Sense HAT su Raspberry Pi
Utilizzo con Python – Speciale AstroPi

Versione 0.2
5/3/2018

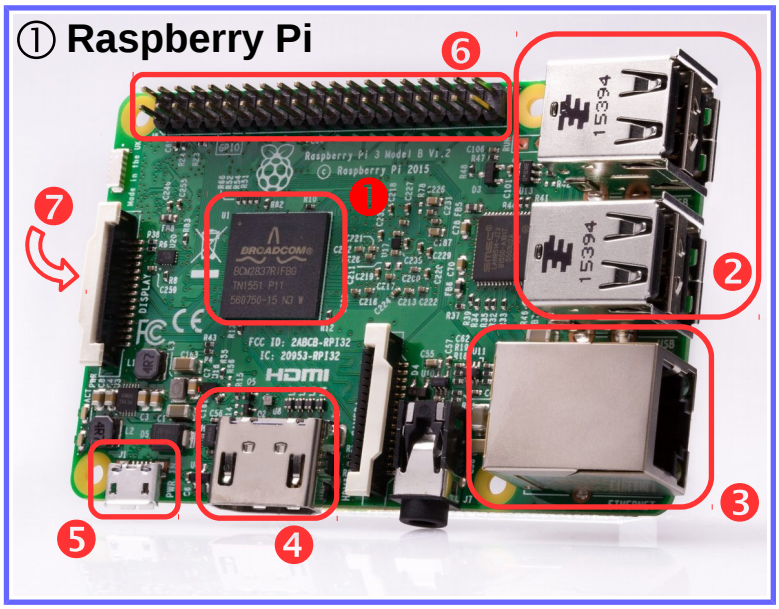
In questo tutorial CoderDojo Trento va in orbita! Prendendo spunto dalla sfida non competitiva lanciata da **AstroPi**, presentiamo l'ambiente, il linguaggio di programmazione e le schede elettroniche utilizzate per completare il progetto

Se ti mancano la scheda Raspberry Pi o il SenseHat, non disperare: non sono necessari ! Basta collegarti al sito **trinket.io/sense-hat** per trovare l'ambiente di programmazione per Python e l'emulatore del Sense HAT, il tutto senza dover installare nulla !! Più facile di così .... :-)

Trovi questo tutorial qua: coderdojotrento.it/astropi1

L'ambiente: Raspberry Pi

Sta comodamente sul palmo della mano, è una scheda elettronica che ricorda molto Arduino che già conosciamo, ma è molto molto di più: **è infatti a tutti gli effetti un computer**!



Se guardiamo il ***Raspberry Pi***, al centro, subito sotto l'inconfondibile marchio del lampone riconosciamo l'integrato che contiene **CPU e RAM** (①); a destra ci sono le quattro **porte USB** alle quali collegheremo periferiche come tastiera e mouse (②) e la **porta Ethernet** per connettere il nostro computer ad una rete (③).

In basso vediamo la **porta HDMI** per il monitor (④); il connettore Micro USB (⑤) serve per **alimentare** la scheda.

In alto ci sono poi due file di pin (in ⑥, sono 40 in tutto) che ricordano quelli di Arduino, ed infatti servono proprio a questo: a queste porte che si chiamano **GPIO (general purpose input/output)** possiamo collegare componenti esterni, come LED o interruttori ed interagire con questi direttamente dal nostro piccolo computer. Forte, no?

Non manca nulla! Quasi. C'è praticamente tutto, a parte la **memoria di massa**: è impossibile trovare un hard disk che si adatti a queste dimensioni, così al suo posto si utilizza una schedina Micro SD (le stesse che si usano in cellulari o macchine fotografiche digitali; la fessura per la

Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported

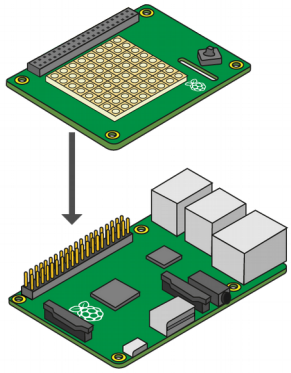
schedina nella foto non si vede, ma sta in ❼, sotto il connettore per il display esterno).

Quest'ultima cosa sembra un difetto, invece è comodissima: sulla memoria di massa mettiamo infatti il **sistema operativo** e tutti i **programmi** che ci interessa far girare sul nostro computer, quindi semplicemente cambiando la schedina potremmo avere due computer completamente diversi (ad esempio uno per la scuola ed un altro dedicato interamente ai giochi)!

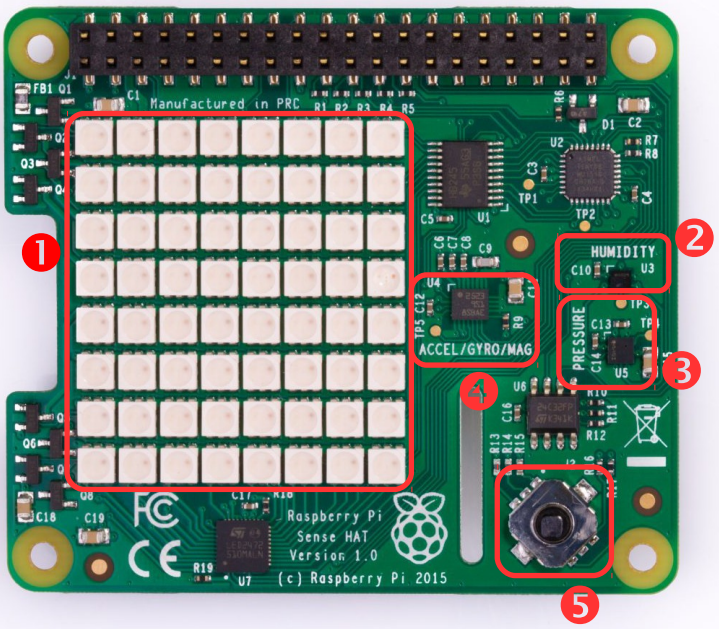
La scheda con i sensori: Sense HAT

Il team di AstroPi (supportato dall' **ESA**, l'agenzia spaziale europea) suggerisce di fare i nostri esperimenti appoggiandosi alla scheda elettronica ***Sense HAT***: molto simile agli *shield* di Arduino, come tutti gli *HATs* (sigla che sta per "*Hardware Attached on Top*", ma che non a caso in inglese significa pure "*cappello*") si applica sopra il Raspberry Pi e contiene diversi sensori che possono essere utilizzati per monitorare l'ambiente circostante; è inoltre dotata di un piccolo joystick e di una matrice di LED colorati sulla quale possiamo far comparire delle scritte scorrevoli oppure dei disegni.

A HATs (cappelli)



B Componenti del Sense HAT



Nella figura **B** vediamo la scheda *Sense HAT* ed il dettaglio di tutti i suoi componenti: la **matrice quadrata di LED** salta subito all'occhio (①); sulla destra troviamo i componenti integrati per la misurazione di **umidità e temperatura** (②), di **pressione** (③) e l'integrato che fa da **accelerometro, giroscopio e magnetometro** (④); quest'ultimo incredibile componente riesce quindi a misurare le variazioni di velocità, l'orientazione della scheda e l'intensità e la direzione del campo magnetico terrestre, come farebbe una bussola. In basso vediamo anche il piccolo **joystick** (⑤).

Il linguaggio di programmazione: Python

Dei diversi modi che esistono per interagire con il *Sense HAT*, AstroPi suggerisce di utilizzare il *linguaggio di programmazione* **Python**: in questo tutorial rinunceremo quindi ai blocchetti di Scratch che ci permettono di costruire degli script complessi mettendo in cascata ed incastrando graficamente dei mattoncini.

Al loro posto, useremo delle ***parole chiave*** (o "*key words*") che appunto costituiscono questo linguaggio che viene compreso dal nostro computer: sono una manciata di parole in inglese che non faremo fatica ad imparare e che corrispondono a delle istruzioni o comandi che verranno eseguiti quando il nostro programma girerà.

Metteremo questi comandi in un file che poi l'**interprete Python** leggerà e farà girare, e tutto

Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported

funzionerà come vogliamo a patto che il nostro programma rispetti la ***sintassi***: come quando scriviamo una lettera ad un amico straniero, non possiamo accontentarci di mettere una lista di vocaboli. Solo se formiamo delle frasi corrette, infatti, possiamo sperare che il messaggio venga compreso e -se il nostro amico di penna sarà gentile ed obbediente come il nostro *Raspberry Pi*- anche eseguito.

Cosa ci serve?

L'ideale è disporre di:

- un *Raspberry Pi* con installato l'ambiente di programmazione "Python 3 IDLE";
- una scheda *Sense HAT* applicata al *Raspberry Pi*;
- tutte le periferiche (tastiera, mouse, monitor) per usare il *Raspberry Pi* come un PC.

Se ci mancano le periferiche (oppure sono occupate da qualcun altro che ha preso il posto prima di noi), possiamo utilizzare il nostro PC e collegarci al *Raspberry Pi* utilizzando il programma VNC (scaricabile dal fornitore **Real VNC**): grazie a questo programma, potremo avere sul nostro PC una scrivania esattamente come se fossimo collegati direttamente al *Raspberry Pi.* In questo caso la lista della spesa sarà:

- un *Raspberry Pi* con installato l'ambiente di programmazione "Python 3 IDLE";
- una scheda *Sense HAT* applicata al *Raspberry Pi*;
- il nostro PC con installato VNC per connettersi al *Raspberry Pi*.

Se ci manca la scheda Sense HAT, peccato, ma possiamo fare lo stesso i nostri esperimenti: al suo posto possiamo usare un **emulatore** che normalmente è installato sul *Raspberry Pi*: l'emulatore è un programma che copia il comportamento della scheda hardware e ci permette di fingere di avere comunque tutti i componenti necessari. Quando lanciamo l'emulatore (lo troviamo tra le applicazioni sotto il menù "Programmazione") compare una finestra con uno disegno del *Sense HAT* e vari controlli che ci permettono di simulare ad esempio i parametri di temperatura ed umidità che verranno letti dal nostro programma, ignaro e convinto di dialogare con una vera scheda *Sense HAT*. Se col nostro programma -poi- accenderemo qualche LED della matrice, il disegno mostrato dall'emulatore cambierà di conseguenza per mostrarci i (finti) LET accesi.

La lista del materiale necessario sarà:

- un *Raspberry Pi* con installato l'ambiente di programmazione "Python 3 IDLE";
- un emulatore della scheda *Sense HAT*;
- il nostro PC con installato VNC per connettersi al *Raspberry Pi*.

E se ci manca anche il *Raspberry Pi*? Non disperiamo: possiamo utilizzare il nostro PC per collegarci al sito **trinket.io/sense-hat** e lì troveremo l'ambiente di programmazione per *Python* e l'emulatore del *Sense HAT*. In quest'ultimo caso, quindi, la lista delle cose che ci servono è:

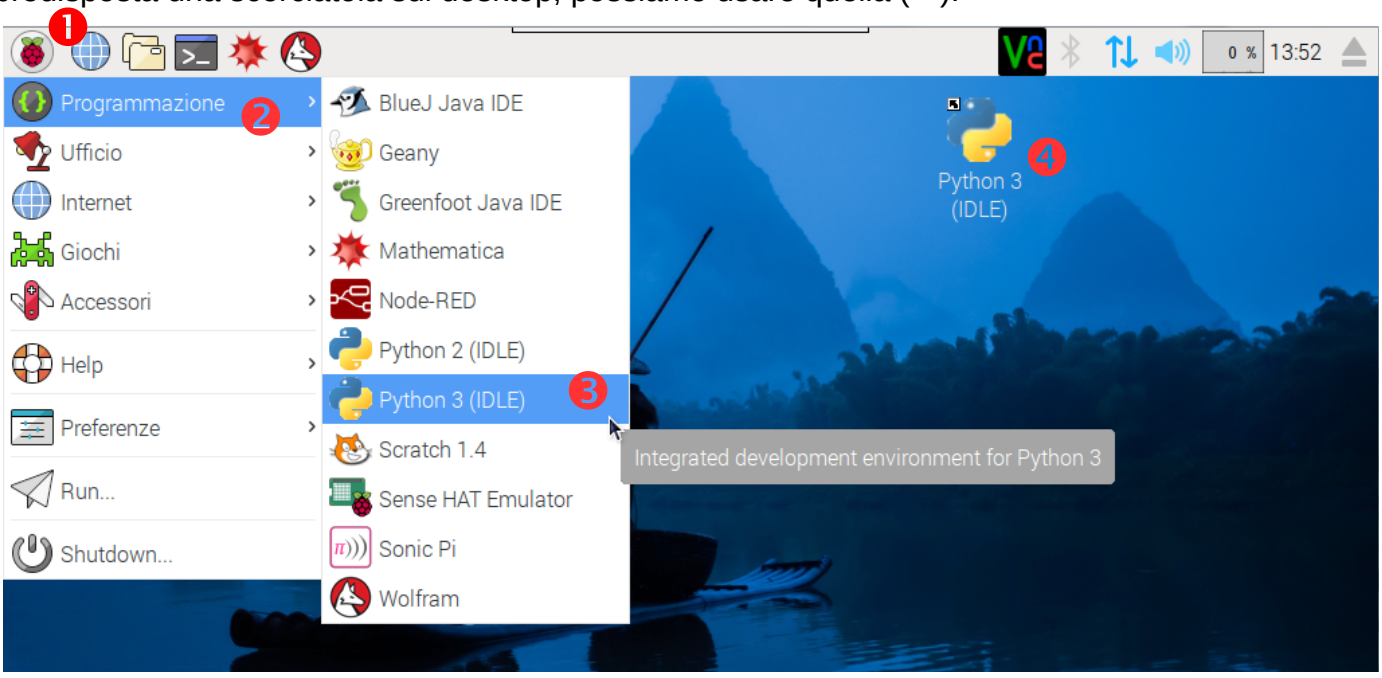
- il nostro PC con un browser qualunque (meglio se Google Chrome*)*;
- collegarsi al sito **trinket.io/sense-hat**

Trovi questo ed altri tutorial su coderdojotrento.it
Realizzato con il supporto di CoderDolomiti.
Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported

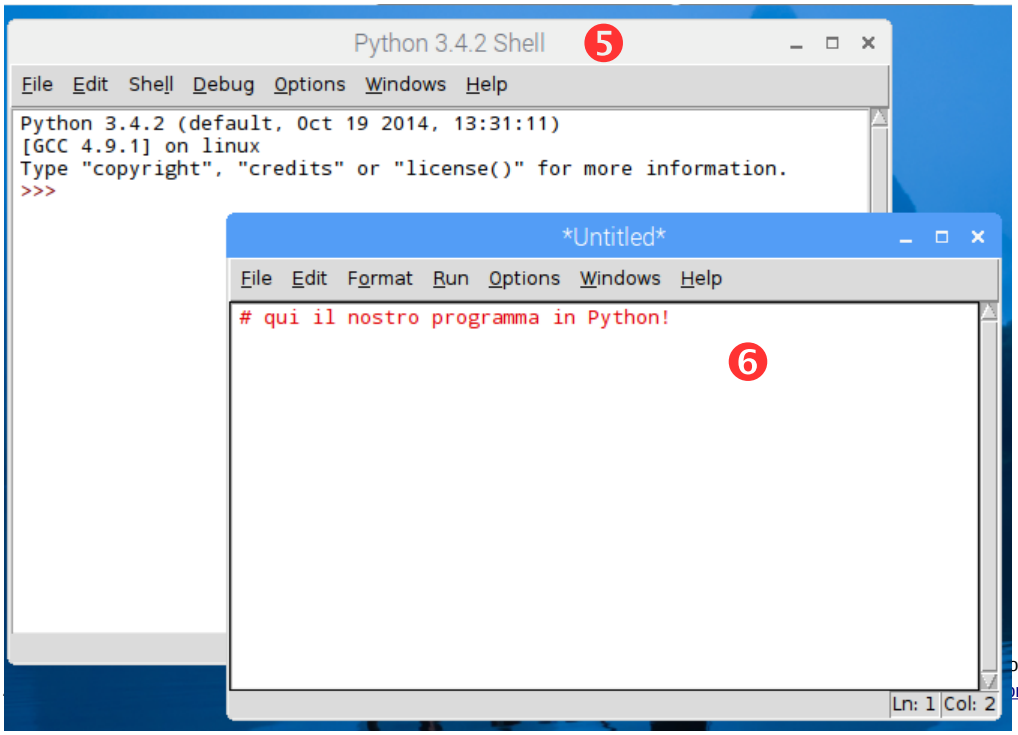
Stavolta la lista è molto corta, ed è un bene: significa che possiamo fare esperimenti con il *Sense HAT* in ogni momento, anche senza disporre dell'hardware che normalmente darebbe richiesto.

L'IDLE per Python

Supponendo di avere un Raspberry Pi, vediamo come far partire l'ambiente per scrivere e lanciare i nostri programmi in Python: dal menù principale (quello col lampone: ①), scegliamo il sottomenù "Programmazione" (②) e quindi scegliamo "Python 3 (IDLE)" (③). In alternativa, se è già stata predisposta una scorciatoia sul desktop, possiamo usare quella (④).



Una volta lanciato il programma, compare la finestra di "shell" (⑤) e qui possiamo sottomettere singoli comandi (prova ad esempio a scrivere `print("Ciao!")` ed a premere il tasto Invio), ma per scrivere dei veri e proprio programmi costituiti da più istruzioni dobbiamo creare un nuovo file per contenerli, usando il menù "File → New File".

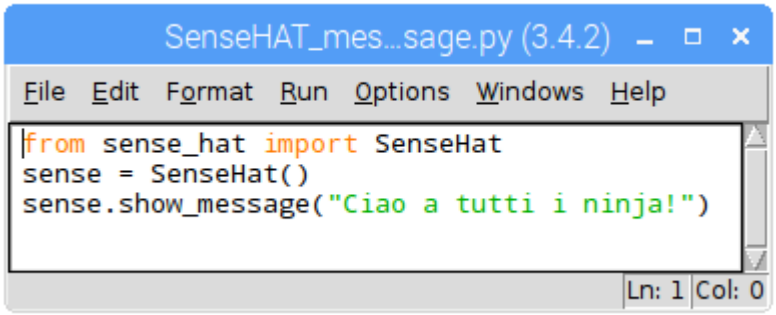


Nella nuova finestra (⑥) metteremo quindi le istruzioni del linguaggio Python che costituiscono il nostro programma (come tutti gli esempi che troveremo nelle pagine successive) e potremo salvarlo sulla memoria del *Raspberry Pi* (menù "File → Save as..."); ricorda che di solito gli script in Python hanno estensione `.py`

o Ebranati, CoderDojo Trento
ons Attribuzione 4.0 Unported

Per lanciare il programma, usremo il menù “Run → Run module” o equivalentemente il tasto F5.

La matrice di LED: messaggi



Possiamo far comparire delle scritte scorrevoli sulla **matrice di LED** usando le istruzioni del programma qui a destra.

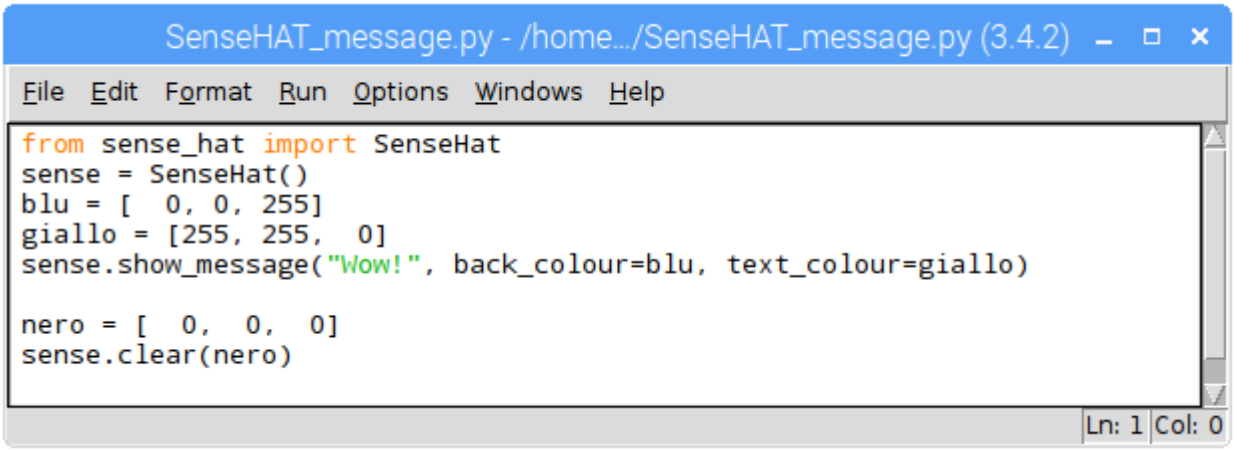
Le prime due righe ci accompagneranno per tutto il tutorial: nella prima importiamo SenseHat (una collezione di funzionalità) dalla libreria che si chiama sense_hat: come puoi notare, le due parole chiave “from” e “import” sono evidenziate (l’ambiente di sviluppo le ha riconosciute come parole note); nella seconda riga creiamo un oggetto dal nome **sense** che avrà tutte le funzionalità offerte dalla classe SenseHat.

Da qui in avanti, **sense** sarà il nostro “interlocutore” per sottomettere dei comandi indirizzati alla scheda Sense HAT: è infatti a lui che chiediamo, nella terza riga, di mostrare il messaggio “Ciao a tutti i ninja!”. Salva il file e lancia il programma con F5 e vediamo che succede!

Puoi cambiare il comportamento del programma variando ad esempio la stringa da mostrare che viene passata -tra parentesi rotonde- come argomento al metodo `show_message`. Oppure possiamo cambiare la velocità di scorrimento del testo, cambiando la terza riga ed aggiungendo un parametro numerico (normalmente -quando non è specificato- questo parametro vale 0.1):

```
sense.show_message("Ciao a tutti i ninja!", 0.04)
```

Possiamo anche variare il colore della scritta: possiamo “costruirci” i colori che desideriamo mescolando della luce rossa, verde e blu in quantità variabili tra 0 e 255 (usiamo la notazione **RGB: red / green / blue**); il blu puro, sarà quindi [0, 0, 255] mentre per il giallo utilizzeremo una uguale quantità di luce rossa e verde. Il nero (che usiamo nel metodo `clear` che ripulisce lo schermo) sarà giustamente privo di ogni componente luminosa: [0, 0, 0].



Puoi verificare la composizione RGB del colore che preferisci andando a visitare questa pagina: https://www.w3schools.com/colors/colors_rgb.asp

Un altro comando da provare è quello per mostrare un singolo carattere:

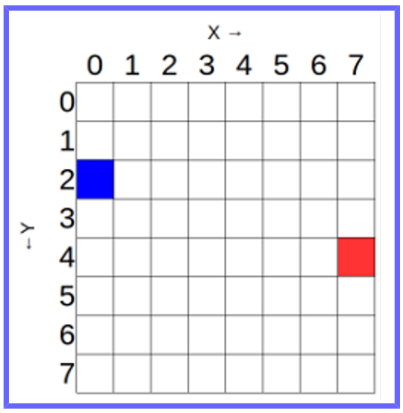
```
sense.show_letter("?", text_colour=blu, back_colour=nero)
```


Trovi questo ed altri tutorial su coderdojotrento.it
Realizzato con il supporto di CoderDolomiti.
Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported

La matrice di LED: pixel e pixels

Possiamo anche comandare l'accensione di singoli LED (che per noi saranno come pixel di un piccolo monitor 8x8) utilizzando le loro **coordinate X e Y** e specificando un colore.

Con il programma qui sotto, possiamo ottenere il disegno a destra.



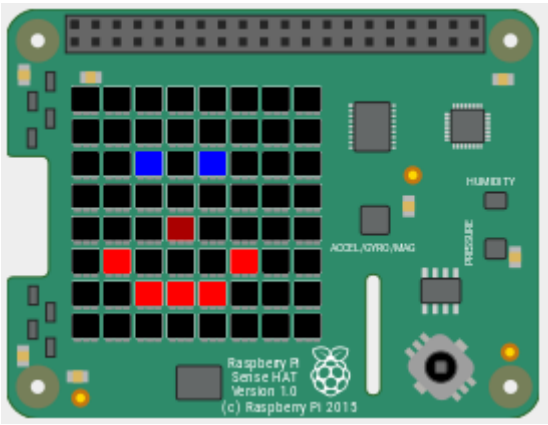

```
from sense_hat import SenseHat
sense = SenseHat()
sense.set_pixel(0, 2, [0, 0, 255])
sense.set_pixel(7, 4, [255, 0, 0])
```

Con un po' di pazienza, puoi creare dei piccoli disegni: qui sotto c'è un programma che usa otto volte la chiamata a `set_pixel` (a destra viene mostrato il risultato sull'emulatore).

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

sense.set_pixel(2, 2, (0, 0, 255))
sense.set_pixel(4, 2, (0, 0, 255))
sense.set_pixel(3, 4, (100, 0, 0))
sense.set_pixel(1, 5, (255, 0, 0))
sense.set_pixel(2, 6, (255, 0, 0))
sense.set_pixel(3, 6, (255, 0, 0))
sense.set_pixel(4, 6, (255, 0, 0))
sense.set_pixel(5, 5, (255, 0, 0))
```



Se comporre un disegno richiamando più volte la funzione `set_pixel` è complicato (ed anche un po' noioso), possiamo disegnare tutti i 64 pixel in un colpo solo usando la chiamata `set_pixels` (alla quale passeremo non uno, ma l'intera lista dei pixel da disegnare).

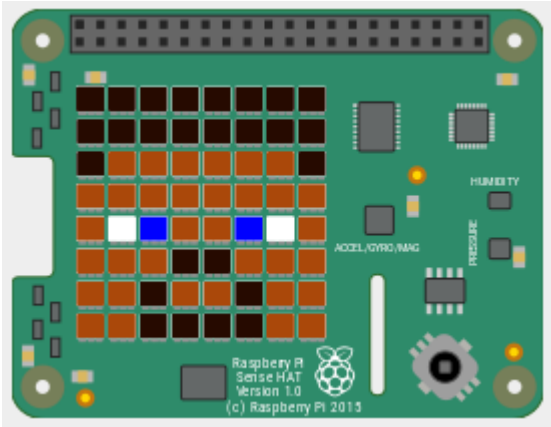
```
from sense_hat import SenseHat
sense = SenseHat()

# Definiamo un po' di colori
M = [102,  51,   0]  # marrone
A = [  0,   0, 255]  # azzurro
R = [205, 133,  63]  # rosa scuro
B = [255, 255, 255]  # bianco

# Decidiamo dove piazzare i colori
steve_pixels = [
    M, M, M, M, M, M, M, M,
    M, M, M, M, M, M, M, M,
    M, R, R, R, R, R, R, M,
    R, R, R, R, R, R, R, R,
    R, B, A, R, R, A, B, R,
    R, R, R, M, M, R, R, R,
    R, R, M, R, R, M, R, R,
    R, R, M, M, M, M, R, R
]

# Mostriamo i colori sulla matrice di LED
sense.set_pixels(steve_pixels)
```

Se ci aiutiamo con delle variabili per memorizzare i colori, riusciremo a comporre velocemente anche dei disegni complessi.



Trovi questo ed altri tutorial su coderdojotrento.it
Realizzato con il supporto di CoderDolomiti.
Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported

Ancora più comodo, è disegnare direttamente una piccola immagine 8x8 utilizzando il nostro editor di immagini preferito e salvare tutto in un file .PNG; in alternativa possiamo lanciare il programma Rpi_8x8GridDraw (scritto anch'esso in Python) che ci permette di disegnare quello che vogliamo sulla griglia 8x8 ed esportare poi il risultato in un file .PNG oppure in un pezzetto di codice che potrà essere utilizzato per pilotare il *Sense HAT*.

Se abbiamo un file di nome `immagine8x8.png`, potremo caricarlo sulla matrice di LED usando il seguente comando:

```
sense.load_image("immagine8x8.png")
```

Si possono variare gli effetti di tutte le istruzioni relative alla matrice LED che abbiamo visto sopra, cambiando l'orientazione della matrice o trasformando l'immagine rendendola speculare tra destra e sinistra oppure scambiando alto e basso.

Questi i comandi:

```
sense.set_rotation(90)   # i valori ammessi sono 0, 90, 180 e 270
sense.flip_h()           # immagine speculare destra/sinistra
sense.flip_v()           # immagine speculare alto/basso
```

Sensori di temperatura, pressione e umidità

La consultazione dei sensori che forniscono la temperatura ambiente, la pressione e l'umidità è molto semplice: otterremo i valori misurati grazie alle seguenti chiamate:

```
sense.get_temperature()
sense.get_pressure()
sense.get_humidity()
```

```
from sense_hat import SenseHat
sense = SenseHat()

while True:
    t = round(sense.get_temperature(),1)
    u = round(sense.get_humidity(),1)
    p = round(sense.get_pressure(),1)


    msg = "Temp. = %s, Umid. = %s, Press. = %s" % (t, u, p)

    print (msg)
    sense.show_message(msg, scroll_speed=0.05)
```

Nel programma qui a destra vengono chieste al *Sense HAT* le tre misure, arrotondate ad una cifra decimale ed assegnate alle variabili t, u e p; queste vengono usate per preparare il messaggio (contenuto nella variabile "msg") che viene stampato sia sulla shell di Python, sia sulla matrice di LED. Queste operazioni sono ripetute per sempre, dato che sono state inserite in un ciclo infinito (l'istruzione `while` che ha una condizione di ripetizione che è costantemente vera).

Puoi interrompere forzatamente un programma che gira all'infinito mandando alla shell un segnale di interruzione: con il Raspberry Pi ed in altri ambienti simili questo segnale si ottiene tenendo premuto il tasto Ctrl e premendo quindi la lettera C.

Ringraziamenti Questo tutorial:

* fa parte delle "risorse" del CoderDojo Trento: coderdojotrento.it/astropi1
* è stato scritto con il supporto di CoderDolomiti

Crediti: molti esempi presentati nel tutorial sono pubblicati sul sito ufficiale Raspberry Pi, così come alcune delle immagini utilizzate. Sono rese disponibili dalla Raspberry Pi Foundation (Licenza CC BY-SA 4.0).

Licenza: CC-BY Alessandro Ebranati, CoderDojo Trento
Quest'opera è distribuita con Licenza Creative Commons Attribuzione 4.0 Unported