Modulo_1:

Condicional IF y uso de Librerías

FORO: https://nac-arduino.herokuapp.com/
REPOSITORIO: https://github.com/rody7val/nac-arduino/

Rodolfo Valguarnera, NAC - Pigüé

Contenidos:

1. Condicional IF.
2. Desafío IF.
3. Uso de Librerías.

Condicional IF

El condicional IF es una estructura de control que redirige el curso de una acción según una evaluación simple.

Este operador IF verifica simplemente si un test lógico es cierto o falso (es decir, si devuelve el valor true o false) y en función de esto realiza (o no) una serie de acciones.

```
if ( test_logico ){
  // realiza una serie de acciones si el test lógico resulta ser verdadero (“true”)
}
```

Ejemplo :

```
if ( varA < varB ){
  digitalWrite(ledRojo, HIGH);
}
```

Condicional IF

El operador IF puede ser complementado con el operador ELSE (sino...). El bloque de acciones asociadas al "else" son ejecutadas si el test lógico del "if" dio false como resultado.

Esto funcionaría de la siguiente manera:

```
if ( test_logico ){
  // si el test lógico es verdadero (“true”) se ejecuta el primer bloque de acciones (if)
} else {
  // en cambio, si es falso (“false”) se ejecuta el segundo bloque de acciones (else)
}
```

Es importante notar que solamente un bloque de acciones es ejecutado dentro de un IF!

Desafío IF

Crear un nuevo prototipo Arduino en circuits.io para comprender el uso del condicional IF. El prototipo debe controlar un LED según el valor obtenido de un sensor de fotocelula.

Respetar los siguientes requisitos mínimos:

- Nombre: Fotocelula-Led-Control.
- Componentes: Arduino Uno R3, Photoresistor (LDR), LED, Resistor, Breadboard.
- Sketch: ----------→

```
int pinDigital = 13;

int luzMaxima = 400;

int valor;

void setup() {

  // configuro pin 13 como salida para poder manipularlo

  pinMode(pinDigital, OUTPUT);

}

void loop() {

  valor = analogRead(pinAnalog);   // leo el valor del sensor

  // si hay poca luz

  if(valor < luzMaxima){

    digitalWrite(pinDigital, HIGH);   // encender foco

  }else{

    digitalWrite(pinDigital, LOW);   // sino, apagar foco

  }

}
```

Uso de librerías

Las librerías son trozos de código hechas por terceros que usamos en nuestro sketch. Esto nos facilita mucho la programación y hace que nuestro programa sea más sencillo de hacer y luego de entender.

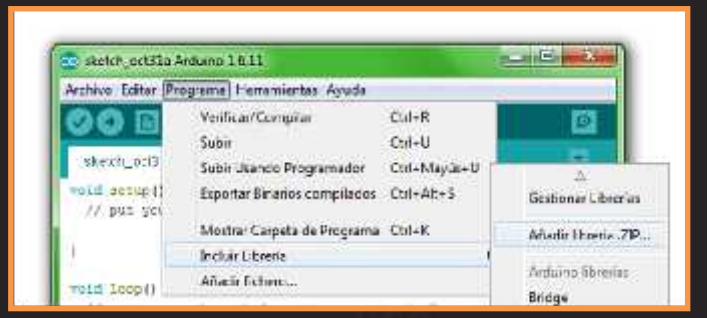
Una librería a diferencia de las funciones debe estar al menos en un fichero diferente con extensión .h y opcionalmente en otro .cpp y además debe ser llamada con #include desde el sketch de Arduino y estar en una ruta accesible desde el IDE de Arduino, ya sea el mismo directorio del sketch o en algunas de las rutas configuradas para librerías.

La ventaja de usar librerías frente a las funciones es que no es necesario incluir el código cada vez que se va a reutilizar sino que con tener la librería instalada en el IDE y llamarla mediante #include ya la puedo usar en mi código.

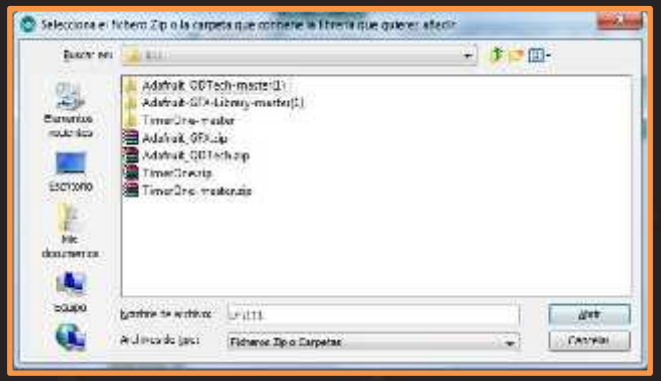
Al llamar a una librería desde un sketch, la librería completa es cargada a la placa de Arduino incrementando el tamaño del espacio usado en el microcontrolador, tanto en la memoria flash como en la RAM.

Uso de librerías

Instalar una librería: Lo primero que tenemos que hacer es descargar la librería en cuestión. Luego elegimos en el menú del IDE Arduino:
Arduino → Programa → Importar Librería → Añadir Librería.



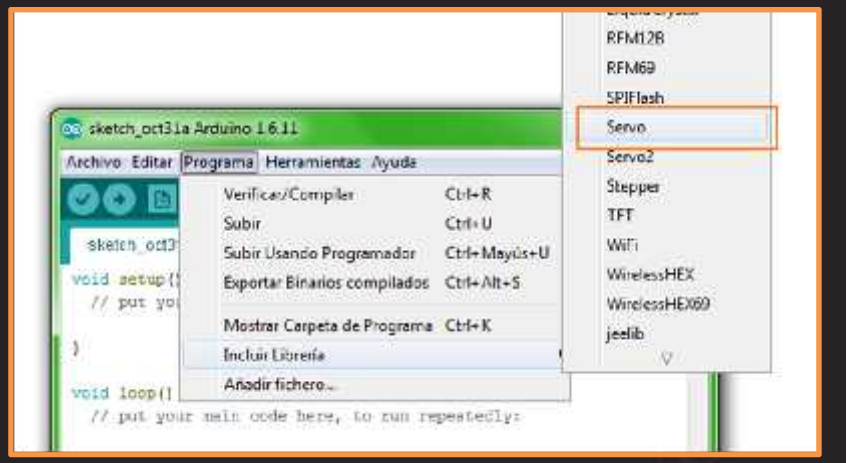
Ahora nos pedirá la dirección del fichero descargado en formato xxx.zip.



Busca y selecciona la librería en cuestión. Cuando la pinchemos se debería instalar. Listo, si no hay ningún problema podemos comprobar que aparece en la lista de librerías instaladas y que podemos seleccionarla para incluirla en nuestros sketch.

Uso de librerías

Incluir una librería en nuestro sketch: Ya sea que hayamos instalado una librería o queramos usar una ya instalada (la librería servo por ejemplo) debemos seleccionar una:



Arduino→Programa→Incluir Librería→[Seleccionar].

Esta acción devolvera un codigo como el siguiente en nuestro sketch:

```
#include <Servo.h>
```

Con esto incorporamos la libreria, aunque tambien podemos escribir el texto directamente. Una vez echo esto ya podemos usar la libreria Servo.
Ahora vamos a definir nuestro objeto Servo, esto es como definir una variable de tipo int o float, pero un poco mas completa con funciones y campos que le pertenecen. Para eso ponemos Servo miServo.

```
Servo miServo;
```

Desafío IF

Crear un nuevo prototipo Arduino en circuits.io para comprender el uso de incluir liobrerias. El prototipo debe mover un motor servo 180 grados y volver a 0 grados.

Respetar los siguientes requisitos mínimos:

- Nombre: Servo-Control.
- Componentes: Arduino Uno R3, mini servo.
- Sketch: ----------

```
#include <Servo.h>

//creamos una variable motor de tipo Servo para poder utilizar las funcione

Servo motor;

int pos;    //inicializamos variable posicion

void setup() {

  motor.attach(9);    //selecciono el pin 9

}

void loop() {

  for (pos = 0; pos <= 180; pos ++) {    //girar 180º

    motor.write(pos);

    delay(15);

  }

  for (pos = 180; pos >= 0; pos --) {    //volver 180º

    motor.write(pos);

    delay(15);

  }

}
```

Fin del tema.

Muchas gracias!

FORO: https://nac-arduino.herokuapp.com/
REPOSITORIO: https://github.com/rody7val/nac-arduino/

Rodolfo Valguarnera, NAC - Pigüé