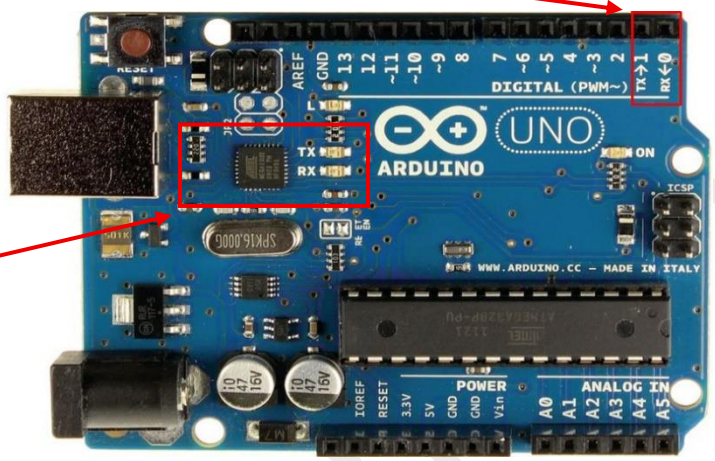
ARDUINO – Monitor Seriale

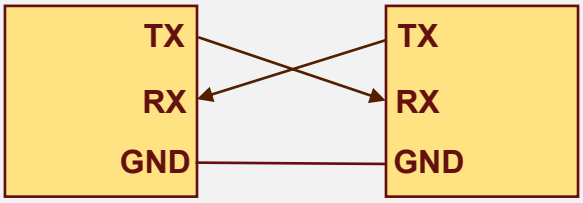
Tra i vari GPIO della scheda Arduino UNO, ce ne sono due che vengono utilizzati per connettere la scheda al PC tramite il cavo USB.

I due pin 0 e 1 sono anche rinonimanti con RX e TX, in quanto comunicano con il PC utilizzando una comunicazione seriale RS232 TTL convertita poi in segnali compatibili con lo standard USB tramite gli appositi componenti presenti sulla scheda.



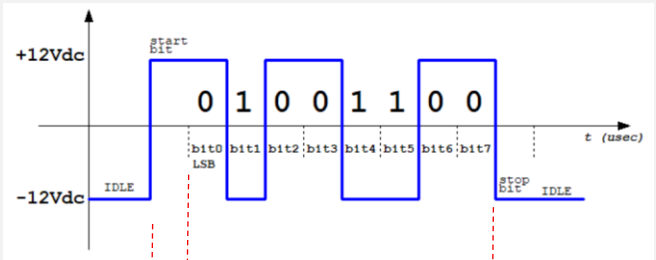
Standard RS232 – Approfondimento

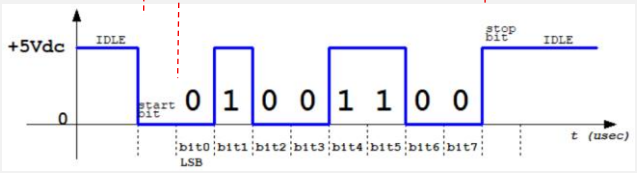
*Lo standard **RS232** (Recommended Standard 232) è un protocollo di comunicazione seriale utilizzato per la trasmissione di dati tra dispositivi elettronici tramite collegamento **punto-punto**.*
*Vengono utilizzati due canali **TX e RX con segnali riferiti alla GND** comune tra i due dispositivi.*



*Lo standard prevede l'invio e la ricezione di bit in modalità seriale **Full-Duplex** in quanto i due dispositivi possono inviare e ricevere dati allo stesso tempo.*

Nella comunicazione seriale RS232, viene inviato un byte alla volta nella seguente maniera.
*La linea normalmente si trova nello stato di **IDLE** corrispondente a -12Volt.*
La trasmissione inizia con il bit di start che corrisponde a +12Volt.
*Successivamente vengono inviati gli 8 bit partendo dal meno significativo mettendo sulla linea una tensione di **+12Volt** per inviare un bit a **ZERO** o **-12Volt** per inviare un bit ad **UNO**.*
Al termine viene inviato uno o più stop bit con la tensione di -12Volt.
*La durata di ogni bit dipende dalla velocità di trasmissione espressa in **bit per secondo**.*



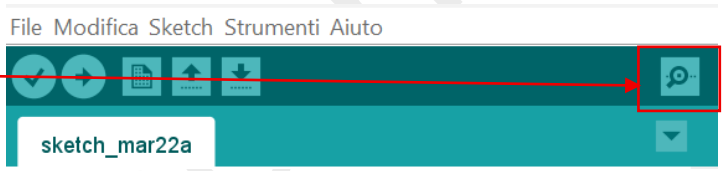


Nel mondo dei microcontrollori lo standard RS232 viene applicato con livelli di tensione differenti 0-5Volt e viene definito RS232-TTL.
Il funzionamento ò lo stesso con la differenza che il livello -12Volt corrisponde a +5Volt e +12Volt corrisponde a 0Volt.

Il microcontrollore presente sulla scheda Arduino UNO dispone di una periferica hardware UART (un circuito interno al microcontrollore) che gestisce automaticamente la comunicazione seriale. I pin utilizzati sono RX e TX, corrispondenti rispettivamente al pin digitale 0 e al pin digitale 1.
I segnali provenienti da questi pin non vengono convertiti direttamente dal microcontrollore principale, ma da un secondo microcontrollore presente sulla scheda, che funge da convertitore USB-seriale, permettendo lo scambio di dati con il PC tramite la porta USB.
Durante la programmazione della scheda, questi stessi pin vengono utilizzati per ricevere il codice inviato dal computer da scrivere nella memoria flash interna del microcontrollore.
Gli stessi pin possono inoltre essere utilizzati durante l'esecuzione del programma per inviare e ricevere dati seriali tramite il **Monitor Seriale** dell'ambiente di sviluppo (IDE).

MONITOR SERIALE

Si può accedere al monitor seriale con il pulsante in alto a destra dell'IDE di Arduino.



Prima di utilizzare il monitor seriale, bisogna però attivare la comunicazione seriale inserendo nel setup il comando Serial.begin:

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

Tra le parentesi va indicata la velocità espressa in bit per secondo, in questo caso 9600 bit per secondo, che sta a significare che ogni bit avrà una durata di 1/9600 secondi, cioè circa 104usec.

Un volta collegata la scheda ed impostata la porta di comunicazione su STRUMENTI – PORTA aprendo il monitor seriale avremo questa schermata, la prima cosa da fare è impostare la stessa velocità indicata nel Setup.

Successivamente scegliere "**Nessun fine riga**".

In questo modo quando invieremo i dati dal PC alla scheda, non invieremo caratteri aggiuntivi di formattazione. Vedremo successivamente questo aspetto.

danielepostacchini.it

Da questo momento in poi sarà possibile inviare e ricevere dati con le seguenti funzioni:

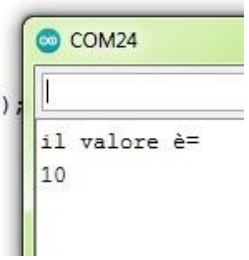
print println write read available

Le funzioni vanno chiamate precedendo il loro nome con la parola **Serial.** d'ora in avanti non le chiameremo **funzioni** ma **metodi**.

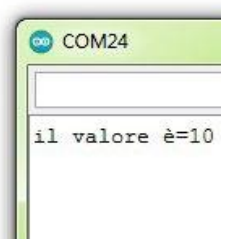
Metodi print e println

I due metodi servono per scrivere sul Monitor seriale un valore di una variabile o una stringa di caratteri.
La differenza è che println dopo aver scritto invia due comandi che indicano di eseguire due operazioni, il "carriage return" ed il "line feed", cioè vai accapo e crea una nuova linea.
Mentre invece il metodo print non invia questi due caratteri, perciò i valori o le stringhe appariranno scritte in maniera consecutiva.

```
int valore=10;

void setup() {
  Serial.begin(9600);
  Serial.println("il valore è=");
  Serial.print(valore);
}//setup
```



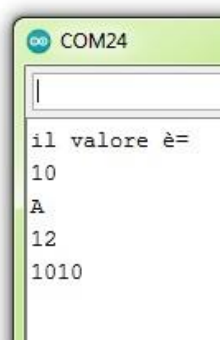

```
int valore=10;

void setup() {
  Serial.begin(9600);
  Serial.print("il valore è=");
  Serial.print(valore);
}//setup
```



In entrambi i casi se vogliamo visualizzare il valore di una variabile, il metodo va richiamato con la seguente sintassi: ***Serial.println(val, format)***

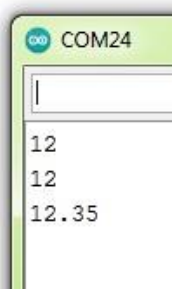
Il primo valore è la variabile da visualizzare, il secondo è il formato che può essere DEC (decimale) HEX (esadecimale) OCT (ottale) BIN (binario).

```
int valore=10;

void setup() {
  Serial.begin(9600);
  Serial.println("il valore è=");
  Serial.println(valore,DEC);
  Serial.println(valore,HEX);
  Serial.println(valore,OCT);
  Serial.println(valore,BIN);
}//setup
```



Si può visualizzare anche direttamente un valore numerico indicando il numero di decimali, dopo il suo valore, nell'esempio 12,348 viene visualizzato con 0 decimali e con 2 decimali.

```
int valore=10;

void setup() {
  Serial.begin(9600);
  Serial.println(12);
  Serial.println(12.348,0);
  Serial.println(12.348,2);
}//setup
```

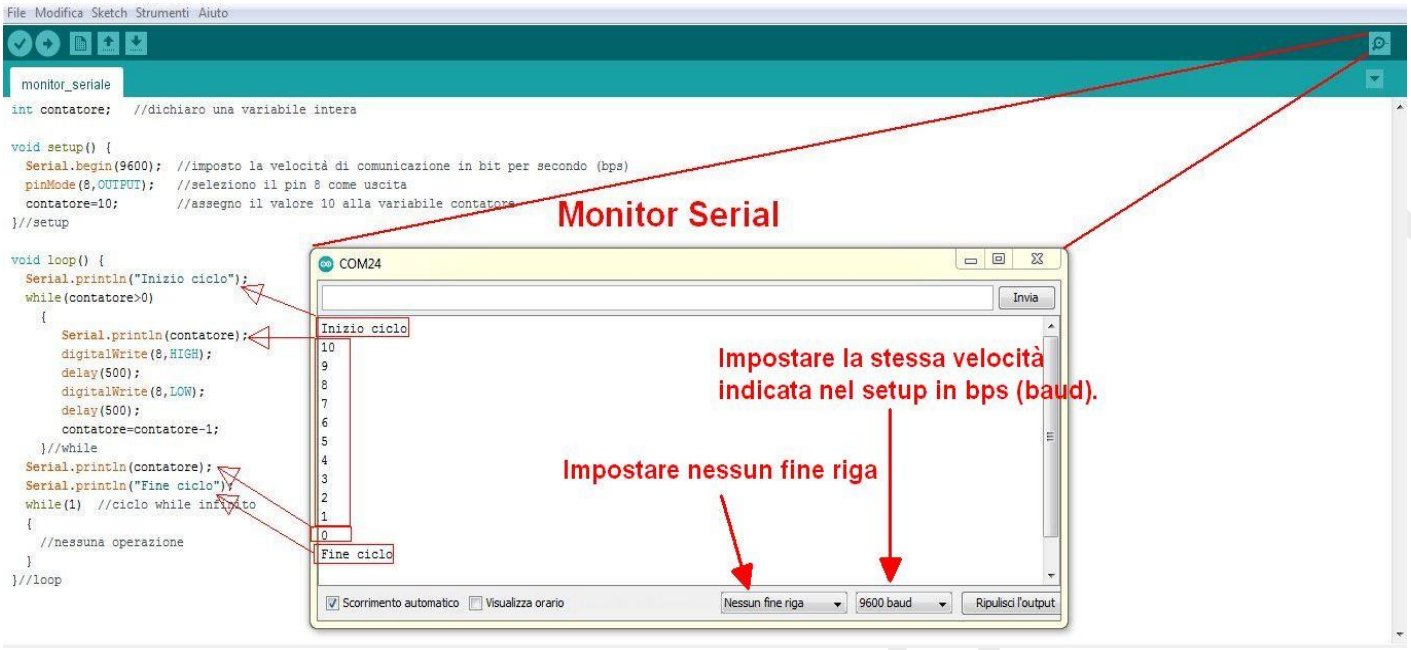


Se vogliamo invece visualizzare una scritta non occorre indicare il formato, ma solo la stringa di caratteri da visualizzare racchiusa tra le virgolette: `Serial.print("Valore");`
Nel caso volessimo invece visualizzare un singolo carattere la sintassi sarà: `Serial.print('A');`

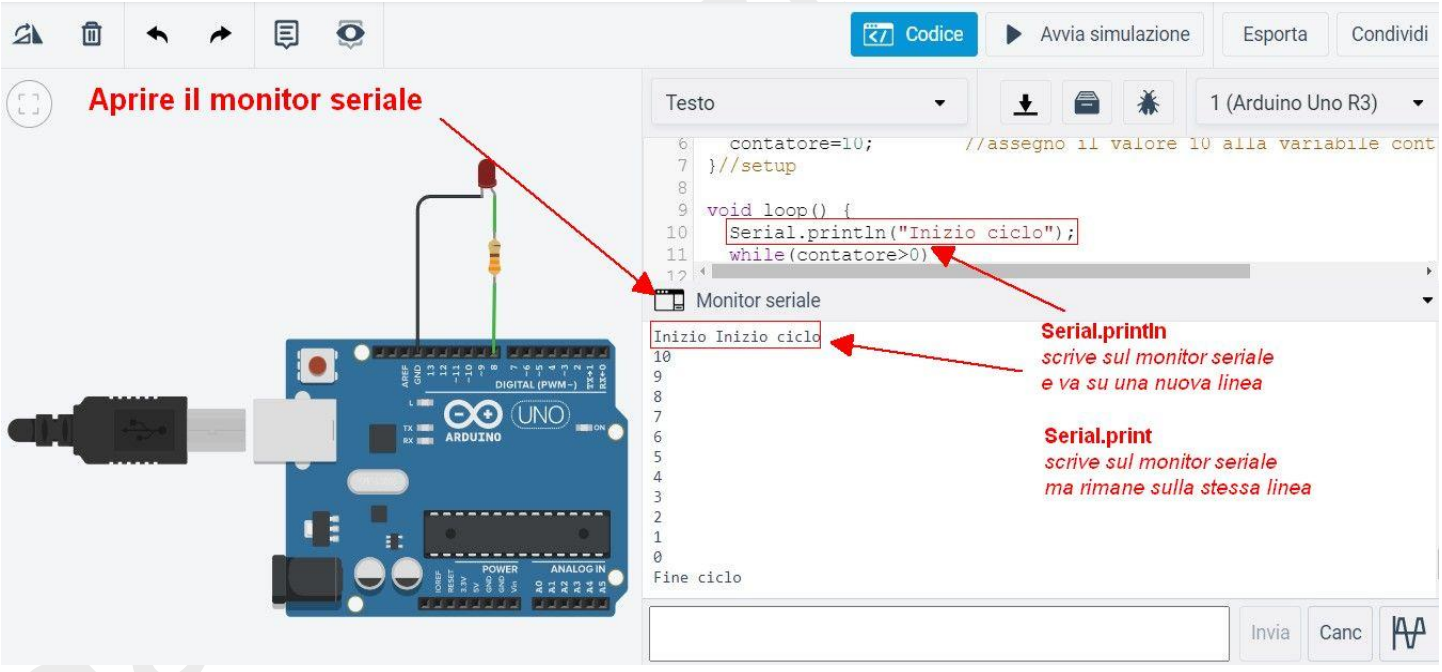
danielepostacchini.it

Proviamo ora a mettere in pratica questi primi due metodi e cioè **begin**, **print** e **println**.

Facciamo un semplice programma per visualizzare un conteggio all'indietro che parte da 10 fino ad arrivare a zero, facendo lampeggiare un led.



Oppure sulla piattaforma Tinkercad con lo stesso programma.



In entrambi i casi, il monitor seriale ci consente di visualizzare in tempo reale il valore della variabile di conteggio.

Codice ASCII – Approfondimento

Il codice ASCII (acronimo di American Standard Code for Information Interchange) è il linguaggio universale che permette ai computer di tradurre i numeri in lettere e simboli che noi possiamo leggere e viceversa.

Ad ogni lettera dell'alfabeto, cifra numerica o simbolo, viene associato un numero che va da 0 a 127, si utilizzano 7 bit.

Oltre alle lettere, numeri e simboli ci sono anche dei comandi che vengono tradotti in numero, come ad esempio LF (line Feed) nuova linea proseguendo dalla posizione del cursore, o CR (Carriage Return) nuova linea con il cursore all'inizio della riga. *Ad esempio la cifra 1 corrisponde al numero 49 in decimale e 31 in esadecimale.*

- *Caratteri di controllo.*
- *Simboli, lettere e cifre.*

NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127

(tabella ASCII estratta dall'IDE MikroC della Mikroelektronika)

Codice ASCII EXTENDED

In questo caso non si utilizzano 7 bit, ma 8 bit, pertanto i numeri andranno da 0 a 255 rappresentando più caratteri.

*Oggi si utilizza anche l'**Unicode (UTF-8):** Unicode può mappare oltre 140.000 caratteri, includendo emoji, geroglifici e alfabeti asiatici.*

I primi 128 caratteri di Unicode sono identici all'ASCII originale, garantendo la retrocompatibilità.

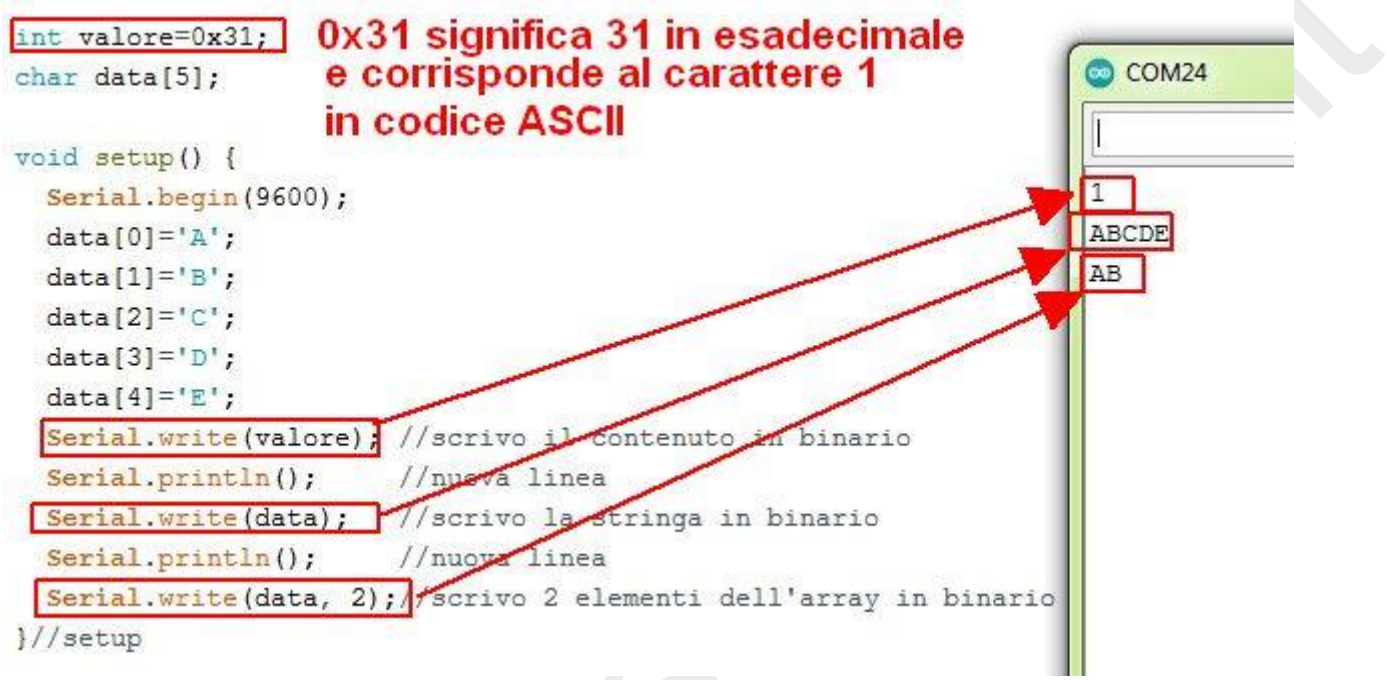
NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
€		‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ		Ž	
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
	‘	’	“	”	•	–	—	˜	™	š	›	œ		ž	Ÿ
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Metodo write

Con questo metodo si possono inviare dei dati in formato grezzo (cioè il byte effettivo) sulla seriale e sul monitor seriale.

Si può inviare il singolo byte o un insieme di byte sottoforma di stringa o array.

La sintassi è la seguente: ***Serial.write(stringa o numero, lunghezza)*** dove con lunghezza identifico quanti byte inviare della stringa selezionata.



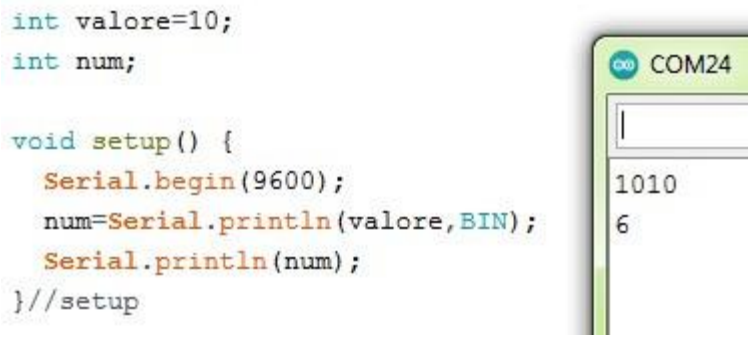
Alla variabile valore è stato associato inizialmente il valore 0x31 che significa 31 in esadecimale cioè 49 in decimale.
Nell'array data invece è stato scritto il carattere mettendolo tra apice, è come se avessimo scritto il numero ASCII corrispondente, stam

---

Sia il metodo print che il metodo write, restituiscono il valore dei byte inviati.

Ciò significa che posso assegnare ad una variabile il valore che il metodo chiamato mi restituirà come risultato.

Nell'esempio a fianco con il valore restituito da **Serial.println**, va in una variabile **num**, che viene poi visualizzata e che vale 6, perché oltre ai 4 byte 1010, sono stati inviati i caratteri **CR** e **LF** e cioè **carriage return** e **line feed** (nuova linea accapo).



Utilizzando invece il metodo **print**, la funzione avrebbe restituito il valore 4.

Lo stesso discorso per il metodo **write**.

danielepos

Metodi - available e read

Se invece vogliamo leggere dei dati dalla seriale o dal Monitor, dobbiamo prima vedere se ci sono dati disponibili, e ciò va fatto col metodo **available**, che va spesso affiancato all'istruzione condizionale **if**.
Se c'è un dato sulla Seriale o sul Monitor seriale, questo poi viene letto con il metodo **read**.

Il dato scritto, sulla linea di comando del Monitor Seriale, viene letto e messo nella variabile lettura che poi verrà visualizzata.

```
char lettura;

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

void loop() {
  if(Serial.available()){
    lettura=Serial.read();
    Serial.println(lettura);
  }//if
}//loop
```

Grazie a queste funzioni, che in realtà sono dei "metodi dell'oggetto Serial", possiamo controllare l'andamento del nostro programma, inserendo delle scritture sul monitor seriale, per verificare il contenuto di determinate variabili, o per controllare il flusso del programma.
Come già detto il Monitor seriale è un ottimo strumento per la ricerca degli errori ed il debug del programma, e le stesse istruzioni si possono utilizzare per inviare dei dati sulla seriale RX,TX su terminali 0 ed 1 di Arduino.
I due piedini ovviamente non si potranno utilizzare come GPIO generici se utilizziamo il monitor seriale o la seriale asincrona RS232.

danielepostac