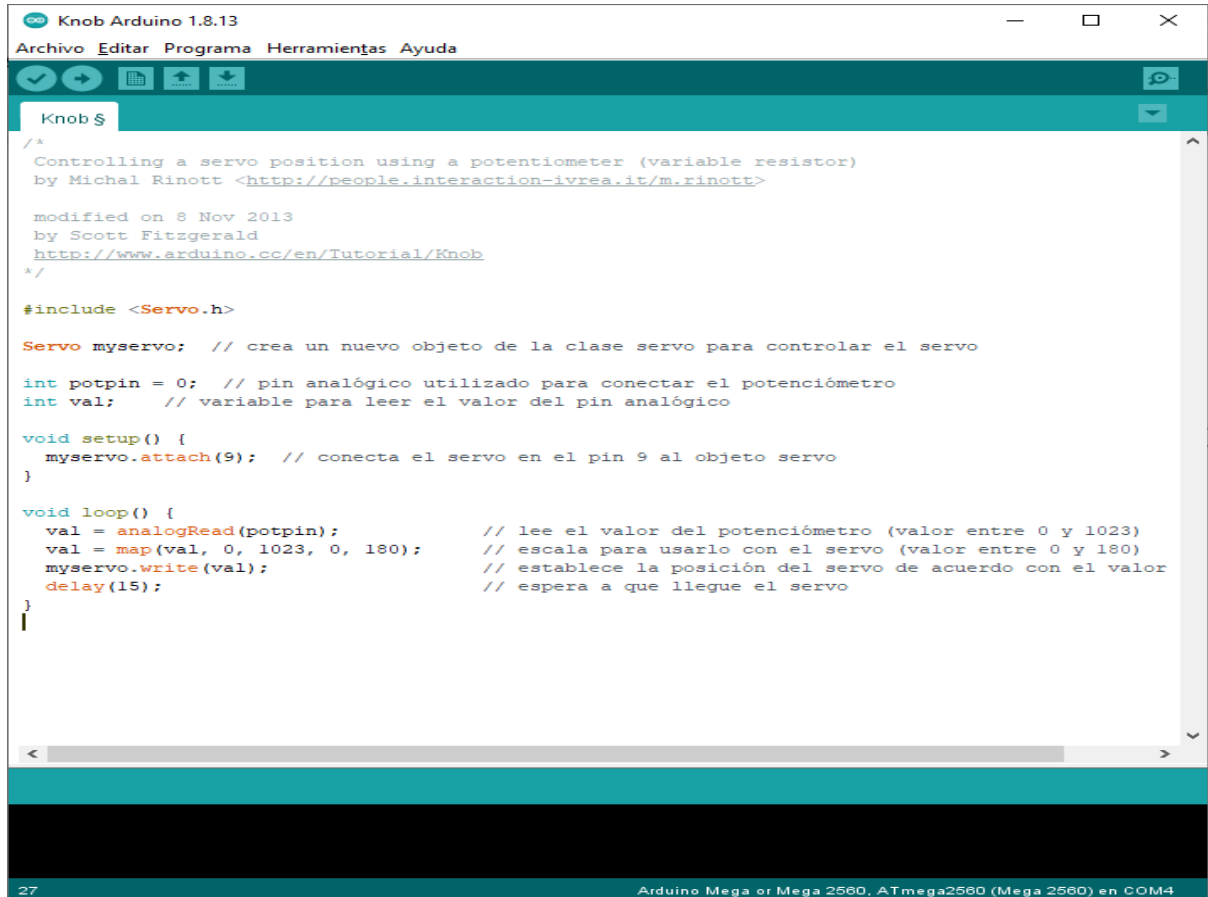
## Knob

Para el uso del sketch **Knob**, se necesitará un potenciómetro con el que regularemos la posición del servo.

Para ello abrimos el sketch correspondiente (Archivo>Ejemplos>Servo>Knob) y conectamos los componentes del siguiente modo:



El código del sketch es el siguiente:



```
Knob Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

Knob $

/*
Controlling a servo position using a potentiometer (variable resistor)
by Michal Rinott <http://people.interaction-ivrea.it/m.rinott>

modified on 8 Nov 2013
by Scott Fitzgerald
http://www.arduino.cc/en/Tutorial/Knob
*/

#include <Servo.h>

Servo myservo;  // crea un nuevo objeto de la clase servo para controlar el servo

int potpin = 0;  // pin analógico utilizado para conectar el potenciómetro
int val;         // variable para leer el valor del pin analógico

void setup() {
  myservo.attach(9);  // conecta el servo en el pin 9 al objeto servo
}

void loop() {
  val = analogRead(potpin);          // lee el valor del potenciómetro (valor entre 0 y 1023)
  val = map(val, 0, 1023, 0, 180);    // escala para usarlo con el servo (valor entre 0 y 180)
  myservo.write(val);                // establece la posición del servo de acuerdo con el valor
  delay(15);                         // espera a que llegue el servo
}
```

27 Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) en COM4

Lo curioso de este sketch, es que, si usamos un servo de giro continuo, podemos hacer cambiar el sentido de giro en función del sentido del potenciómetro. De hecho, podemos ajustar que el sentido de giro coincida, ya que con solo intercambiando la polarización del pin positivo y negativo del potenciómetro lo modificamos. Además, en un punto concreto del potenciómetro podemos parar el servo.

Por otro lado, si usamos un servo de 180°, el potenciómetro sirve para guiar el movimiento de las aspas a la par que lo giras.