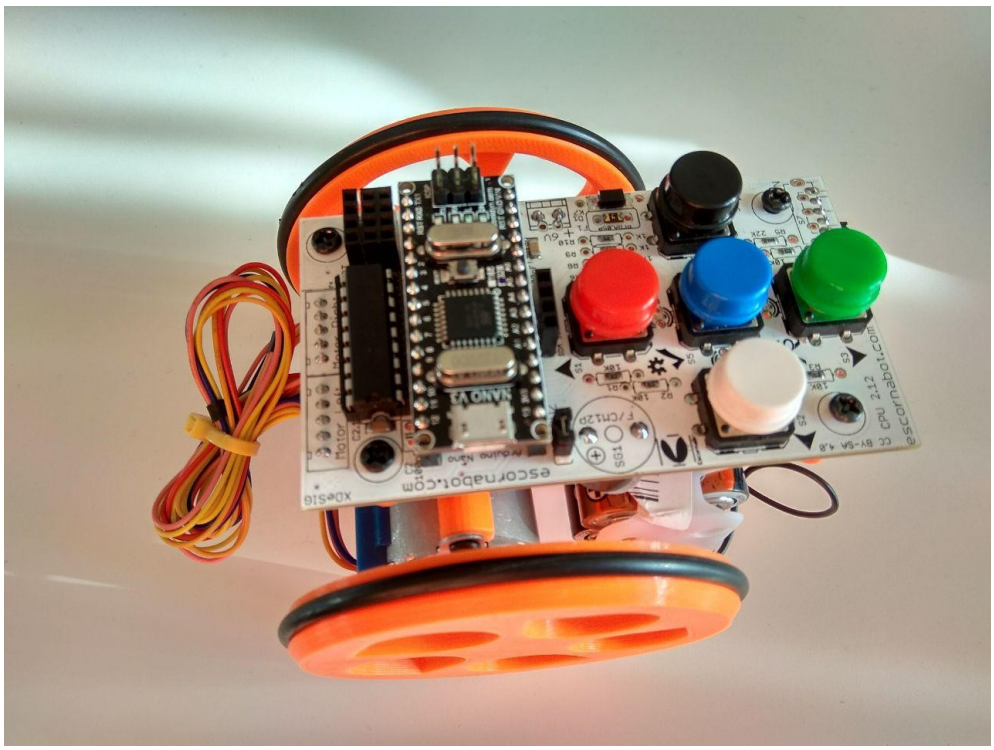
MANUAL DE USO DE LIBRERÍA DE ARDUINO PARA ESCORNABOT



Escornabot con CPU 2.12

Documento creado por: *Pedro Ruiz Fernández*

ÍNDICE

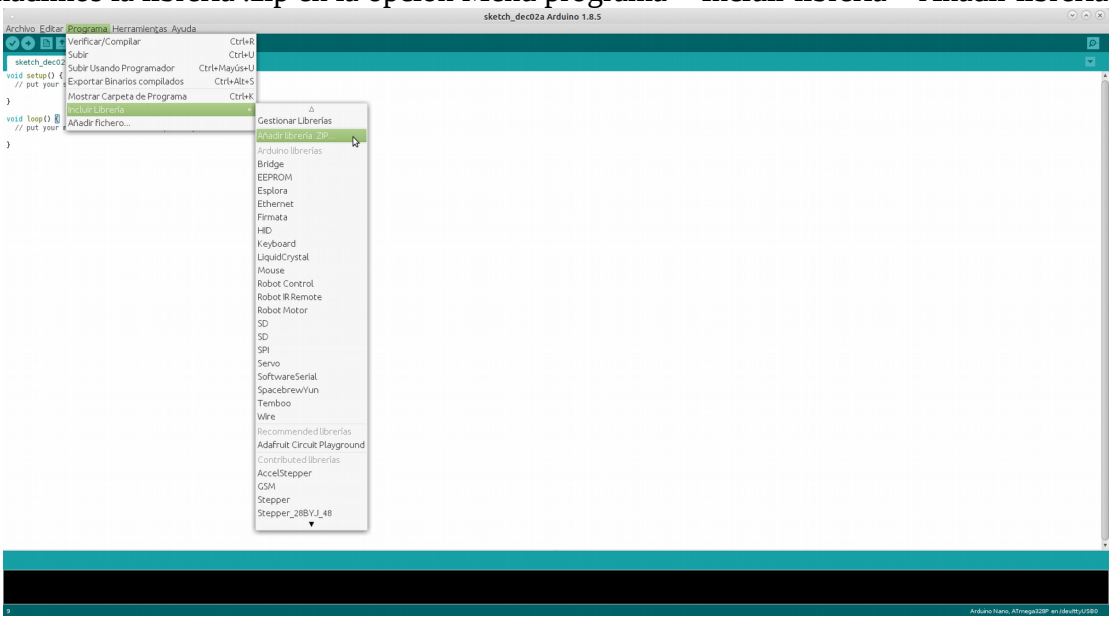
1. ¿Por qué la librería?

La librería nace en el seno del *Club de Tecnología, programación y robótica de Granada*, con objeto de cubrir la necesidad de programar escornabot en un entorno de programación textual (IDE Arduino). Escornabot en un principio ha sido diseñado para programarlo exclusivamente con la botonera pensando en el alumnado de primaria, por tanto para alumnado más mayor y con más capacidad de abstracción (tercer ciclo de primaria y ESO) se hacía necesario adaptarle una serie de instrucciones para poder manejar escornabot con programación textual.
La librería es desarrollada por Prudencio Luna (@plunax) y Pedro Ruiz (@pedroruizf).

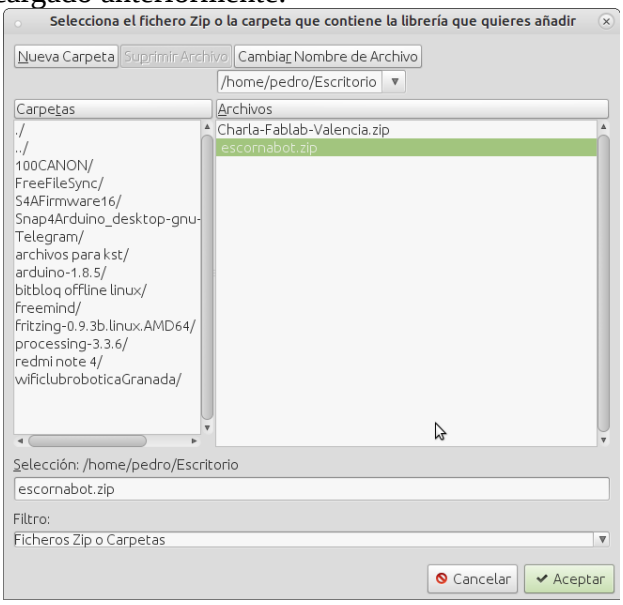
2. ¿Cómo incorporar la librería en IDE de Arduino?

Para ello descargamos el archivo "escornabot.zip" de la librería, y lo incorporamos como librería a nuestro entorno arduino según la siguiente secuencia:

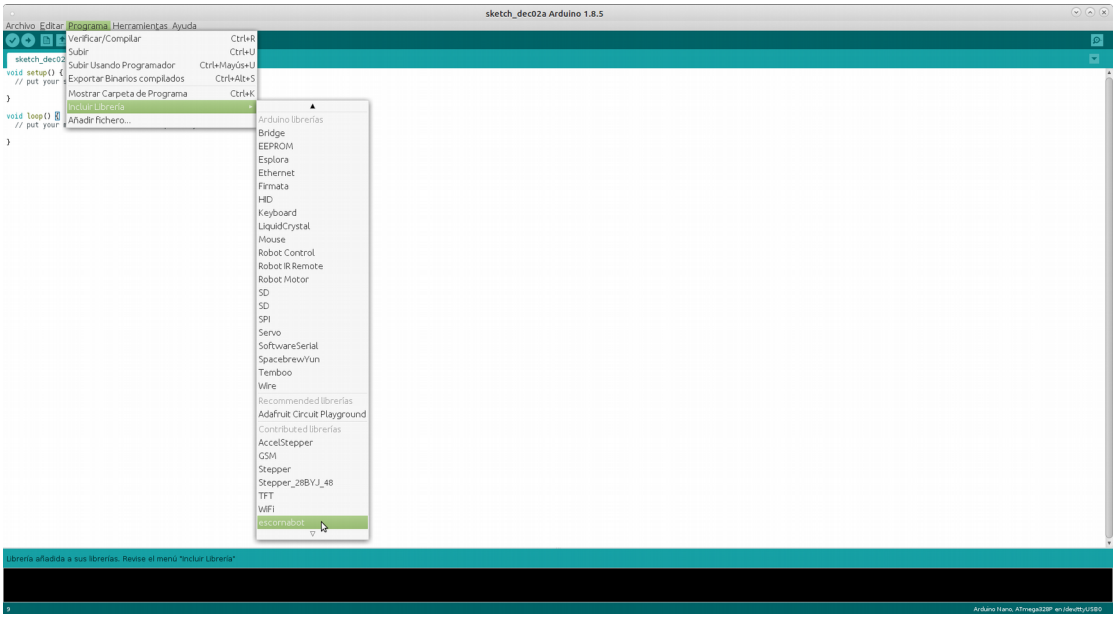
1. Añadimos la librería .zip en la opción Menu programa > Incluir librería > Añadir librería .zip



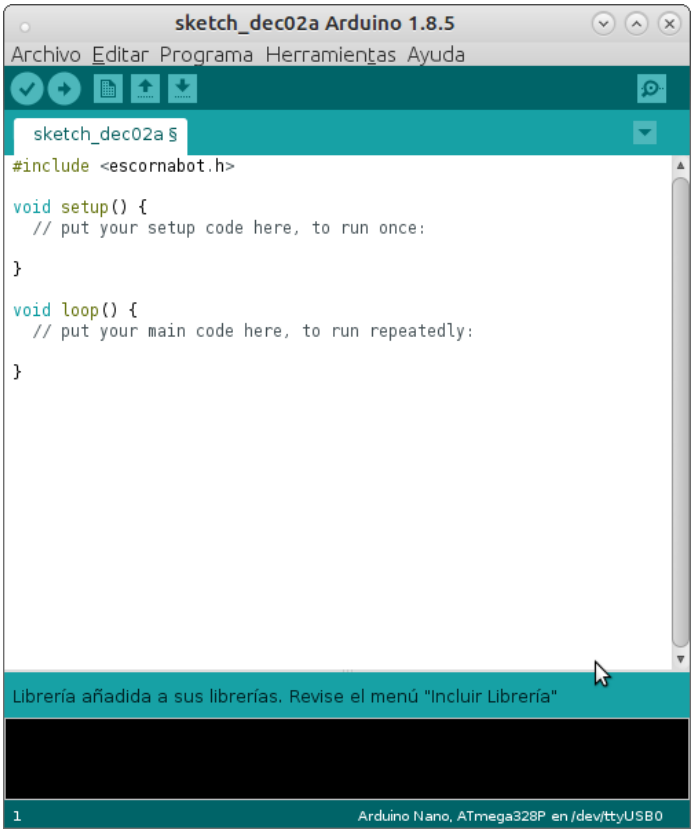
2. Ahora nos aparece un menú de elección del fichero zip, debemos elegir nuestro fichero "escornabot.zip" descargado anteriormente.



3. Una vez incluido ya tenemos la librería disponible en nuestro listado de librerías.

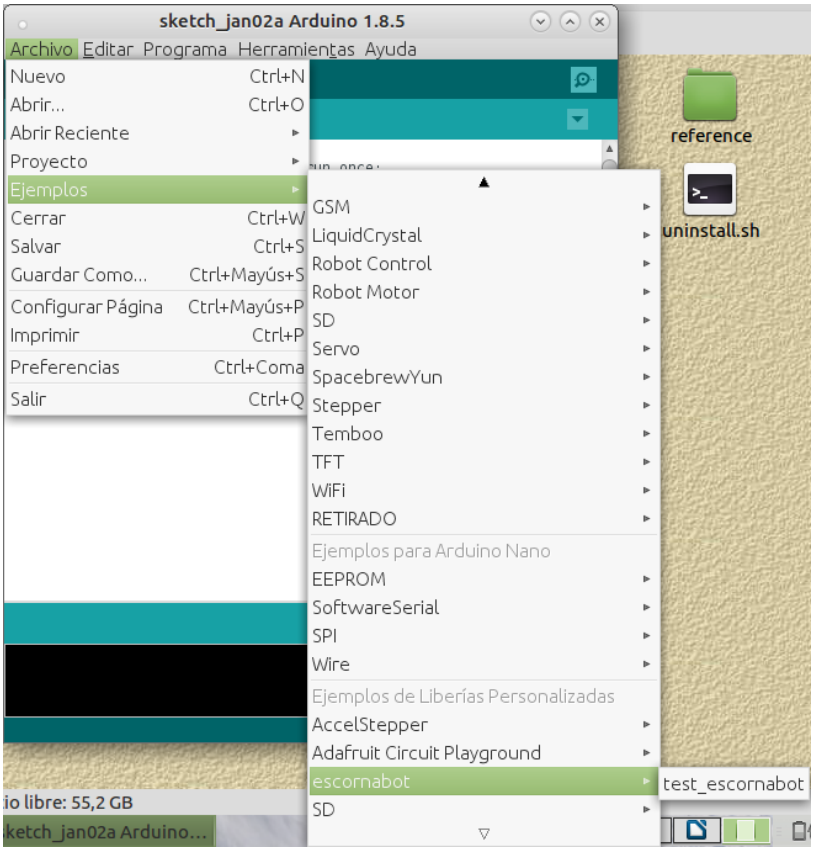


4. Seleccionándola la incluimos en nuestro programa “#include <escornabot.h>”, a partir de ahora ya puedo utilizar las funciones de la misma.



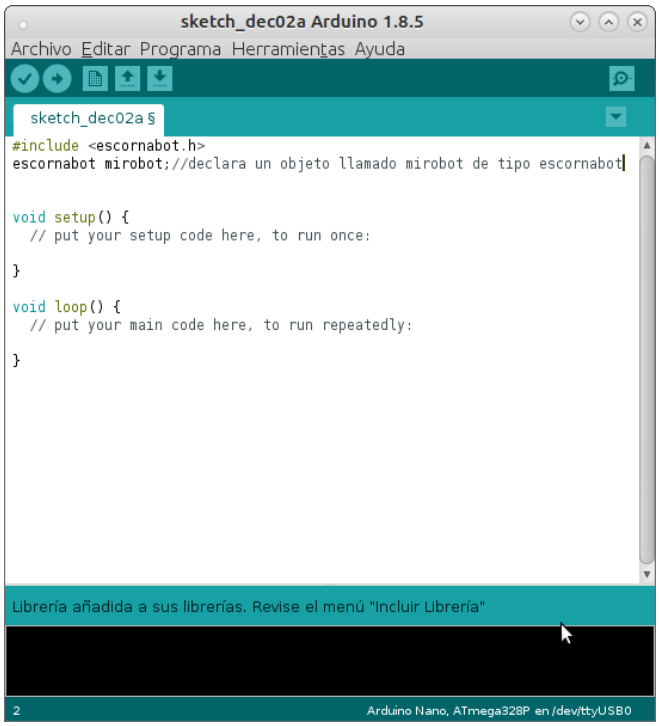
3. ¿Cómo cargar el ejemplo de la librería?

Todas las librerías suelen tener asociado al menos un ejemplo, para cargarlo una vez incorpordada la librería a el IDE de Arduino (ver punto 2), debemos ir la opción del menú Archivo > Ejemplos > escornabot > test_escornabot.



4. ¿Cómo mover escornabot?

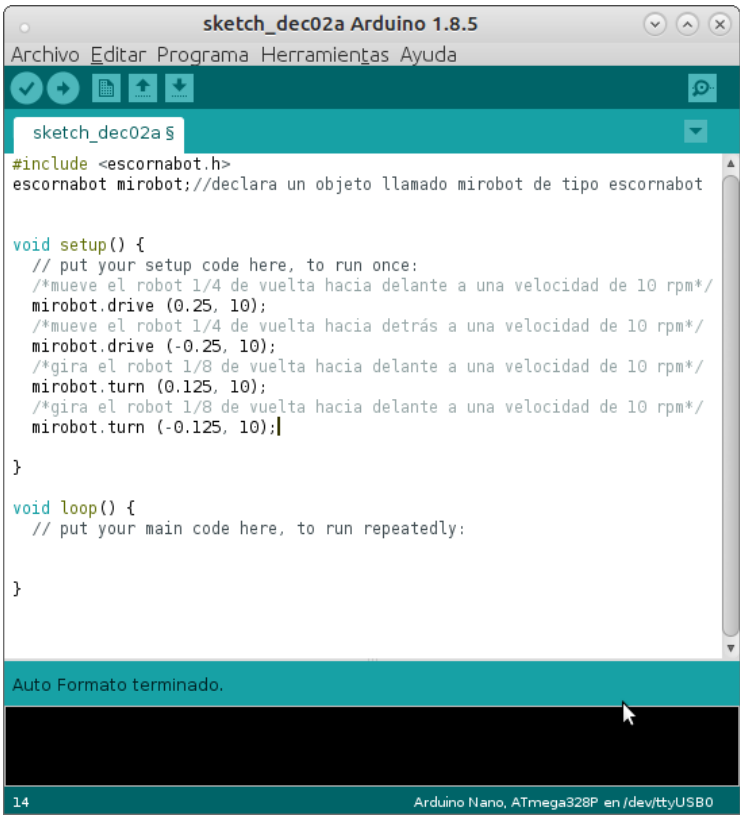
Para mover escornabot o realizar cualquier otra función con el mismo, debemos en nuestro programa declarar un objeto tipo escornabot, para que el mismo pueda realizar todas las funciones (procedimientos) de un escornabot.
Para ello en nuestro programa en la parte de declaración de variables vamos a incluir dicho objeto, en mi caso se llama “mirobot”.



Ahora a este objeto “mirobot” podemos decirle que haga varias acciones de movimiento. Las posibilidades de movimiento del mismo son:

- **objetoEscornabot.drive (vueltas, velocidad)**: Sirve para avanzar o retroceder. Se mueve el número de vueltas indicado, si son negativas va en el sentido contrario. La velocidad se da rpm.
- **objetoEscornabot.driveD (distancia, velocidad)**: Sirve para avanzar o retroceder. Se mueve la distancia en cm indicada, si son negativas va en el sentido contrario. La velocidad se da rpm.
- **objetoEscornabot.turn (vueltas, velocidad)**: Sirve para girar. Se indica como antes el número de vueltas o fracción a girar, si son positivas gira en un sentido y negativas en el contrario. La velocidad se da en rpm.
- **objetoEscornabot.turnA (angulo, velocidad)**: Sirve para girar. Se indica el ángulo a girar, si es positivo gira en un sentido y negativo en el contrario. La velocidad se da en rpm.
- **objetoEscornabot.stop ()**: detiene los dos motores.

En el siguiente ejemplo (ejemplo 1) las acciones las hemos programado en el “*setup()*”, para que sólo se ejecute una vez. El robot se mueve hacia delante, hacia detrás, gira hacia derecha y gira hacia izquierda quedándose en la posición de partida:



5. Luces y sonido

Escornabot dispone de cuatro leds y un zumbador, para ellos también tenemos procedimientos para accionarlos o apagarlos, que son:

- **objetoEscornabot.ledON (número de led o posición en inglés)**: sirve para encender los leds de escornabot. Los leds son: 1 o forward (azul, posición delantera), 3 o backward (ámbar, posición trasera), 4 o right (verde, posición derecha), y 2 o left (rojo, posición izquierda).

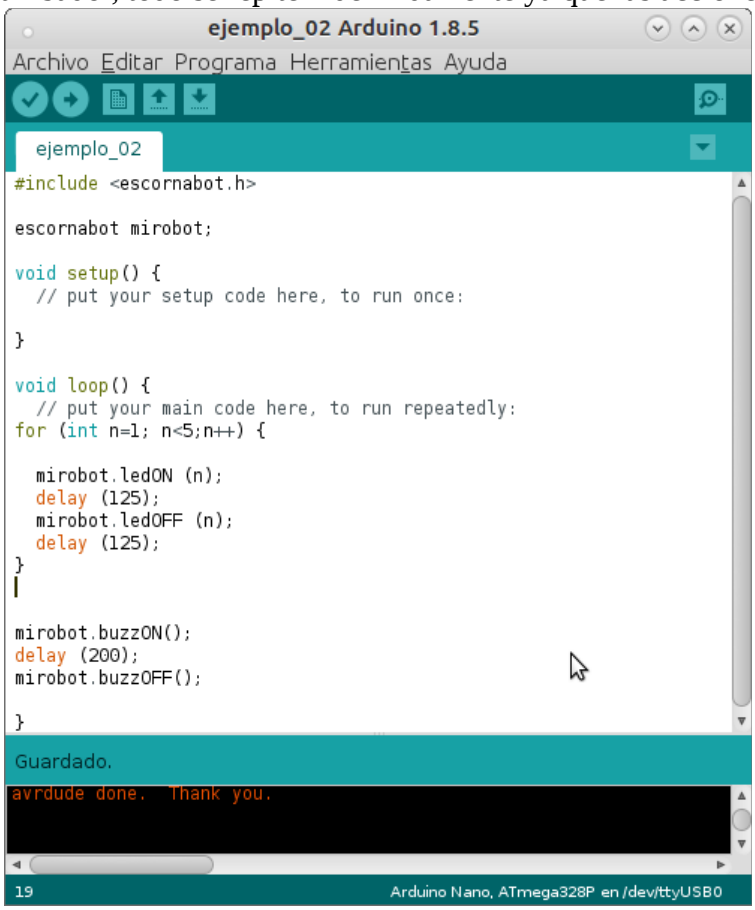
  •**objetoEscornabot.ledOFF (número de led o posición en inglés)**: sirve para apagar los leds de escornabot.

  •**objetoEscornabot.ledState (número de led o posición en inglés)**: devuelve el estado del led, encendido (1 o HIGH) o apagado (0 o LOW).

  •**objetoEscornabot.buzzON ()**: enciende el zumbador.

  •**objetoEscornabot.buzzOFF ()**: apaga el zumbador.

En el siguiente ejemplo (ejemplo 2) procedemos a encender los leds secuencialemente, y cuando termina el último suena el zumbador, todo se repite indefinidamente ya que las acciones están el *"loop()"*:



6. Control mediante botonera

Hasta ahora sólo hemos accionado actuadores, ha llegado el momento de inetractuar con la botonera, para ello disponemos de un procedimiento para evaluar que botón ha sido pulsado, en función de su valor podemos accionar los actuadores (motores, leds y zumbador). El procedimiento es:

- **objetoEscornabot.pushButton()**: devuelve el valor del botón pulsado o la posición en inglés. 1 o forward (delantero), 3 o backward (trasero), 4 o right (derecho), 2 o left (izquierdo), 5 o central (central).

En el siguiente programa (ejemplo 3) al pulsar cualquiera de los cuatro botones de dirección (adelante, atrás, derecha, izquierda) realiza la correspondiente acción de movimiento o giro y mientras la está haciendo se enciende el led correspondiente que se apagará cuando termine la acción. Cuando se pulsa el botón central se enciende el zumbador durante un tiempo, a la vez que los cuatro leds:

```
#include <escornabot.h>

escornabot mirobot;

void setup() {
  Serial.begin (9600);
}

void loop() {
  //prueba de librería

  switch (mirobot.pushButton()) {
```

```
  case forward://si pulsamos el botón delantero, se enciende led delantero, se mueve 1/4 de vuelta hacia delante, y se apaga el led delantero
    mirobot.ledON (forward);
    mirobot.drive (0.25, 10);
    mirobot.ledOFF (forward);
    break;

  case backward://si pulsamos el botón trasero, se enciende led trasero, se mueve 1/4 de vuelta hacia atrás, y se apaga el led trasero
    mirobot.ledON (backward);
    mirobot.drive (-0.25, 10);
    mirobot.ledOFF (backward);
    break;

  case right://si pulsamos el botón derecho, se enciende led derecho, se mueve 1/8 de vuelta hacia la derecha, y se apaga el led derecho
    mirobot.ledON (right);
    mirobot.turn (0.125, 10);
    mirobot.ledOFF (right);
    break;

  case left://si pulsamos el botón izquierdo, se enciende led izquierdo, se mueve 1/8 de vuelta hacia la izquierda, y se apaga el led izquierdo
    mirobot.ledON (left);
    mirobot.turn (-0.125, 10);
    mirobot.ledOFF (left);
    break;

  case central://si pulsamos el botón central, suena el zumbador y se enciende todos los leds durante un segundo, después se apagan el zumbador y los leds
    mirobot.buzzON ();
     for (int i = 1; i < mirobot.numberLeds; i++)//La constante mirobot.numberLeds esta definida en la librera y vale 5
    {
      mirobot.ledON(i);
    }
    delay (1000);
    mirobot.buzzOFF();

    for (int i = 1; i < mirobot.numberLeds; i++)
    {
      mirobot.ledOFF(i);
    }

  }


}
```

7. Control mediante Bluetooth

Escornabot en su versión con cpu 2.12 tiene la posibilidad de comunicarse por bluetooth. Para ello se ha diseñado un procedimiento para capturar un carácter enviado desde un dispositivo móvil, y en función del mismo podemos decirle a nuestro escornabot la función a realizar. El procedimiento es el siguiente:

- **objetoEscornabot.blueT()**: devuelve el valor numérico correspondiente a el carácter enviado por bluetooth a escornabot.

En nuestro caso hemos diseñado un programa (ejemplo 4) que en función del carácter enviado desde la aplicación “Arduino Bluetooth controller” realizamos una acción u otra. Si se envía el carácter:

- ‘A’: se mueve hacia delante (avanza).
- ‘R’: se mueve hacia atrás (retrocede).
- ‘D’: gira hacia la derecha.
- ‘I’: gira hacia la izquierda.
- ‘1’: enciende o apaga el led 1 (delantero), en función de como estuviese anteriormente.
- ‘2’: enciende o apaga el led 2 (izquierda), en función de como estuviese anteriormente.
- ‘3’: enciende o apaga el led 3 (trasero), en función de como estuviese anteriormente.
- ‘4’: enciende o apaga el led 4 (derecha), en función de como estuviese anteriormente.
- ‘5’: enciende o apaga el zumbador, en función de como estuviese anteriormente.

```
#include <escornabot.h>

escornabot mirobot;
boolean led1 = false;
boolean led2 = false;
boolean led3 = false;
boolean led4 = false;
boolean buzz = false;

void setup() {
  Serial.begin (9600);
}

void loop() {
  //prueba de librería

    switch (mirobot.blueT()) {//en función del caracter emitido por bluetooth hace varias acciones
    case 'A':
      mirobot.drive (0.25, 12);
      break;
    case 'R':
      mirobot.drive (-0.25, 12);
      break;
    case 'D':
      mirobot.turn (0.125, 12);
      break;
    case 'I':
      mirobot.turn (-0.125, 12);
      break;
    case '1':
      /*led1 = !led1;
      if (led1) {
```

```
      mirobot.ledON(forward);
     }
     else {
      mirobot.ledOFF(forward);
     }*/
     invierteLed(forward);
     break;
   case '2':
     led2 = !led2;
     if (led2) {
      mirobot.ledON(left);
     }
     else {
      mirobot.ledOFF(left);
     }
     break;
   case '3':
     led3 = !led3;
     if (led3) {
      mirobot.ledON(backward);
     }
     else {
      mirobot.ledOFF(backward);
     }
     break;
   case '4':
     led4 = !led4;
     if (led4) {
      mirobot.ledON(right);
     }
     else {
      mirobot.ledOFF(right);
     }
     break;
   case '5':
     buzz = !buzz;
     if (buzz) {
      mirobot.buzzON();
     }
     else {
      mirobot.buzzOFF();
     }
     break;
     //default:
     // statements
  }

}

void invierteLed(int i){
  if (mirobot.ledState(i)) {
    mirobot.ledOFF(i);
```

```
  } else {
    mirobot.ledON(i);
  }
}
```

Previamente en la aplicación “Arduino bluetooth controller” tenemos que editar el carácter que enviará cada botón, esto lo hacemos en la configuración de la aplicación.