TEMPORIZACIONES Y CONTROL DE INTERVALOS

Medir el tiempo transcurrido en un programa es muy importante, y cuanto mas preciso sea la medición mejor.

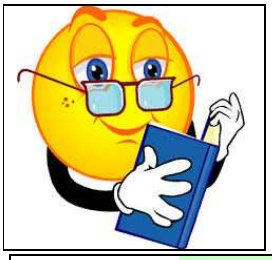
Lamentablemente Arduino, solo puede cronometrar el tiempo cuando esta funcionando, y cada vez que se apaga pierde la noción del tiempo. Aunque existen dispositivos que mantienen un reloj funcionando (con una pila), aun cuando Arduino esté apagado.

En esta guía, desarrollamos algoritmos para medir el tiempo de múltiples formas. Estos ejemplos permitirán que cada alumno, elija el método de control de tiempo que más le convenga, según sea el dispositivo que deban construir. Comenzamos desde la mas simple, pero en todos los casos, evitamos usar la función "**delay()**", ya que esta detiene la ejecución de todos los procesos en curso.

1) Mostrar por el monitor del puerto serie, los segundos transcurridos desde que Arduino comienza a funcionar.

Se les presenta dos algoritmos que muestran los segundos transcurridos desde el inicio de Arduino. Sin embargo cada uso tiene sus ventajas y limitaciones. **El primero (Algoritmo A)** es mas rápido y simple, pero es Arduino quien cuenta. **En el segundo (Algoritmo B)** Somos nosotros quien contamos, y esto nos permitirá un sin fin de soluciones a problemas que se nos presentaran.

	(Programa "060_Tiempo_01_A_Cuenta_Segundos")	(Programa "060_Tiempo_01_B_Cuenta_Segundos")
1	unsigned long TotalSegundos, Segundos;	unsigned long Ini_Tiempo, Segundos;
2		
3	void setup(){	void setup(){
4	Serial.begin(9600);	Serial.begin(9600);
5	while(!Serial);	while(!Serial);
6	Serial.println("Contador de tiempo A");	Serial.println("Contador de tiempo B");
7	}	Ini_Tiempo = millis();
8		Segundos = 0;
9	void loop(){	}
10	// Recordar que % calcula el resto de la división entera	
11	TotalSegundos = millis()/1000;	void loop(){
12	Serial.print(TotalSegundos < 10 ? "0" : "");	if((millis() - Ini_Tiempo) > 1000){
13	Serial.println(TotalSegundos);	Ini_Tiempo = millis();
14	delay(1000);	Segundos += 1;
15	}	
16		Serial.print(Segundos < 10 ? "0" : "");
17	// Dos Algoritmos para medir el tiempo.!	Serial.println(Segundos);
18		}
19		}



CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: Placa Arduino y 1 Cable USB.

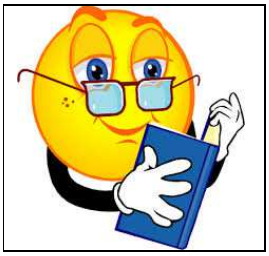
Los cables deberán ser conectados: En este ejemplo no hay cables que conectar, salvo el USB desde Arduino a La PC.

Una medición muy precisa del tiempo que un pin se encuentra activo se puede hacer mediante el manejo de hilos (del inglés thread) de procesos. Consulta con tu profesor

2) Mostrar por el monitor del puerto serie, los Días, Horas, Minutos y segundos transcurridos desde que Arduino comienza a funcionar.

Se les presenta dos algoritmos (Ampliaciones de los dos anteriores) que muestran los días, horas, minutos y segundos transcurridos desde el inicio de Arduino. Sin embargo cada uso tiene sus ventajas y limitaciones. **El primero (Algoritmo A)** es mas rápido y simple, pero es Arduino quien cuenta. **En el segundo (Algoritmo B)** Somos nosotros quien contamos, y esto nos permitirá un sin fin de soluciones a problemas que se nos presentaran.

	(Programa "060_Tiempo_02_A_Cuenta_DHMS")	(Programa "060_Tiempo_02_B_Cuenta_DHMS")
1	unsigned long totalSeconds, minutes, seconds;	unsigned long Ini_Tiempo, Segundos;
2	unsigned int hours;	unsigned long Minutos, Seg_Anterior;
3	int dias;	unsigned int Horas;
4		int Dias;
5	void setup(){	
6	Serial.begin(9600);	void setup(){
7	while(!Serial);	Serial.begin(9600);
8	Serial.println("Contador de tiempo (D:H:M:S)");	while(!Serial);
9	}	Serial.println("Contador de tiempo (D:H:M:S)");
10		Ini_Tiempo = millis();
11	void loop(){	}
12		
13	// Recordar que % calcula el resto de la división entera	void loop(){
14	totalSeconds = millis()/1000;	Seg_Anterior = Segundos;
15		Segundos = (millis() - Ini_Tiempo)/1000;
16	// Guarda Cantidad de Horas Transcurridas	if(Segundos >= 60){
17	hours = totalSeconds / 3600;	Ini_Tiempo = millis();
18		Minutos += (int) Segundos/60;
19	// Descarta Horas y Guarda Minutos	Segundos = Segundos % 60;
20	minutes = (totalSeconds % 3600) / 60;	}
21		if(Minutos >= 60){
22	//Descarta los segundos que no completan el minuto	Horas += (int) Minutos/60;
23	seconds = totalSeconds % 60;	Minutos = Minutos % 60;
24		}
25	// Guarda los dias completos	if(Horas >= 24){
26	dias = (int) hours / 24;	Dias += (int) Horas/24;
27		Horas = Horas % 24;
28	Serial.print(dias < 10 ? "0" : "");	}
29	Serial.print(dias);	
30	Serial.print(":");	if(Seg_Anterior != Segundos){
31	Serial.print(hours < 10 ? "0" : "");	Serial.print(Dias < 24 ? "0" : "");
32	Serial.print(hours);	Serial.print(Dias);
33	Serial.print(":");	Serial.print(":");
34	Serial.print(minutes < 10 ? "0" : "");	Serial.print(Horas < 10 ? "0" : "");
35	Serial.print(minutes);	Serial.print(Horas);
36	Serial.print(":");	Serial.print(":");
37	Serial.print(seconds < 10 ? "0" : "");	Serial.print(Minutos < 10 ? "0" : "");
38	Serial.println(seconds);	Serial.print(Minutos);
39	delay(1000);	Serial.print(":");
40	}	Serial.print(Segundos < 10 ? "0" : "");
41		Serial.println(Segundos);
42	// Dos Algoritmos para medir el tiempo.!	}
43		}



CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: Placa Arduino y 1 Cable USB.

Los cables deberán ser conectados: En este ejemplo no hay cables que conectar, salvo el USB desde Arduino a La PC.

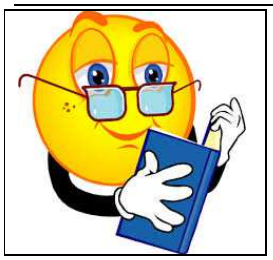
Una medición muy precisa del tiempo que un pin se encuentra activo se puede hacer mediante el manejo de hilos (del inglés thread) de procesos. Consulta con tu profesor

3) Hacer un programa que permita configurar un intervalo de tiempo y muestre segundo a segundo como decrece. Al finalizar el intervalo, debe recomenzar.

Este programa, permitiría desparar en evento o rutina determinada cada cierto intervalo de tiempo.

(Programa "060_Tiempo_03_Marca_Intervalos")

```
1   unsigned long Inicia_Segundos,
2                 IntervaloSegundos,
3                 Pasaron,
4                 RegresivaSegundos,
5                 Anterior_RegresivaSegundos;
6
7   int Mostrar_Tiempo(){
8     int ok=0;
9     if(RegresivaSegundos != Anterior_RegresivaSegundos) ok=1;
10    return (ok);
11  }
12
13  unsigned long Intervalo(){
14      Pasaron =  millis()/1000 - Inicia_Segundos;
15      Anterior_RegresivaSegundos = RegresivaSegundos;  // Para saber cuando Mostrar el Tiempo
16      RegresivaSegundos = IntervaloSegundos - Pasaron;
17      if( Pasaron >= IntervaloSegundos){
18          Inicia_Segundos = millis()/1000;
19          RegresivaSegundos = 0;    // Aseguro que retorne Cero
20      }
21      return (RegresivaSegundos);
22  }
23
24  void Inicia_Intervalo(unsigned int H, unsigned long M, unsigned long S){
25      IntervaloSegundos = H*3600 + M*60 + S;
26      Inicia_Segundos = millis()/1000;
27      Serial.print("Iniciando Intervalo de ");
28      Serial.print(IntervaloSegundos);
29      Serial.println(" Segundos");
30  }
31
32  void setup(){
33      Serial.begin(9600);
34      while(!Serial);
35      Inicia_Intervalo(0,1,10);
36  }
37
38  unsigned long Queda;
39
40  void loop(){
41    Queda = Intervalo();
42
43    if(Mostrar_Tiempo()) Serial.println(Queda);
44
45    if(Queda == 0){
46       Inicia_Intervalo(0,1,10);
47     }
48  }
```



CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: Placa Arduino y 1 Cable USB.

Los cables deberán ser conectados: En este ejemplo no hay cables que conectar, salvo el USB desde Arduino a La PC.

Una medición muy precisa del tiempo que un pin se encuentra activo se puede hacer mediante el manejo de hilos (del inglés thread) de procesos. Consulta con tu profesor

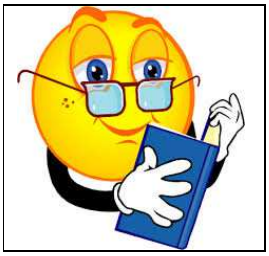
4) Hacer un programa que permita configurar diferentes intervalos de tiempo simultáneamente y muestre segundo a segundo como decrece cada uno. Al finalizar cada intervalo, debe recomenzar independientemente de los otros.

Este programa, permitiría disparar varios eventos o rutinas a intervalos de tiempo distintos e independientes entre si, reiniciando cada intervalo independientemente cada vez que sea requerido. *En este programa evito usar Clases, ya que con una estructura es suficiente y por ahora más claro.*

(Programa "060_Tiempo_04_Marca_Multiples_Intervalos")

```
1   struct Intervalo{
2     unsigned long IntervaloSegundos,
3                   Pasaron,
4                   RegresivaSegundos,
5                   Inicia_Segundos;
6
7     void Inicia(unsigned long S);
8     void Inicia(unsigned int H, unsigned long M, unsigned long S);
9     unsigned long Regresivo();
10  };
11
12   void Intervalo::Inicia(unsigned long S){
13     Inicia(0,0,S);
14  }
15
16  void Intervalo::Inicia(unsigned int H, unsigned long M, unsigned long S){
17     Inicia_Segundos =  millis()/1000;
18     IntervaloSegundos =   H*3600 + M*60 + S;
19  }
20
21  unsigned long Intervalo::Regresivo(){
22     Pasaron = millis()/1000 - Inicia_Segundos;
23     RegresivaSegundos = IntervaloSegundos - Pasaron;
24     if( Pasaron >= IntervaloSegundos){
25        Inicia_Segundos = millis()/1000;
26        RegresivaSegundos = 0; // Aseguro que retorne Cero
27     }
28     return(RegresivaSegundos);
29  }
30
31  Intervalo A,B,C;
32
33  void setup(){
34     Serial.begin(9600);
35     while(!Serial);
36     A.Inicia(15);
37     B.Inicia(23);
38     C.Inicia(0, 1, 6);
39  }
40
```

```
41  unsigned long T1,T2,T3;
42
43  void loop(){
44    T1 = A.Regresivo();
45    Serial.print("T1: ");
46    Serial.print(T1);
47
48    T2 = B.Regresivo();
49    Serial.print(" T2: ");
50    Serial.print(T2);
51
52    T3 = C.Regresivo();
53    Serial.print(" T3: ");
54    Serial.println(T3);
55
56    if(T1 == 0){
57      Serial.println("1- Termino Intervalo");
58      A.Inicia(15);
59    }
60    if(T2 == 0){
61      Serial.println("2- Termino Intervalo");
62      B.Inicia(23);
63    }
64    if(T3 == 0){
65      Serial.println("3- Termino Intervalo");
66      C.Inicia(0, 1, 6);
67    }
68  }
```



CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: Placa Arduino y 1 Cable USB.

Los cables deberán ser conectados: En este ejemplo no hay cables que conectar, salvo el USB desde Arduino a La PC.

Una medición muy precisa del tiempo que un pin se encuentra activo se puede hacer mediante el manejo de hilos (del inglés thread) de procesos. Consulta con tu profesor

5) Medir el tiempo que un botón se encuentra presionado.

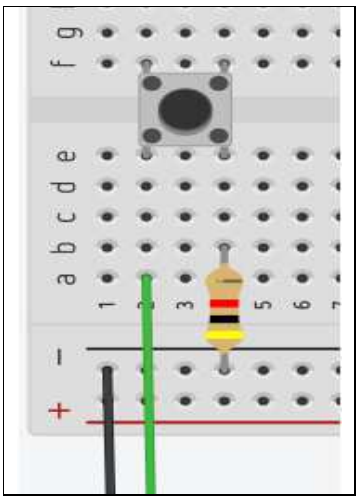
En este programa intentaremos medir el tiempo (segundos, minutos, horas, días) que un botón se encuentra presionado por el operador. Por demás esta decir que en una aplicación real, el solo hecho de presionar un botón activaría algún dispositivo.

(Programa "060_Tiempo_05_Cuenta_el_Tiempo_Que_Un_PIN_esta_Activo")

```
1   const int Pin_Boton = 5;
2   unsigned long T_Actual,
3                 T_Previo,
4                 T_Segundos;
5
6   unsigned long Dias,
7                 Horas,
8                 Minutos,
9                 Segundos;
10
11  unsigned long EstadoBoton;
12
13  void setup(){
14     Serial.begin(9600);
15     while(!Serial);
16     Serial.print("Cuanta Tiempo Activo del Pin: ");
```

```
17      Serial.println(Pin_Boton);
18
19      pinMode(Pin_Boton,  INPUT_PULLUP);
20      T_Segundos = 0;
21      T_Previo   = 0;
22    }
23
24  long int Cuenta_Tiempo_Activo_del_PIN(int PIN){
25      T_Actual = millis();
26      if( digitalRead(PIN)== LOW && T_Actual - T_Previo >= 1000 ){
27          T_Previo = T_Actual;
28          T_Segundos += 1;
29
30          Horas    =   T_Segundos / 3600;             // Calcula Cantidad de Horas Transcurridas
31          Minutos  =  (T_Segundos % 3600) / 60;  // Descarta Horas y Guarda Minutos
32          Segundos =   T_Segundos % 60;           //Descarta todo menos los segundos que no completan el minuto
33          Dias     =  (int) Horas / 24;
34
35          Serial.print(Dias < 10 ? "0" : "");
36          Serial.print(Dias);
37          Serial.print(":");
38          Serial.print(Horas < 10 ? "0" : "");
39          Serial.print(Horas);
40          Serial.print(":");
41          Serial.print(Minutos < 10 ? "0" : "");
42          Serial.print(Minutos);
43          Serial.print(":");
44          Serial.print(Segundos < 10 ? "0" : "");
45          Serial.println(Segundos);
46      }
47      return(T_Segundos); // No es obligatorio tomar este valor fuera de la función
48  }
49
50  void loop(){
51    Cuenta_Tiempo_Activo_del_PIN(Pin_Boton);
52    EstadoBoton = digitalRead(Pin_Boton);
53  }
```

CIRCUITO PARA NUESTRO PROYECTO



Lista de Materiales: 1 Botón Pulsador (de 2 o 4 Patas) - 1 Resistencia de 470Ω - 1 Protoboard - 2 Cables de conexión Macho/Macho - Placa Arduino y 1 Cable USB.

Los cables deberán ser conectados: De Izquierda a Derecha. Cable Negro: GND común de Arduino. Cable Verde: al Pin configurado para el Botón (en este caso el 5).

Una medición muy precisa del tiempo que un pin se encuentra activo se puede hacer mediante el manejo de hilos (del inglés thread) de procesos. Consulta con tu profesor

6) Medir el tiempo de la Respuesta Humana. El test consiste en lo siguiente: Se prende un Led, y en cuanto el participante lo ve, debe apagarlo lo antes posible, presionando un Botón.

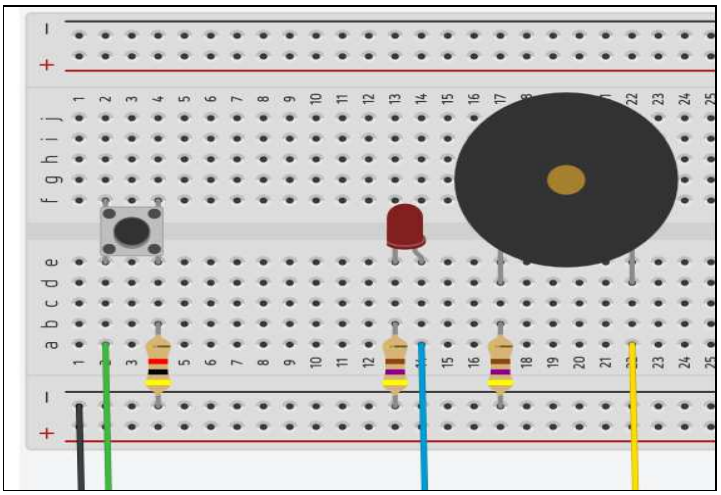
En este programa intentaremos medir el tiempo en Milisegundos, que una persona puede apagar un led, desde el momento que lo ve prendido.

(Programa "060_Tiempo_06_Medir_Tiempo_Respuesta_Humano")

```
1   const int     Pin_Led   = 8;
2   const int     Pin_Boton = 5;
3   unsigned long Tiempo_Max_Encendido = 5000,
4                     TiempoInicio,
5                     TiempoTranscurrido;
6
7   int           EstadoBoton;
8
9   const int   SpeakerPin = 10;
10  double      TiempoEsperaSpeaker = 500;
11  int           Frecuencia =3500;
12
13  void setup(){
14   Serial.begin(9600);
15   while(!Serial);
16   Serial.println("Inicia Programa - Tiempo Respuesta Humana\n");
17
18   pinMode(Pin_Led, OUTPUT);
19   digitalWrite(Pin_Led, LOW);
20
21   pinMode(SpeakerPin, OUTPUT);
22   pinMode(Pin_Boton,  INPUT_PULLUP);
23  }
24
25  void AlarmaSpeaker(){
26      tone(SpeakerPin, Frecuencia);
27      delay(TiempoEsperaSpeaker);
28      noTone(SpeakerPin);               //Libero SpeakerPin
29      delay(TiempoEsperaSpeaker/4);
30  }
31
32  void loop(){
33   do{
34      if( digitalRead(Pin_Boton) == LOW){
35        Serial.println("El Juego todavia no ha comenzado - Suelte el Boton.");
36        AlarmaSpeaker();
37        }
38   } while( digitalRead(Pin_Boton) == LOW);
39
40   Serial.println("====== Inicia Juego ======");
41   digitalWrite(Pin_Led, HIGH);
42   TiempoInicio = millis();
43
44   do {
45     EstadoBoton = digitalRead(Pin_Boton);
46     TiempoTranscurrido = millis() - TiempoInicio;
47   }while( EstadoBoton == HIGH && Tiempo_Max_Encendido > TiempoTranscurrido);
48
49   digitalWrite(Pin_Led, LOW);
50
51   if(TiempoTranscurrido < Tiempo_Max_Encendido) {
52    Serial.print("Se Pulsa Botón tras ");
53    Serial.print(TiempoTranscurrido);
54    Serial.println(" milisegundos Transcurridos");
55   } else {
56    Serial.print("Transcurrieron ");
57    Serial.print(Tiempo_Max_Encendido);
```

```
58      Serial.println(" Segundos sin pulsar Boton.");
59    }
60    delay(2000);
    }
```

CIRCUITO PARA NUESTRO PROYECTO



Lista de Materiales: 1 Botón Pulsador (de 2 o 4 Patas) – 1 Speaker - 1 Led - 3 Resistencias de 470Ω - 1 Protoboard - 4 Cables de conexión Macho/Macho - Placa Arduino y 1 Cable USB.

Los cables deberán ser conectados: De Izquierda a Derecha. Cable Negro: GND común de Arduino. Cable Verde: al Pin configurado para el Botón (en este caso Pin 5). Cable Azul: Al Pin Configurada para el Led (En este caso el 8). Cable Amarillo, al Pin Digital que se haya configurado el parlante o Speaker (en este caso el 10).

Quieres Más?

Si este tema realmente te interesa, te sugiero, veas la partes de Acceso a Ethernet / Internet (Está junto con el uso del **Ethernet Shield W5100).**

Ahí encontrarás los ejemplos donde te conectas a un servidor remoto, le pides Fecha y Hora y actualizas un reloj en nuestra placa Arduino.

Mira También donde se explica como Reiniciar Arduino.

01000101 01110011 00100000 01110010 01101001 01100100 11000011 10101101 01100011
01110101 01101100 01101111 00100000 01110110 01101001 01110110 01101001 01110010
00100000 00110001 00110000 00110000 00100000 01100001 11000011 10110001 01101111
01110011 00100000 01111001 00100000 01110011 11000011 10110011 01101100 01101111
00100000 01110011 01100101 01110010 00100000 01100011 01100001 01110000 01100001
01100011 01100101 01110011 00100000 01100100 01100101 00100000 01110010 01100101
01100011 01101111 01110010 01100100 01100001 01110010 00100000 00110011 00110000
00100000 01101101 01101001 01101100 01101100 01101111 01101110 01100101 01110011

00100000 01100100 01100101 00100000 01100010 01111001 01110100 01100101 01110011
00101110 00100000 01001111 00100000 01110011 01100101 01100001 00101100 00100000
01101101 01100101 01101110 01101111 01110011 00100000 01110001 01110101 01100101
00100000 01110101 01101110 00100000 01100011 01101111 01101101 01110000 01100001
01100011 01110100 00100000 01100100 01101001 01110011 01100011 00101110 00100000
01001100 01100001 00100000 01100011 01101111 01101110 01100100 01101001 01100011
01101001 11000011 10110011 01101110 00100000 01101000 01110101 01101101 01100001
01101110 01100001 00100000 01110011 01100101 00100000 01101000 01100001 01100011
01100101 00100000 01101101 11000011 10100001 01110011 00100000 01101111 01100010
01110011 01101111 01101100 01100101 01110100 01100001 00100000 01100011 01100001
01100100 01100001 00100000 01101101 01101001 01101110 01110101 01110100 01101111

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