LIBRERIAS CON ARDUINO (Parte 1)

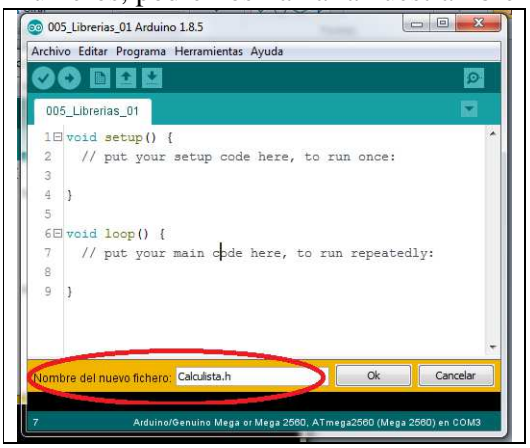
Las librerías son segmentos de código reutilizables, hechos por terceros o por nosotros, que guardamos aparte, para ser usados cuando nos hagan falta en nuestros programas. Esto nos facilita mucho la programación y hace que nuestro programa sea más sencillo de hacer y de entender.

Entonces, las librerías son entre otras cosas colecciones de código, listos para usar cuando sean requeridos, por ejemplo, si ya sabemos usar o interconectar sensores, pantallas, módulos electrónicos, etc, no hace falta hacer todo cada vez que lo usemos, simplemente almacenamos el código y luego lo usamos. El entorno de arduino, ya incluye algunas librerías, pero las analizaremos un poco más adelante.

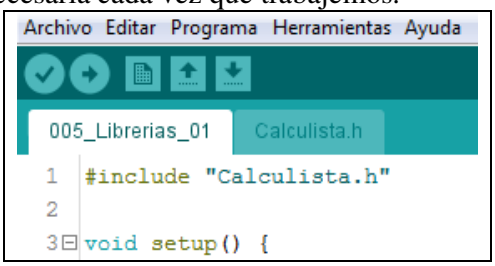
Generalmente cada dispositivo que compramos, sensores, actuadores, motores, etc… vienen o tienen su propia librería que debemos instalar para poder usarlos. Estas librerías podemos verlas, modificarlas o incluso añadir funcionalidades. Entonces, a la hora de elegir un elemento de hardware, uno de los argumentos importantes es la librería que nos ofrece el fabricante o la comunidad y su facilidad de uso. Pero esto también lo veremos más adelante.

UNA LIBRERÍA ESCRITA POR NOSOTROS

Comencemos por lo más fácil, una librería hecha por nosotros, con un contenido que ya conocemos y usaremos seguidamente.
El nombre que le pondremos a esta librería puede ser cualquiera, pero siempre debe terminar (tiene la extensión) "**h**". Por ejemplo, si queremos hacer una librería que, contenga una función que sume dos números, podremos llamar a nuestra librería "**Calculista.h**". Entonces los pasos serian:



1- Entrar al entorno del IDE de Arduino.
2- Después teclear Ctrl+Shift+N. (Shift es la tecla que usamos para escribir una letra con mayúscula y continuar escribiendo en minúscula - Generalmente, en los teclados hay dos teclas "**Shift**", una a la Izquierda, debajo del "Bloq Mayus" y la otra a la derecha debajo de la tecla "**Enter**").
3- Se abre una ventana en la parte inferior del IDE de Arduino (ver imagen).
4- Colocar nombre del nuevo archivo, en nuestro caso: "Calculista.h"
5- Cuando grabemos el programa, la biblioteca será guardada en nuestra carpeta de trabajo y quedará automáticamente asociada a nuestro programa, pudiendo accesarla cada vez que trabajemos.
6- **Recordar siempre** incluir la biblioteca en nuestro programa, al comienzo (arriba de todo) usando la siguiente línea de código: #include "Calculista.h".



IMPORTANTE: La directiva **#include** existe en dos versiones. En una se pone el nombre de archivo entre comillas, en la otra entre los signos mayor y menor **< >**.

a- **#include <Biblioteca.h>** La versión con los paréntesis angulares busca los archivos de librería en todos los directorios que se han especificado en la configuración del compilador. Generalmente estos archivos de biblioteca se encuentran en la carpeta donde se instalo el IDE Arduino.

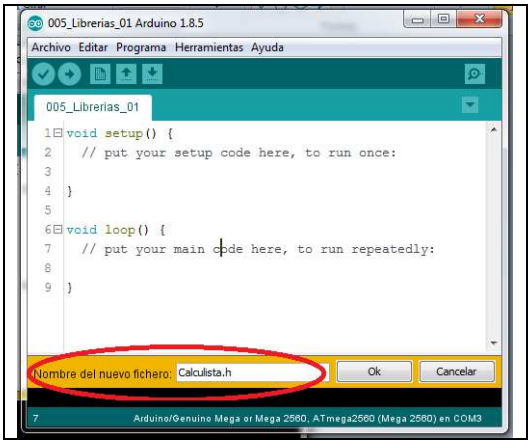
b- **#include "Biblioteca.h"** Cuando se incluye un archivo de Librería entre comillas, el compilador busca este archivo primero en el mismo directorio que el archivo actualmente compilado (nuestro programa) y después en los demás directorios.

Es decir, la versión con comillas se diferencia de la versión con signo mayor y menor, únicamente por buscar primero en el directorio o carpeta donde guardamos nuestro programa.

Ahora repitamos el ejemplo, pero haciendo todo de verdad.

1- Nuestra primer Librería.

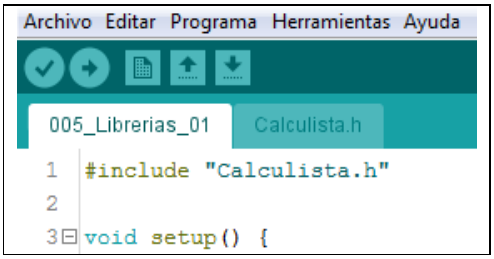
Este programa envía por "Monitor Serie", el resultado de sumar dos números. La suma es **realizada por una función de la librería** llamada "**Calculista.h**", creada por nosotros (Algo muy simple para comenzar). Ahora a Trabajar.! Primero Creamos muestro programa, ya lo hemos hecho muchas veces, pero lo repetiremos una vez mas.



a- Ingrese al IDE de Arduino, seleccione del menú "**Archivo**" la opción "**Nuevo**" y a continuación, nuevamente desde menú "**Archivo**" seleccionamos, esta vez la opción "**Guardar Como**"… Escribimos el nombre que tendrá nuestro archivo "**005_Librerias_01**" – (Respete el nombre pues lo usaremos en próximos ejercicios). Bueno, ya hemos creado el cuerpo del programa (sin código), ahora crearemos la librería.

b- Ya creado nuestro programa, presionar la siguiente combinación de teclas, sin soltar ninguna **Ctrl+Shift+N** y al soltar las teclas…… Al instante se abre una ventana en la parte inferior del IDE de Arduino.

c- Colocar nombre de la nueva Liberia, en este caso: "**Calculista.h**". Presione OK y esta será creada.



d- Ya creada la Librería, será guardada en la carpeta de trabajo y asociada a nuestro programa, pudiendo accesarla cada vez que trabajemos. Por ahora están vacíos el programa y la Librería, pero la llenaremos con todo el código.

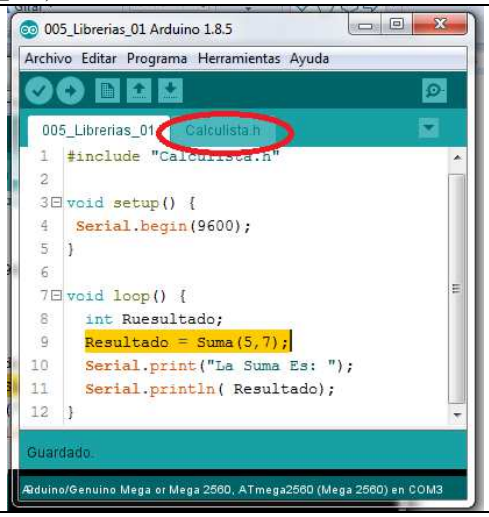
e- Posicionémonos en el programa. Clic en la solapa correspondiente, y procedamos a incluir la biblioteca, al comienzo (arriba de todo) usando la siguiente línea de código:

f- Escribamos el código correspondiente al programa.

g- **RECORDAR:** Este Programa, no necesita de un circuito, trabajara en la placa Arduino, y solo interactúa con la PC a través del "Monitor Serie". Los números que se sumaran, son siempre los mismos 5 y 7, solo sirven como ejemplo. El número que se recibirá en la PC (la suma) será siempre el mismo e igual a 12.

(Programa "**005_Librerias_01**")

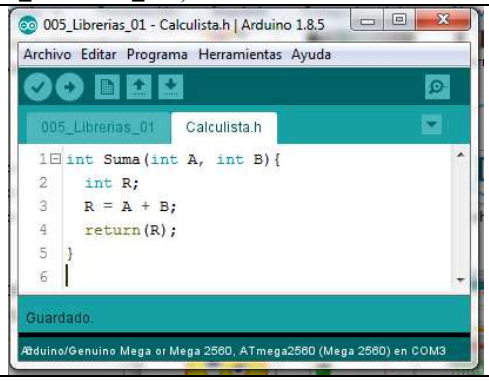
```
1  #include "Calculista.h"
-
3  void setup( ) {
4    Serial.begin(9600);
5  }
-
6  void loop( ) {
7    int Resultado;
8    Resultado = Suma(5,7);
9    Serial.print("La Suma Es: ");
10   Serial.println( Resultado);
11 }
```



Hora hay que escribir en la Librería el código de la función, para esto hacer clic en la solapa de "Calculista.h. y Escribamos:

(Librería "Calculista.h" del Programa "**005_Librerias_01**")

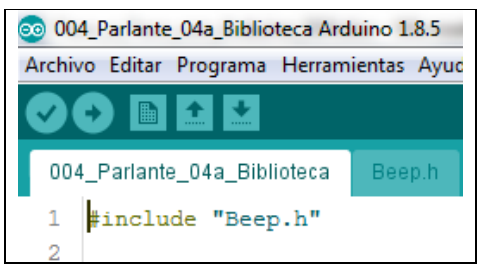
```
1  //  A y B son los parámetros de entrada
3  // es decir por donde ingresan los datos que
4  // entregamos al llamar la función
5  int Suma(int A, int B){ // Nuestra Función
6    int R;
     R = A + B;
     return(R);
   }
```



2- Trabajando con Librerías Propias 01.

En este ejercicio, trabajaremos nuevamente con el programa "004_Parlante_03a" Que usamos en la "**guía 3**" y el enunciado **dice:** "Emitir Beep-Beep por el Speaker/Zumbador- (Versión B) – Programación Multitarea, sin usar "**delay( ).**"

a. Abra el programa y guárdelo como "004_Parlante_04a_Biblioteca" sobre este trabajaremos ahora.
b. Ahora crearemos el archivo que usaremos como librería y llamaremos Beep.h. Si no se encuentra en el IDE de Arduino, ábralo y cargar el programa "004_Parlante_04a_Biblioteca".
c. Después teclear Ctrl+Shift+N.
d. En este instante, se abrió una ventana en la parte inferior del IDE de Arduino.
e. Colocar nombre del nuevo archivo, en nuestro caso: "Beep.h". Presione tecla OK y este se guardara.
f. Ya Guardada la Librería, se encuentra en nuestra carpeta de trabajo y esta asociada a nuestro programa, pudiendo accesarla cada vez que trabajemos. Por ahora esta varia, pero ya la llenaremos con todo el código.
g. **Posicionémonos en el programa. Clic en la solapa correspondiente, y procedamos a** incluir la biblioteca, al comienzo (arriba de todo) usando la siguiente línea de código: #include "Beep.h".



h. A continuación, moveremos el código necesario para usar la función "**HaceBeep( )**". Esto incluye La función misma, variables que usamos y el código de inicialización, que estaba en la función "**setup( )**" del programa, ahora lo pondremos dentro de una función nuestra, que llamaremos "**IniciaSpeaker( )**" y le daremos como dato (Parámetro), el numero de PIN en el que deberá trabajar. Sin más explicaciones, veamos como queda nuestro programa con su librería "Beep.h"

(Programa "004_Parlante_04a_Biblioteca")

```
1   #include "Beep.h"  // Inclusión de nuestra Librería
-
2   const int SpeakerPin = 10;
-
3   void setup( ) {
4     IniciaSpeaker(SpeakerPin);
5   }
-
6   void loop( ) {
7     HaceBeep( );  // Llamada a la Función Contenida en la Librería
8     // Otras Acciones
9     // Otras Acciones
10  }
```

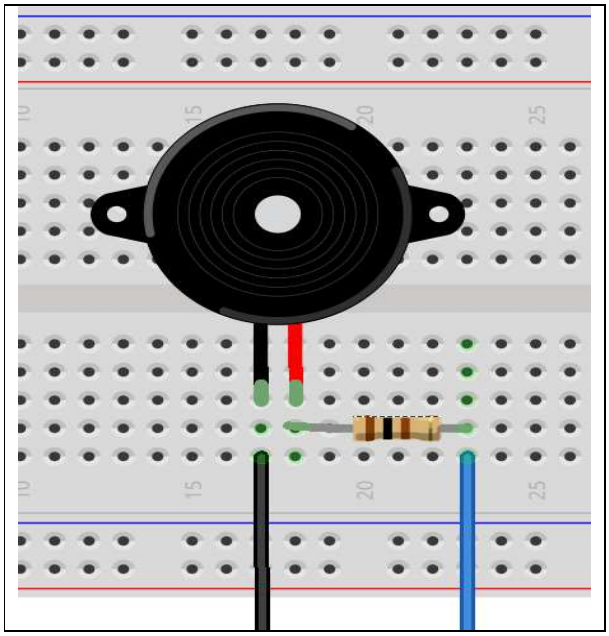
A continuación, generamos y grabamos la Librería **Beep.h**

(Librería "Beep.h" del Programa "004_Parlante_04a_Biblioteca")

```
1   /*
2   Librería Ejemplo.
3   Creada por Castelli Horacio P.  (Enero 10, 2007). Domiño Público.
4   Para crear el Archivo Librería
5   1-  Entrar al entorno del IDE de Arduino.
6   2-  Después se debe teclear  Ctrl+Shift+N.
7   3-  En ese instante se abre una  ventana en la parte inferior del IDE de Arduino.
8   4-  Colocar nombre del nuevo archivo, en nuestro caso:  "Beep.h"
9   5-  Cuando grabemos el programa, la Librería será guardada en nuestra carpeta de
10  trabajo y quedará automáticamente asociada a nuestro programa.
11  6-  IMPORTANTE:  Recordar siempre incluir la biblioteca en nuestro programa,
12  Arriba de todo:   #include "NombreLibreria.h"
13  */
-
14  double TiempoInicialSpeaker,
15         TiempoEsperaSpeaker;
-
16  int   PinUsadoSpeaker,
17        EstadoBeepSpeaker,
18        FrecuenciaBeep;
-
19  void IniciaSpeaker(int PinParlante){
20    PinUsadoSpeaker = PinParlante;
21    pinMode(PinUsadoSpeaker,OUTPUT);
22    TiempoInicialSpeaker = millis( );
23    TiempoEsperaSpeaker = 500;
24    EstadoBeepSpeaker = 0;
25    FrecuenciaBeep = 500;
26  }
-
```

```
27  void HaceBeep(void){
28    if(millis( ) < TiempoInicialSpeaker + TiempoEsperaSpeaker ){
29      if(EstadoBeepSpeaker == 1){
30        tone(PinUsadoSpeaker,FrecuenciaBeep); // Asociando SpeakerPin con una
-                                               // Frecuencia determinada.
31      }else{
32        noTone(PinUsadoSpeaker);    // Libero SpeakerPin
33      }
34    }else{
35      EstadoBeepSpeaker = 1 - EstadoBeepSpeaker;
36      TiempoInicialSpeaker = millis( );
37    }
38  }
```

CIRCUITO PARA NUESTRO PROYECTO



Lista de Materiales: 1 Resistencia de 470Ω, 2 Cables Macho/Macho - 1 Placa Protoboard - Placa Arduino y 1 Cable USB.

-o-

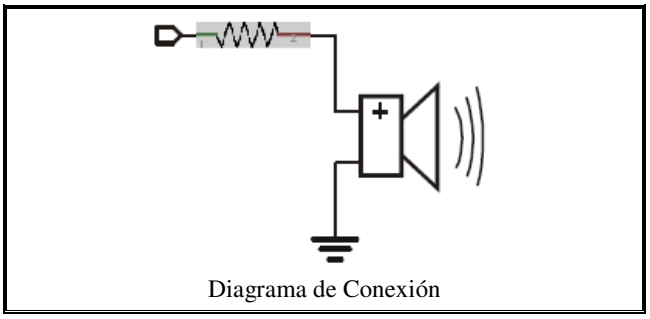


Diagrama de Conexión

Conexión de cables

Abajo, de Izquierda a Derecha: Cable Negro conectar a GND, y cable Azul, conectar al PIN configurado en el programa a través de la constante "SpeakerPin".

Guarde este programa, lo usaremos nuevamente en breve.

3- Trabajando con Librerías Propias 02.

Acá les dejo unas melodías de películas: Harry Potter y Star Wars. El objetivo es usar librerías creadas por nosotros. Más adelante trabajaremos meticulosamente con sonido, y usaremos más programación Multitarea. Es para los que saben algo de Música, espero les guste.

(Programa "005_Librerias_02")

```
1   #include "Musica.h"
2   int SpeakerPin = 12; // Speaker se conecta a GND y pin 12
-
3   void setup( ){
4     IniciaParlante(SpeakerPin);
5   }
6   void loop( ){
7     HarryPotter( );
8     StarWars( );
9     EntreDosAguas( );
```

```
10  }
```

-o-

(Librería "Musica.h" del Programa "005_Librerias_02")

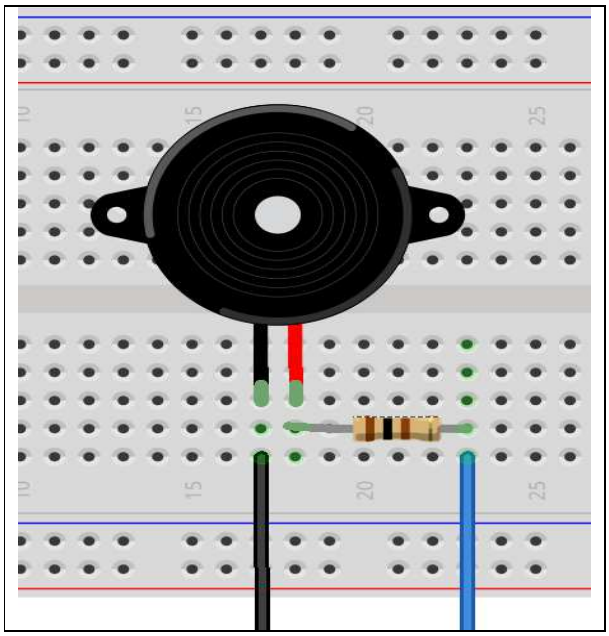
```
1   /*
2   Librería Ejemplo.
3   Creada Enero 10, 2015.
4   Domiño Público.
5   Para crear el Archivo Librería
6   1-  Entrar al entorno del IDE de Arduino.
7   2-  Después se debe teclear  Ctrl+Shift+N.
8   3-  En ese instante se abre una  ventana en la parte inferior del IDE de Arduino.
9   4-  Colocar nombre del nuevo archivo, en nuestro caso:  "Musica.h"
10  5-  Cuando grabemos el programa, la Librería será guardada en nuestra carpeta de
11  trabajo y quedará automáticamente asociada a nuestro programa.
12  6-  IMPORTANTE:  Recordar siempre incluir la biblioteca en nuestro programa,
13  Arriba de todo:   #include "nombre.h"
14  */
-
15  int Parlante;
-
16  int c[5]= {131,262,523,1046,2093};    // frecuencias 4 octavas de Do
17  int cs[5]={139,277,554,1108,2217};    // Do#
18  int d[5]= {147,294,587,1175,2349};    // Re
19  int ds[5]={156,311,622,1244,2489};    // Re#
20  int e[5]= {165,330,659,1319,2637};    // Mi
21  int f[5]= {175,349,698,1397,2794};    // Fa
22  int fs[5]={185,370,740,1480,2960};    // Fa#
23  int g[5]= {196,392,784,1568,3136};    // Sol
24  int gs[5]={208,415,831,1661,3322};    // Sol#
25  int a[5]= {220,440,880,1760,3520};    // La
26  int as[5]={233,466,932,1866,3729};    // La#
27  int b[5]= {247,494,988,1976,3951};    // Si
-
28  void IniciaParlante(int spk){
29    Parlante = spk;
30  }
31  void nota(int frec, int t){
32      tone(Parlante,frec);      // suena la nota frec recibida
33      delay(t);                 // para despues de un tiempo t
34  }
-
35  void HarryPotter( ){
36    nota(b[2], 500);
37    nota(e[3],1000);
38    nota(g[3], 250);
39    nota(fs[3],250);
40    nota(e[3],1000);
41    nota(b[3],500);
42    nota(a[3],1250);
43    nota(fs[3],1000);
44    nota(b[2], 500);
45    nota(e[3],1000);
46    nota(g[3],250);
47    nota(fs[3],250);
```

```
48    nota(d[3],1000);
49    nota(e[3],500 );
50    nota(b[2],1000 );
-
51    noTone(Parlante);
52    delay(1000);
53    nota(b[2], 500);
54    nota(e[3],1000);
55    nota(g[3], 250);
56    nota(fs[3],250);
57    nota(e[3],1000);
58    nota(b[3],500);
59    nota(d[4],1000);
60    nota(cs[4],500);
61    nota(c[4],1000);
62    nota(a[3],500);
63    nota(c[4],1000);
64    nota(b[3],250);
65    nota(as[3],250);
66    nota(b[2],1000);
67    nota(g[3],500);
68    nota(e[3],1000);
-
69    noTone(Parlante);
70    delay(2000);
71  }
-
72  void StarWars( ){
73  /**** tema principal ****/
74    nota(d[1],150);noTone(Parlante);delay(50);
75    nota(d[1],150);noTone(Parlante);delay(50);
76    nota(d[1],150);noTone(Parlante);delay(50);
77    nota(g[1],900);noTone(Parlante);delay(150);
78    nota(d[2],900);noTone(Parlante);delay(50);
79    nota(c[2],150);noTone(Parlante);delay(50);
80    nota(b[1],150);noTone(Parlante);delay(50);
81    nota(a[1],150);noTone(Parlante);delay(50);
82    nota(g[2],900);noTone(Parlante);delay(150);
83    nota(d[2],900);noTone(Parlante);delay(100);
84    nota(c[2],150);noTone(Parlante);delay(50);
85    nota(b[1],150);noTone(Parlante);delay(50);
86    nota(a[1],150);noTone(Parlante);delay(50);
87    nota(g[2],900);noTone(Parlante);delay(150);
88    nota(d[2],900);noTone(Parlante);delay(100);
89    nota(c[2],150);noTone(Parlante);delay(50);
90    nota(b[1],150);noTone(Parlante);delay(50);
91    nota(c[2],150);noTone(Parlante);delay(50);
92    nota(a[1],1200);noTone(Parlante);delay(2000);
-
93  /**** marcha del imperio   ****/
94    nota(g[2],500);noTone(Parlante);delay(100);
95    nota(g[2],500);noTone(Parlante);delay(100);
96    nota(g[2],500);noTone(Parlante);delay(100);
97    nota(ds[2],500);noTone(Parlante);delay(1);
```

```
98    nota(as[2],125);noTone(Parlante);delay(25);
99    nota(g[2],500);noTone(Parlante);delay(100);
100   nota(ds[2],500);noTone(Parlante);delay(1);
101   nota(as[2],125);noTone(Parlante);delay(25);
102   nota(g[2],500);noTone(Parlante);delay(2000);
103  }

104  void EntreDosAguas( ){
105   nota(a[1],400);noTone(Parlante);delay(400);
106   nota(e[1],400);noTone(Parlante);delay(400);
107   nota(a[1],400);noTone(Parlante);delay(200);
108   nota(e[1],200);noTone(Parlante);delay(200);
109   nota(a[1],200);noTone(Parlante);delay(200);
110   nota(as[1],100);noTone(Parlante);delay(100);
111   nota(b[1],400);noTone(Parlante);delay(400);
112   nota(fs[1],400);noTone(Parlante);delay(400);
113   nota(b[1],400);noTone(Parlante);delay(200);
114   nota(fs[1],200);noTone(Parlante);delay(200);
115   nota(b[1],200);noTone(Parlante);delay(200);
116   nota(as[1],100);noTone(Parlante);delay(100);
117   nota(a[1],400);noTone(Parlante);delay(400);
117   nota(e[1],400);noTone(Parlante);delay(400);
119   nota(a[1],400);noTone(Parlante);delay(400);
120  }
```

CIRCUITO PARA NUESTRO PROYECTO



Lista de Materiales: 1 Resistencia de 470Ω, 2 Cables Macho/Macho - 1 Placa Protoboard - Placa Arduino y 1 Cable USB.

-o-

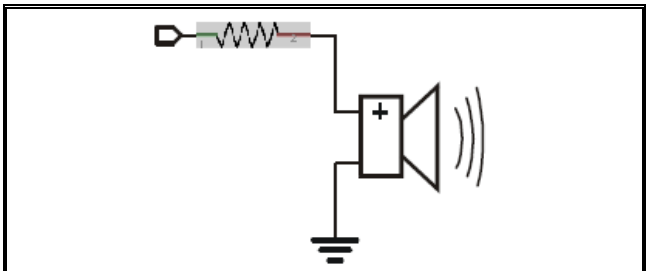


Diagrama de Conexión

Conexión de cables

Abajo, de Izquierda a Derecha: Cable Negro conectar a GND, y cable Azul, conectar al PIN configurado en el programa a través de la constante "SpeakerPin".

Guarde este programa, lo usaremos nuevamente en breve.

4- PROGRAMA QUE CONTIENE BIBLIOTECA DE SONIDOS MUY DIFUNDIDA, PRÁCTICAMENTE SE PUEDE CONSIDADERAR STANDARD.

El siguiente código usa la biblioteca: "pitches.h" que contiene todos los valores de las frecuencias de las notas típicas. Por ejemplo, NOTE_C4 es una C media. NOTE_FS4 es F aguda, y así sucesivamente. Esta tabla de notas fue originalmente escrita por Brett Hagman, en el que está basada la función "**tone**( )". La encontrarás útil cada vez que tengas que

reproducir notas musicales. Se deber tener en cuenta que la declaración de estas variables se puede hacer en el archivo principal, pero para que nuestro código principal no se vea muy extenso, se realiza de esta forma. Es para los que saben algo de Música, espero les guste.

Programa "004_Parlante_06_Biblioteca_Vector"

```
#include "pitches.h"

int PinMusical = 10;
int Cantidad    = 8;

int Melodia[Cantidad] = { NOTE_C4,
                          NOTE_G3,
                          NOTE_G3,
                          NOTE_A3,
                          NOTE_G3,
                          0,
                          NOTE_B3,
                          NOTE_C4
                        };
/* duración de las notas:
   4 = cuarto de nota,
   8 = octavo de nota, etc.*/
int NotasDuracion[Cantidad] = { 4,
                                8,
                                8,
                                4,
                                4,
                                4,
                                4,
                                4
                              };

void setup(void) {
  pinMode(PinMusical, OUTPUT);
}

void Reproduce(void){
  int Duracion;
  int PausaEntreNotas;
  int EstaNota;

  for (EstaNota = 0; EstaNota < Cantidad; EstaNota++){
    Duracion = 1000 / NotasDuracion[EstaNota];
    tone( PinMusical, Melodia[EstaNota], Duracion);
    PausaEntreNotas = Duracion * 1.30;
    delay(PausaEntreNotas);
   }
}

void loop(void) {
  Reproduce( );
  delay(1000);
}
```

Librería "**pitches.h**"

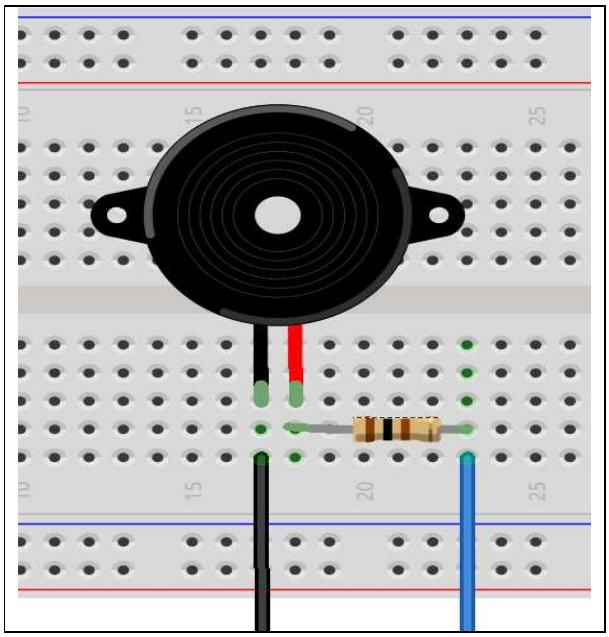
```
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
```

	/* Se deber tener en cuenta que la declaración de estas variables se pueden hacer en el archivo principal, pero para que nuestro código principal no sea muy extenso, se realiza de esta forma, practicando a la vez el uso de librerías propias. */	#define NOTE_CS5 554 #define NOTE_D5 587 #define NOTE_DS5 622 #define NOTE_E5 659 #define NOTE_F5 698 #define NOTE_FS5 740 #define NOTE_G5 784 #define NOTE_GS5 831 #define NOTE_A5 880 #define NOTE_AS5 932 #define NOTE_B5 988 #define NOTE_C6 1047 #define NOTE_CS6 1109 #define NOTE_D6 1175 #define NOTE_DS6 1245 #define NOTE_E6 1319 #define NOTE_F6 1397 #define NOTE_FS6 1480 #define NOTE_G6 1568 #define NOTE_GS6 1661 #define NOTE_A6 1760 #define NOTE_AS6 1865 #define NOTE_B6 1976 #define NOTE_C7 2093 #define NOTE_CS7 2217 #define NOTE_D7 2349 #define NOTE_DS7 2489 #define NOTE_E7 2637 #define NOTE_F7 2794 #define NOTE_FS7 2960 #define NOTE_G7 3136 #define NOTE_GS7 3322 #define NOTE_A7 3520 #define NOTE_AS7 3729 #define NOTE_B7 3951 #define NOTE_C8 4186 #define NOTE_CS8 4435 #define NOTE_D8 4699 #define NOTE_DS8 4978

Explicación Líneas Importantes	
1	El siguiente código usa la biblioteca: "**pitches.h**". Este archivo contiene todos los valores de las frecuencias de las notas típicas. Por ejemplo, NOTE_C4 es una C media. NOTE_FS4 es F aguda, y así sucesivamente. Esta tabla de notas fue originalmente escrita por Brett Hagman, en el que está basada la función "**tone( )**". La encontrarás útil cada vez que tengas que reproducir notas musicales.
2	Configuración del Pin donde se conectara el Speaker.
4 a12	Carga de un vector, con las notas Pre definidas en la biblioteca, que se reproducirán en este programa
13 a 21	Carga de un vector, con las duración de cada una de loas notas a utilizar: 4 = cuarto de nota, 8 = octavo de nota, etc
30	Para calcular la duración de la nota, se toma un segundo y se divide por el tipo de nota, por ejemplo cuarto de nota = 1000/4, octavo de nota = 1000/8, etc.

32 y 33	Para distinguir las notas, se configura un tiempo entre nota y nota
38	Este delay( ) no es necesario, solo esta por estética.

CIRCUITO PARA NUESTRO PROYECTO



Lista de Materiales: 1 Resistencia de 470Ω, 2 Cables Macho/Macho - 1 Placa Protoboard - Placa Arduino y 1 Cable USB.

-o-

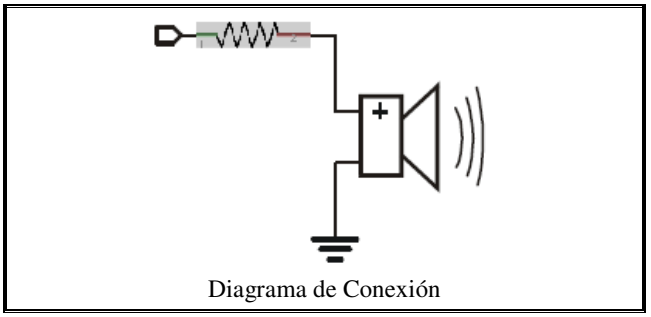


Diagrama de Conexión

Conexión de cables

Abajo, de Izquierda a Derecha: Cable Negro conectar a GND, y cable Azul, conectar al PIN configurado en el programa a través de la constante "SpeakerPin".

LIBRERIAS CON ARDUINO (Parte 2)

Como habíamos visto antes, las librerías son **segmentos de código reutilizables**, hechos por terceros o por nosotros, que guardamos aparte (en otro archivo), para ser usados cuando hagan falta en nuestros programas. Esto nos facilita mucho la programación y hace que nuestro programa sea más sencillo de hacer y de entender.

Es muy importante, antes de comprar un dispositivo o un sensor nuevo, consultar e investigar, si disponemos de las librerías que necesitamos para manejarlos. Arduino ya tiene algunas cosas incluidas, pero no tiene de todo para cualquier dispositivo.

Si Arduino no dispone de las librerías necesarias, podemos buscar en Internet, ya que existen infinidad de material (librerías desarrolladas por terceros), que nos ayudarán a conectar prácticamente cualquier dispositivo de forma muy sencilla o hacer tareas de programación complejas.
Una vez tengamos en nuestro poder estas librerías, podremos verlas, modificarlas o incluso añadir funcionalidades.

Entonces, a la hora de elegir un elemento de hardware, uno de los argumentos importantes, es la librería que nos ofrece el fabricante o la comunidad y su facilidad de uso.
Cuando adquirimos una nueva librería de algún dispositivo, generalmente nos entregan un archivo comprimido (ZIP) que pueden incluir simplemente los archivos o una carpeta que los contiene, donde encontraremos:

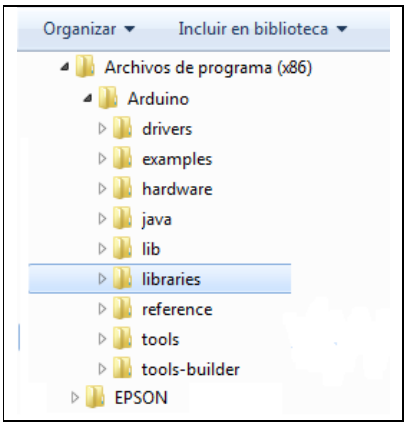
- Un archivo .cpp (código de C++). Al menos uno pero puede haber más.
- Un archivo "**.h**" o encabezado de C (header), que contiene las propiedades y métodos o funciones de la librería. Cada archivo "**.cpp**" tiene su correspondiente archivo "**.h**". Al menos uno pero puede haber más.

- Un archivo Keywords.txt, que contiene las palabras clave que se resaltan en el IDE (opcional).
- Un archivo README.md con información adicional de lo que hace y con instrucciones de como usarla. (opcional)
- Directorio denominado "**examples**" con varios sketchs de ejemplo que nos ayudará a entender cómo usar la librería (opcional).
- Adicionalmente puede haber otros archivos menos importantes como el de la licencia y otros.
- Es una Buena costumbre, que NOSOTROS agreguemos un archivo llamado "**leeme.txt**" en que nosotros escribamos el nombre del sensor o dispositivo con el cual usamos esta librería, y cualquier comentario que consideremos de interés, como referencia para el futuro. Por ejemplo la dirección de donde la tomamos.

¿Como y donde Instalar esta Librería Manualmente?

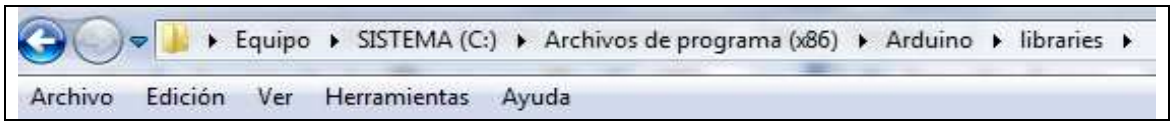
1- Una ves tengamos el Archivo comprimido en nuestra PC, simplemente extraemos los archivitos, **y si están dentro de una carpeta, así los dejamos**, pero si están sueltos, cremamos una carpeta con el mismo nombre del archivo "**.h**" y ahí dentro ponemos todos los archivos.

2- Buscar la carpeta en la que hayamos instalado el IDE de Arduino. Generalmente en "C:\Program Files\Arduino" aunque esto puede variar, ya que el instalador pregunta al usuario donde quiere instalarlo. Una ves localizado el lugar donde se instaló Arduino, copiar la carpeta de la librería dentro de "\libraries".



IMPORTANTE: La directiva **#include** existe en dos versiones. En una se pone el nombre de archivo entre comillas, en la otra entre los signos mayor y menor < >.

Acá es donde nuestro IDE buscara una librería que hayamos incluido con el signo Mayor y Menos en nuestro programa.



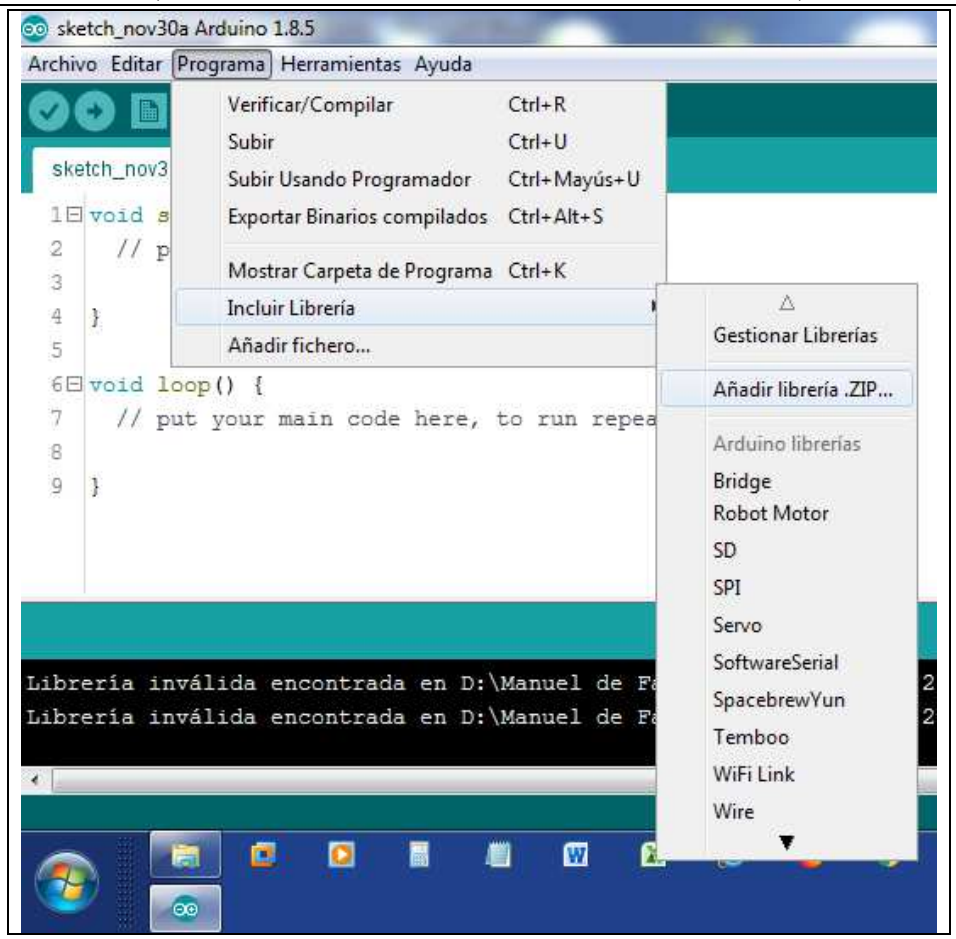
3- Es necesario reiniciar el IDE antes de poder utilizar la librería.

El propio IDE de Arduino ya trae integradas varias librerías, pero además podemos descargar otras e incorporarlas a nuestro IDE y luego usarlas en nuestros programas.
Mas adelante veremos como instalar una librería Mediante el IDE de Arduino de forma automática o Desde el gestor de librerías.

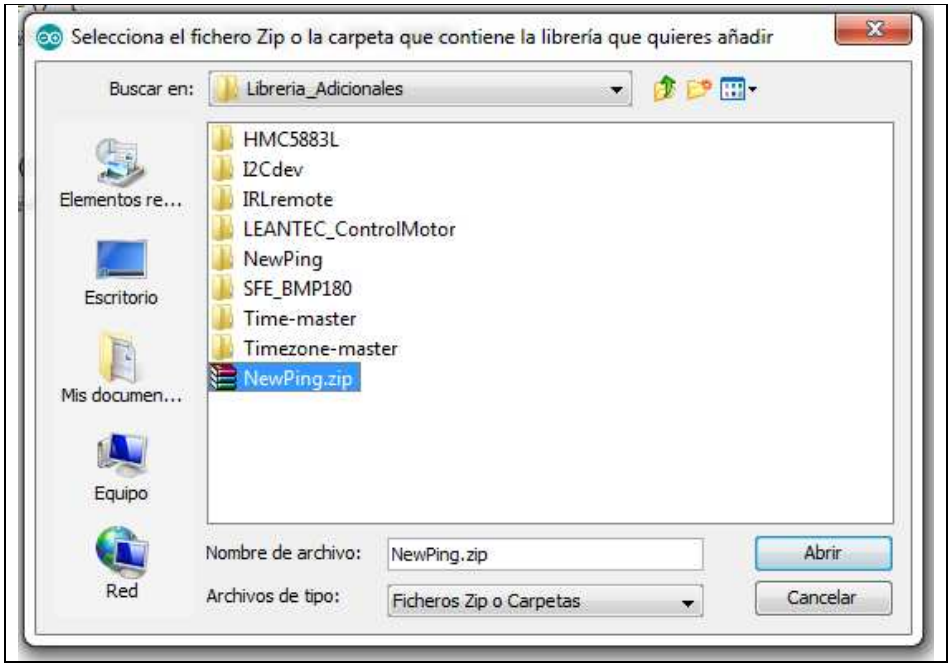
LIBRERIAS CON ARDUINO (Parte 3)

¿Como instalar una librería con el IDE?

1.- Ubicar y descargar (de Internet o donde se encuentre almacenada) la librería de arduino que requerimos y guardarla en nuestra PC. Normalmente nos encontraremos con archivos en formato ZIP.
2.- Abrir el IDE de Arduino y hacer click en el menú "**Programa**" y seleccionar la opción "**Incluir Librería**" del nuevo menú, seleccionar la opción "**Añadir Librería Zip**".



3.- Abrirá una ventana que nos permitirá encontrar el lugar en que la habíamos descargado (guardado en el punto 1). Buscar nuestra librería y seleccionarla.



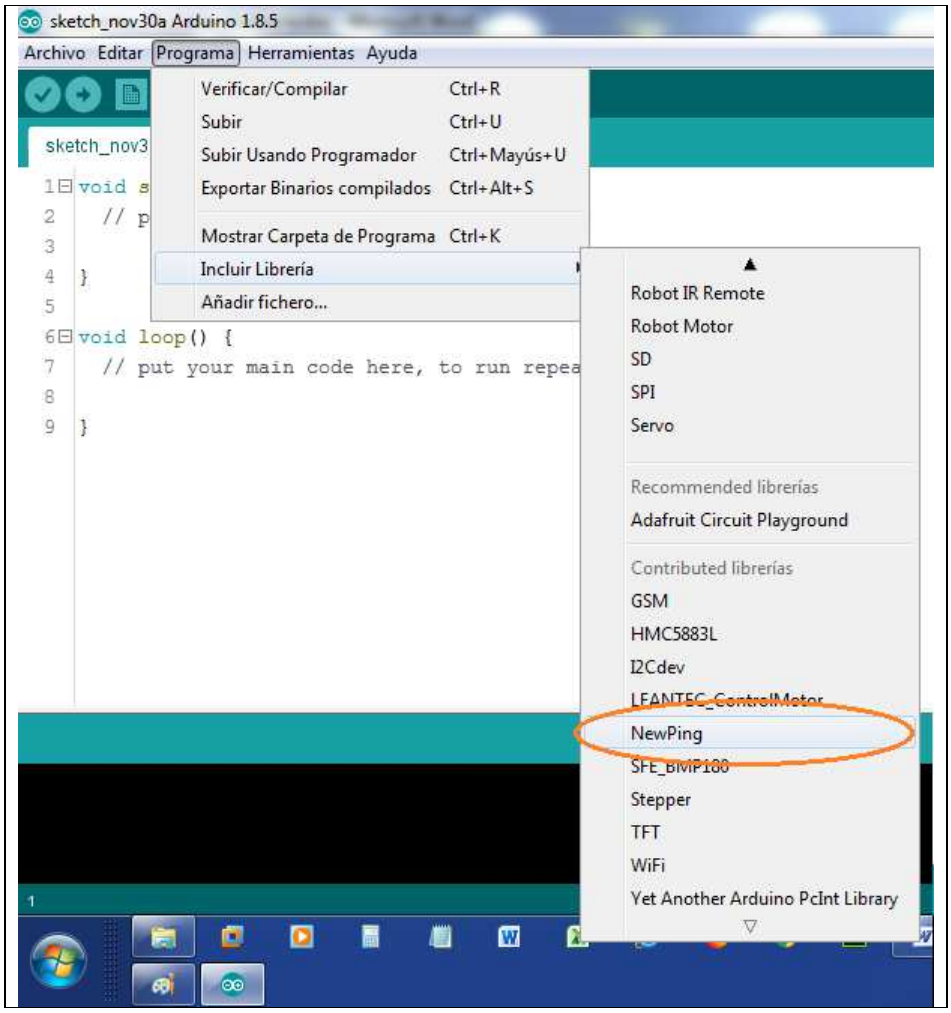
4.- Y si no hubo error en la importación, ya esta lista para ser utilizada en nuestros programas.

¿Como Usamos una librería instalada, Manualmente o con el IDE?

Utilizar una librería instalada en nuestro IDE, sin importar si se importó Manualmente o por medio del IDE. Hay dos formas de hacerlo:

a- Manualmente ponemos en la cabecera de nuestro programa **#include <NombreLibreria.h>** (tal como se explico con una librería de las nuestras.

b- Realizamos la inclusión con ayuda del IDE. Hacer click en el menú "**Programa**" y seleccionar la opción "**Incluir Librería**" del nuevo menú, seleccionar la librería que buscamos. Esto realizará el o los "**#includes**" necesarios, en la cabecera de nuestro programa.

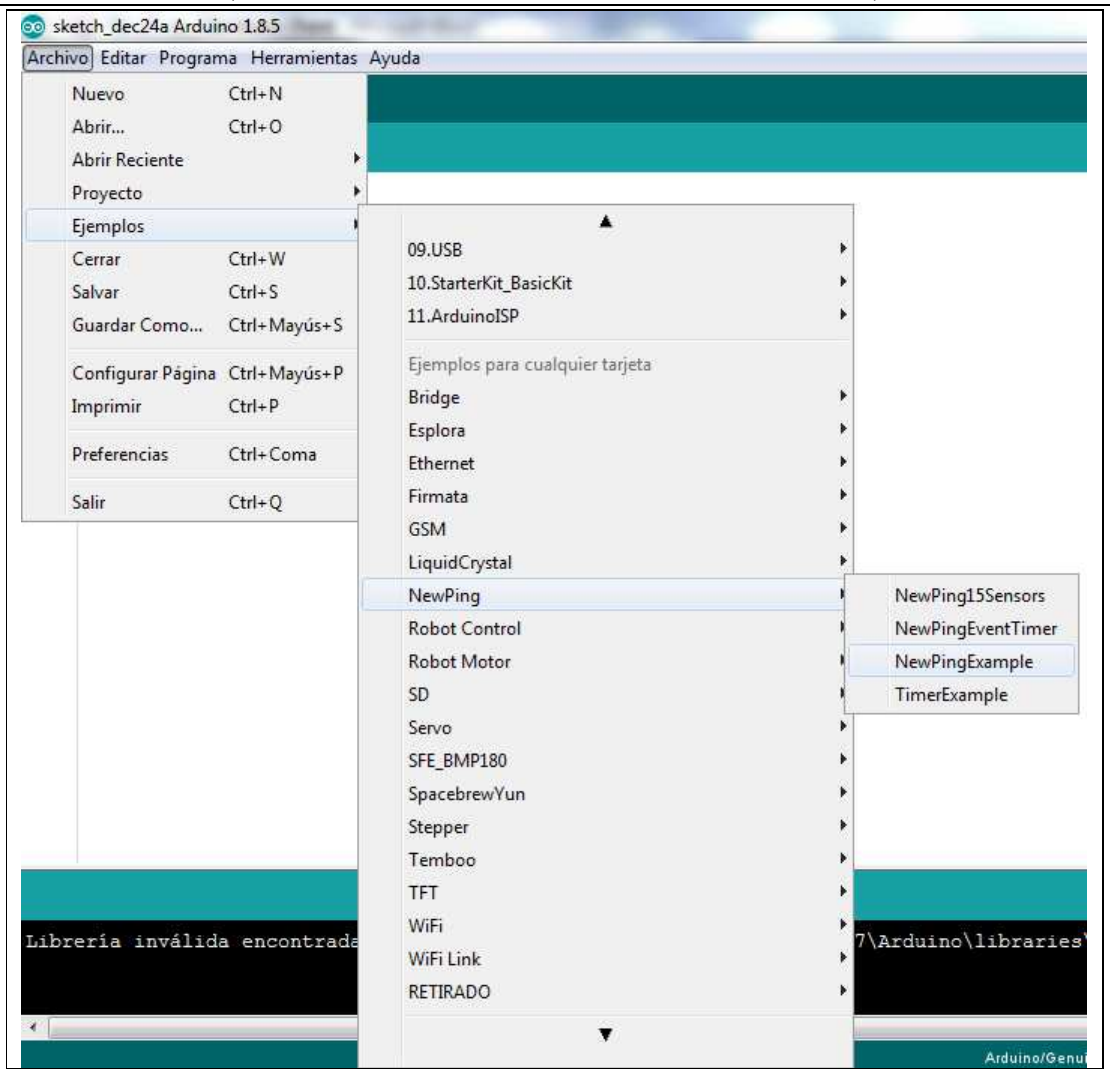


VISUALIZAR EJEMPLOS QUE CONTIENEN LAS LIBRERIAS

Visualizar los ejemplos es algo muy simple, solo hay que entender del tema para analizar los ejemplos, sin embargo, son para tener en cuenta.

Para quien empieza con Arduino, quizás estos ejemplos no sean muy prácticos, debido al grado de complejidad que muchas veces presentan, sin embargo, en poco tiempo, generalmente ya se esta en condiciones de analizarlos detalladamente.

Bueno, basta de preámbulos y veamos como ver un ejemplo. En la imagen puede visualizar como hacerlo: Menú Archivo, seleccionar la opción Ejemplos. Seleccionar la librería que queremos analizar ejemplos, elegir el ejemplo y listo.



01000101 01101110 00100000 01101101 01101001 00100000 01000011 01100001 01101100
01101101 01100001 00100000 01010011 01101111 01101100 01100101 01100100 01100001
01100100 00101100 00100000 01000101 01110011 01100011 01110101 01100011 01101000
01101111 00100000 01101100 01100001 01110011 00100000 01001101 01100101 01101100
01101111 01100100 01101001 01100001 01110011 00100000 01100100 01100101 01101100
00100000 01000001 01101100 01101101 01100001 00101110

Si tienes algunas Correcciones y/o Sugerencias, por favor contáctame.