Acordeón Arduino

Tomado mayormente de la Referencia de Lenguaje Arduino:
http://arduino.cc/en/Reference/HomePage

Estructura y flujo

Estructúra básica del programa

```
void setup() {
  // Corre una vez cuando el
  // programa inicia
}
void loop() {
  // Se ejecuta repetidamente
```

Estructuras de control

```
if (x < 5) { ... } else { ... }
while (x < 5) { ... }
do { ... } while ( x < 5);
for (int i = 0; i < 10; i++) { ... }
break;  //sale del bucle inmediatamente
continue; //va a la siguiente iteración
switch (miVariable) {
   case 1:
     ...
     break;
   case 2:
     ...
     break;
   default:
     ...
}
return x; // o "return;" para vacíos
```

Operadores

Operadores generales

```
=  (operador de asignación)
+  (adición)     -  (sustracción)
*  (multiplicación)
/  (división)    %  (módulo)
== (igual a)     != (desigual a)
<  (menor que)   >  (mayor que)
<= (igual o menor que)
>= (mayor o igual que)
&& (y)    || (ó)    ! (negación)
```

Operadores compuestos

```
++ (incremento)
-- (decremento)
+= (suma compuesta)
-= (resta compuesta)
*= (multiplicación compuesta)
/= (división compuesta)
&= (AND binario compuesto)
|= (OR binario compuesto)
```

Operadores a nivel de bit

```
& (AND binario)    | (OR binario)
^ (XOR binario)    ~ (NOT binario)
<< (desplazamiento a la izquierda)
>> (desplazamiento a la derecha)
```

Variables, vectores y datos

Tipos de datos

```
void      vacío
boolean   (0, 1, true, false)
char      (ej. 'a'  -128 a 127)
int       (-32768 a 32767)
long      (-2147483648 a 2147483647)
unsigned char  (0 a 255)
byte           (0 a 255)
unsigned int   (0 a 65535)
word           (0 a 65535)
unsigned long  (0 a 4294967295)
float     (-3.4028e+38 a 3.4028e+38)
double    (igual que los flotantes)
```

Calificadores

```
static   //persiste entre llamadas
volatile //usa la RAM
const    //sólo lectura
PROGMEM  //usar la flash
```

Vectores y matrices

```
int myInts[6]; //vector de 6 enteros
int myPins[]={2, 4, 8, 3, 6};
int mySensVals[6]={2, 4, -8, 3, 2};
myInts[0]=42;  //asigna al primero
               //en el índice
myInts[6]=12;  //ERROR! El índice va
               //de 0 a 5
```

Constants

```
HIGH | LOW
INPUT | OUTPUT
true | false
143       //Decimal
0173      //Octal (comenzando en 0)
0b11011111 //Binario
0x7B      //Hex (hexadecimal)
7U        //forzar unsigned
10L       //forzar long
15UL      //forzar long unsigned
10.0      //forzar floating point
2.4e5     //240000
```

Punteros

```
&  (referencia: obtener puntero)
*  (valor: seguir puntero)
```

Cadenas

```
char S1[8] =
 {'A','r','d','u','i','n','o'};
 //cadena sin terminación
 //puede producir error
char S2[8] =
 {'A','r','d','u','i','n','o','\0'};
 //incluye terminación nula \0
char S3[]="arduino";
char S4[8]="arduino";
```

Funciones incluídas

E/S Digital

```
pinMode(pin,[INPUT, OUTPUT])
digitalWrite(pin, valor)
int digitalread(pin)
  //Escribe HIGH en entradas para
  //usar los pull-ups
```

E/S Analógicas

```
analogReference([DEFAULT,
    INTERNAL, EXTERNAL])
int analogRead(pin)
analogWrite(pin, valor)  //PWM
```

Advanced I/O

```
tone(pin, freqhz)
tone(pin, freqhz, duracion_ms)
noTone(pin)
shiftOut(pinDatos, pinReloj,
  [MSBFIRST,LSBFIRST], valor)
unsigned long pulseIn(pin,
  [HIGH,LOW])
```

Tiempo

```
unsigned long millis()
  //desbordamiento en 50 dias
unsigned long micros()
  //desbordamiento en 70 minutos
delay(ms)
delayMicroseconds(us)
```

Matemáticas

```
min(x, y)   max(x, y)   abs(x)
sin(rad)    cos(rad)    tan(rad)
sqrt(x)     pow(base, exponente)
constrain(x, valMin, valMax)
map(val, deBAJO, deALTO,
aBAJO,aAlto)
```

Números aleatorios

```
randomSeed(semilla) //long ó int
long random(max)
long random(min, max)
```

Bits y Bytes

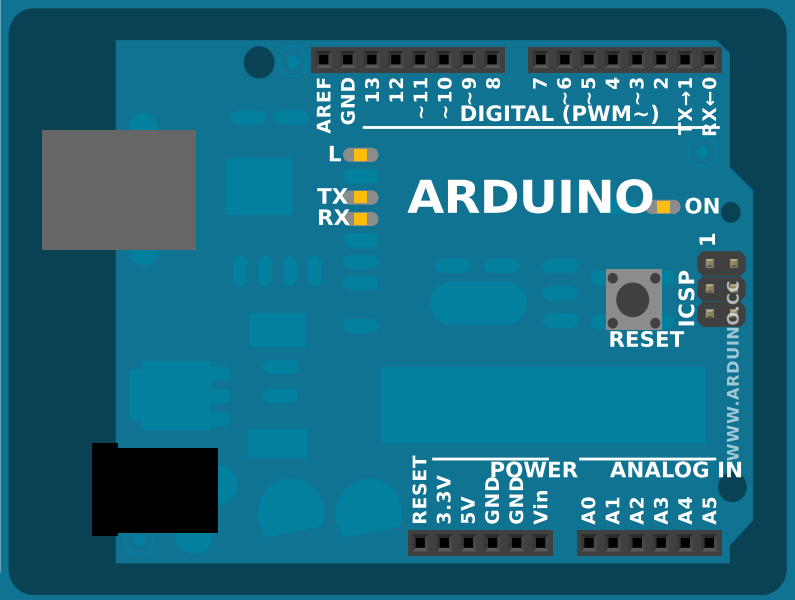
```
lowByte(x)    highByte(x)
bitRead(x, bitn)
bitWrite(x, bitn, bit)
bitSet(x, bitn)
bitClear(x, bitn)
bit(bitn)  // bitn: 0=LSB 7=MSB
```

Conversiones

```
char()        byte()
int()         word()
long()        float()
```

Interrupciones Externas

```
attachInterrupt(interrup, func,
 [LOW, CHANGE, RISING, FALLING])
detachInterrupt(interrupción)
interrupts()
noInterrupts()
```



Bibliotecas

Serie

```
begin([300, 1200, 2400, 4800,
       9600, 14400, 19200, 28800,
       38400, 57600, 115200])
      //Puede ser cualquier número
end()
int available()
byte read()
byte peek()
flush()
print(misDatos)
println(misDatos)
write(misBytes)
```

EEPROM (#include <EEPROM.h>)

```
byte read(dirInterna)
write(dirInterna, miByte)
```

Servo (#include <Servo.h>)

```
attach(pin, [min_uS, max_uS])
write(ángulo) // 0, 180
writeMicroseconds(uS)
     //1000-2000; 1500 es en medio
read()       //0 - 180
attached()   //regresa boleano
detach()
```

SoftwareSerial(RxPin, TxPin)

```
  (#include <softwareSerial.h>)
begin(long velocidad) //hasta 9600
char read()     //espera los datos
print(misDatos)
println(misDatos)
```

Wire (#include <Wire.h>) //para I²C

```
begin()        //se une a maestro
begin(addr)    //se une a esclavo @dir
requestFrom(dirección, cuenta)
beginTransmission(dir)  // Paso 1
send(miByte)            // Paso 2
send(char * miCadena)
send(byte * datos, tamaño)
endTransmission()       // Paso 3
byte available()   // Num de bytes
byte receive() //Regresa el sig byte
onReceive(manejador)
onRequest(manejador)
```


por Mark Liffiton

Traducción de Antonio Maldonado

Adaptado de:
- Idea original por Gavin Smith
- Versión SVG por Frederic Dufourg
- Dibujo del Arduino de Fritzing.org