```
ï¿½/*SINGLE SERVO
Sweep a servo back and forth through its full range of motion.
A "servo", short for servomotor, is a motor that includes
feedback circuitry that allows it to be commanded to move to
specific positions. This one is very small, but larger servos
are used extensively in robotics to control mechanical arms,
hands, etc. You could use it to make a (tiny) robot arm,
aircraft control surface, or anywhere something needs to be
moved to specific positions.*/

// If we had to write a sketch to control a servo from scratch,
// it would be a lot of work. Fortunately, others have done the
// hard work for you. We're going to include a "library"
// that has the functions needed to drive servos.
// A library is an set of additional functions you can add to
// your sketch. Numerous libraries are available for many uses,
// see http://arduino.cc/en/Reference/Libraries for information
// on the standard libraries, and Google for others. When you're
// using a new part, chances are someone has written a library
// for it.

#include <Servo.h> // servo library
// Once you "include" a library, you'll have access to those
// functions. You can find a list of the functions in the servo
// library at: http://arduino.cc/en/Reference/Servo
// Most libraries also have example sketches you can load from
// the "file/examples" menu.
// Now we'll create a servo "object", called myservo. You should
// create one of these for each servo you want to control.
// You can control a maximum of twelve servos on the Uno
// using this library. (Other servo libraries may let you
// control more). Note that this library disables PWM on
// pins 9 and 10!

Servo myservo; // servo control object
void setup()
{
  // We'll now "attach" the servo1 object to digital pin 9.
  // If you want to control more than one servo, attach more
  // servo objects to the desired pins (must be digital).
  // Attach tells the Arduino to begin sending control signals
  // to the servo. Servos require a continuous stream of control
  // signals, even if you're not currently moving them.
  // While the servo is being controlled, it will hold its
  // current position with some force. If you ever want to
  // release the servo (allowing it to be turned by hand),
  // you can call myservo.detach().
  myservo.attach(9);
}

void loop() {
  int position;
  // To control a servo, you give it the angle you'd like it
  // to turn to. Servos cannot turn a full 360 degrees, but you
  // can tell it to move anywhere between 0 and 180 degrees.
  // Change position at full speed:
myservo.write(90);
delay(1000);
myservo.write(180);
delay(1000);
```

```
myservo.write(0);
delay(1000);
 // Tell servo to go to 90 degrees
// Pause to get it time to move
 // Tell servo to go to 180 degrees
// Pause to get it time to move
 // Tell servo to go to 0 degrees
// Pause to get it time to move
  // Change position at a slower speed:
  // To slow down the servo's motion, we'll use a for() loop
  // to give it a bunch of intermediate positions, with 20ms
  // delays between them. You can change the step size to make
  // the servo slow down or speed up. Note that the servo can't
  // move faster than its full speed, and you won't be able
  // to update it any faster than every 20ms.
  // Tell servo to go to 180 degrees, stepping by two degrees
  for(position = 0; position < 180; position += 2)
  {
    myservo.write(position);  // Move to next position
    delay(20);               // Short pause to allow it to move
  }
  // Tell servo to go to 0 degrees, stepping by one degree
  for(position = 180; position >= 0; position -= 1)
  {
    myservo.write(position);  // Move to next position
    delay(20);               // Short pause to allow it to move
  }
}
```