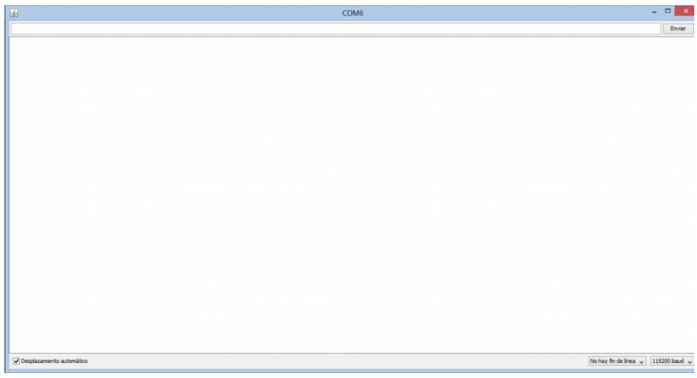
Comunicación Serial con Arduino

Por Antony García González - Enero 27, 2013

Arduino posee como principal característica la habilidad de comunicarse con nuestra computadora a través del puerto serie. Esto se conoce como comunicación serial. Debido a que el uso de este puerto ha quedado un poco en desuso a favor de la tecnología USB, Arduino cuenta con un convertidor de Serial a USB que permite a nuestra placa ser reconocida por nuestra computadora como un dispositivo conectado a un puerto COM aún cuando la conexión física sea mediante USB.

Arduino IDE nos proporciona una herramienta que nos permite enviar y visualizar los datos que se manejan a través del puerto Serie. Dicha herramienta se conoce como Monitor Serial y se puede encontrar en el menú de herramientas, en la opción "Monitor Serial". Es la forma más simple que existe para establecer la comunicación serial con Arduino.



A través de esta ventana se puede enviar o recibir información utilizando el puerto serie. Nótese que para poder abrir esta ventana es necesario que tengamos nuestra placa Arduino conectada a nuestra PC mediante USB.
Para iniciar la comunicación serial con Arduino utilizando el Monitor Serial debemos establecer algunos comandos en el Arduino IDE y luego subirlos al microcontrolador.

El código es el siguiente:

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

void loop(){
  Serial.println('1');
  delay(1000);
}
```

En la función **setup** inicializamos la comunicación serial con la sentencia Serial.begin(9600).

El 9600 indica el baud rate, o la cantidad de baudios que manejará el puerto serie. Se define baudio como una unidad de medida, usada en telecomunicaciones, que representa el número de símbolos por segundo en un medio de transmisión ya sea analógico o digital.

Para nuestros propósitos utilizaremos comúnmente una velocidad de símbolo de 9600.

Siempre que vayamos a comunicarnos con Arduino vía puerto serie se necesita invocar la sentencia Serial.begin(9600).

Ahora, en la función **loop** nos encontramos con una sentencia interesante: Serial.println('1').

Cuando usamos println le estamos diciendo al microcontrolador que tendrá que imprimir un caracter a través del puero serie. Hay otros métodos como **Serial.print** o **Serial.write** que nos sirven para imprimir o escribir en el puerto serie, sin embargo el método **Serial.println** agrega un salto de línea cada vez que se envía un dato, lo que es provechoso para nosotros en algunos casos, especialmente cuando utilizamos el monitor serial.

En el método **Serial.println** se debe establecer lo que se quiere imprimir entre paréntesis y con comillas. En este caso vamos a imprimir el caracter '1'. He colocado un delay de 1 segundo para que la impresión en el Monitor Serial sea pausada y se produzca de segundo en segundo.

Si subimos el código a nuestra placa, podremos observar el comportamiento de nuestro algoritmo a través del monitor serial.

Como podemos observar, en el monitor serial empieza a aparecer el '1' que le indicamos a nuestro microcontrolador que debía imprimir. En la placa, el LED con la inscripción TX parpadea a medida que aperecen los caracteres en el monitor serial. El LED TX indica cuándo el Arduino está haciendo un envío de datos a través del puerto serie.

De esta manera podemos obtener información desde el micricontrolador. Esto es aprovechado por nosotros para obtener datos desde un sensor o algún dispositivo de conmutación.

Ahora que sabemos como enviar datos desde Arduino a través del puerto Serie, podemos pasar al siguiente paso que es hacer que Arduino reciba datos seriales y los interprete. Dichos datos los enviaremos a través del monitor serial utilizando el teclado de nuestra computadora.

El código será el siguiente:

```
int input;

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

void loop(){
  if (Serial.available()>0){

    input=Serial.read();

    Serial.println(input);

  }
}
```

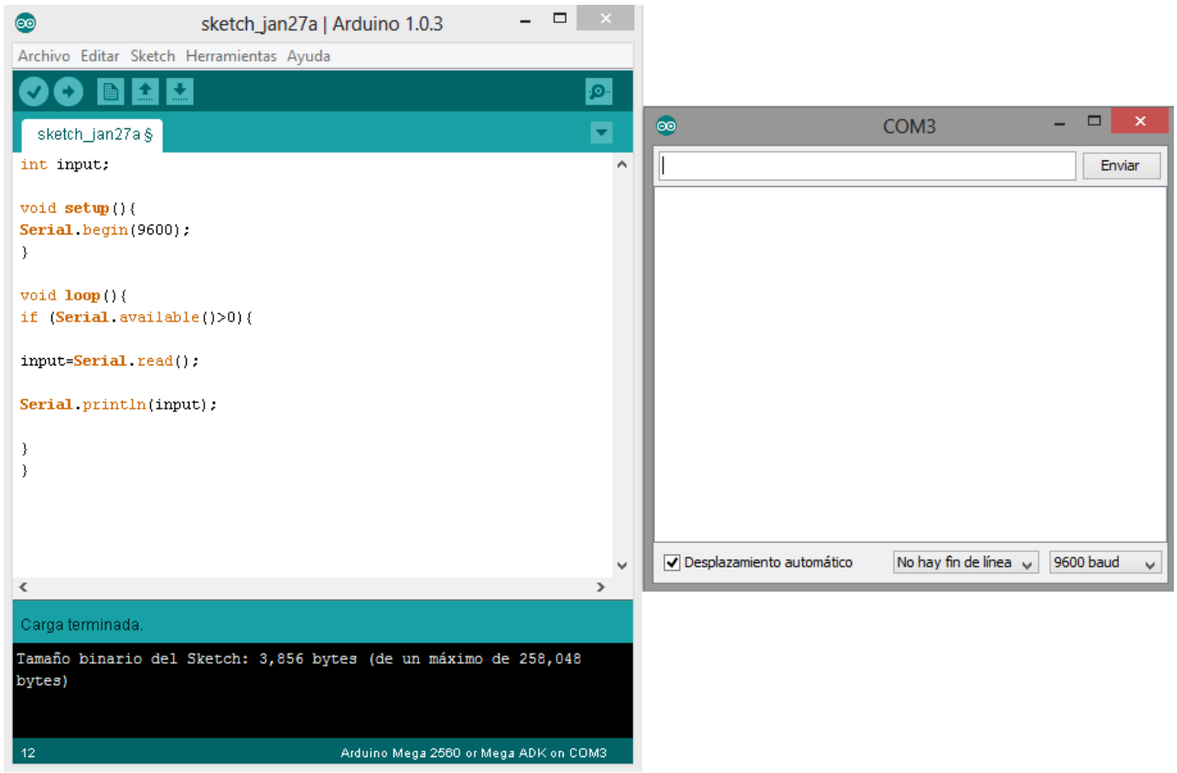
La función **setup** permanece igual. Al principio del código se declara una variable **tipo int** llamada **input.** En ella se guardará los valores que obtendremos a través del puerto serie.

En la función **loop** es donde está la parte importante. La línea que dice **Serial.available()>0** es sumamente importante ya que limita la ejecución del código solamente cuando haya datos disponibles en el puerto serie. Está metida dentro de una estructura selectiva (**if**). Cuando haya datos disponibles para leer en el puerto serie, el método **Serial.available()** tendrá un número mayor a cero (>0) por lo que cumplirá con la condición establecida en la estructura selectiva y el programa procederá a ejecutar los comandos que estén dentro de las llaves ({}) del **if**.

Cuando entre en la estructura selectiva, a la variable **input** se le asignará un valor que se lea desde el puerto serie mediante el método **Serial.read().** Siempre que necesitamos leer algo lo hacemos mediante este método.

Luego de asignarle a **input** un valor, haremos que el microcontrolador nos devuelva por el puerto serie el mismo valor que le enviamos desde la computadora. Para ello imprimimos el valor de **input** mediante el método **Serial.println(input)**.

Veamos como funciona el código cuando lo subimos a la placa.



Abrimos el monitor serial y en la ventana de comandos escribimos 1 y le damos clic en enviar. Luego enviamos 2, 3, 4 y 5.

Nos aparecerá lo siguiente:

49

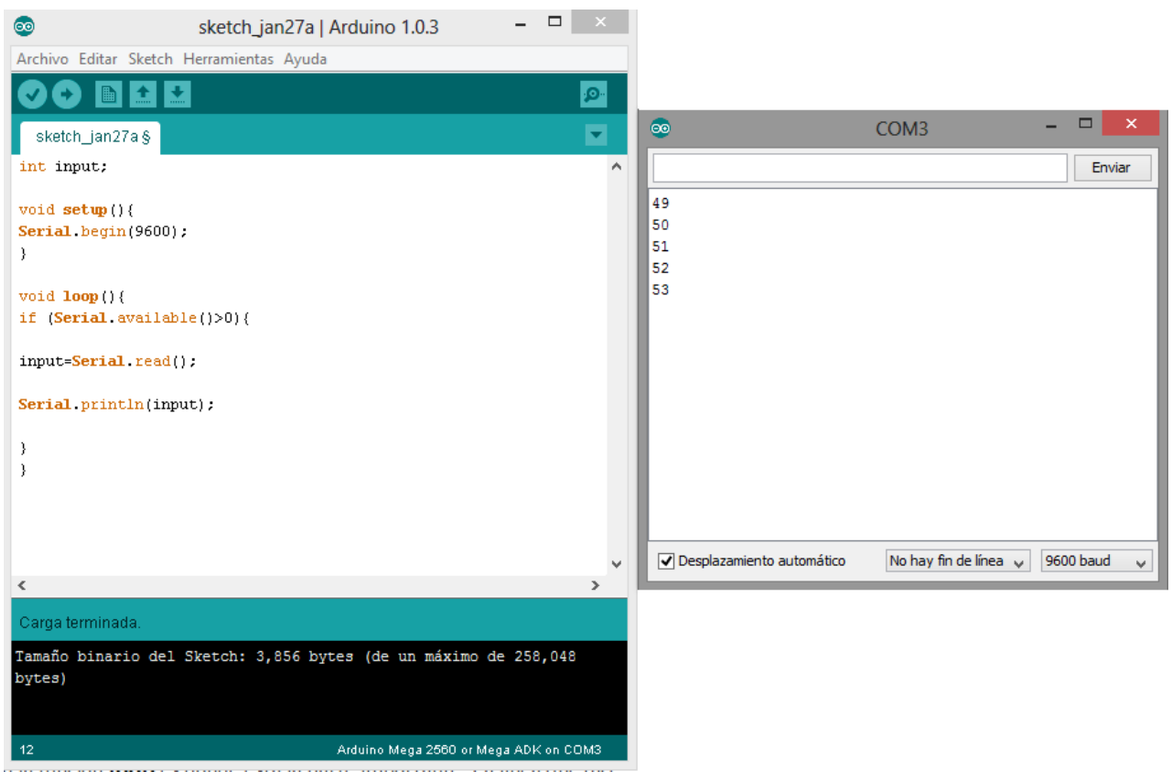
50

51

52

53

Lo podemos apreciar en esta imagen:



Se preguntarán, si he tecleado el 1 y me aparece el 49, con el 2 me aparece el 50, con el 3 el 51 y así sucesivamente… ¿que está sucediendo?

Simplemente es la forma como estamos imprimiendo los caracteres con el método **Serial.println().**

Cuando imprimíamos el '1' en el ejemplo anterior, aparecía el 1 en el monitor serial. Ahora estamos imprimiendo un valor entero a través de la variable **input**.

Cuando se intenta imprimir un entero (1,2,3,4,5…), el método **Serial.println()** imprimirá dicho entero en formato ASCII.

El ASCII es muy utilizado en programación y telecomunicaciones. Existen tablas que permiten traducir caracteres de ASCII a valores alfanuméricos y viceversa.

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido (Página de código 437)

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

En esta tabla se muestra los valores ASCII que representa cada caracter del alfabeto. A los números del 0 al 9 les corresponde los valores del 48 (0) al 57 (9).

Si se desea que los números que aparezcan en el monitor serial aparezcan tal y como lo introducimos por teclado, podemos agregar la línea

input=input-48;

Esta sentencia le quitará 48 a cualquier número ASCII, lo que hará que nos aparezcan números del 0 al 9.

Es cuestión de cada persona empezar a teclear valores en el monitor serial y ver cual es el resultado.